SYS-CON

**WHY IS AGILE DEVELOPMENT HARD?** PAGE 6

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

# JDJ

JDJ.SYS-CON.COM VOL.11 ISSUE:7

## Agile Java Development

with Spring, Hibernate & Eclipse

OCT 3-4 2006 AJAXWORLD CONFERENCE & EXPO

SEE PAGES 58-59 FOR DETAILS

SANTA CLARA CONVENTION CENTER

WWW.AJAXWORLDEXPO.COM

RETAILERS PLEASE DISPLAY UNTIL SEPTEMBER 30, 2006

$5.99US $6.99CAN

**PLUS...**

- Write Right Java Faster
- A JNI-bridged Java Desktop Application
- Innovative Solutions for Enterprise Developers

# Fireproof Your Code

## Prevent Java code fires with JProbe®

Constantly fighting Java code fires? Prevent infernos — before code moves into production — with award-winning JProbe from Quest Software.

Quickly pinpoint Java code hot spots with line-level analysis. Discover and debug memory leaks to dramatically improve performance. Automate the task of performing code analysis during off-peak hours. And release applications with confidence, knowing they have been fully tested. JProbe is the proactive solution that gives you higher levels of productivity and end user satisfaction.

Stop Java code performance flare ups — before they start. Improve code quality and increase application efficiency with JProbe.

Watch JProbe in action. View the new product demo at:
**www.quest.com/JavaCode**

**Application Management** | Database Management | Windows Management

QUEST SOFTWARE®

**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.

# Is There Life Beyond Google?

**Jeremy Geelan**
Group Publisher

In one of my (several) former professional lives, I used to publish books about the future, including, for example, the world's first full-length book about groupware.

That was back in 1994 and the book was called Groupware on the 21st Century. If I'd been clairvoyant, I guess I would have called it simply The Future Is Google, but the Web hadn't yet taken off, let alone Google, Inc. – mainly because Sergey Brin and Larry Page were both still only 21 years old.

Fast-forward 12 years and the landscape has changed so much. It turns out the world was neither flat, nor round, but Google-shaped. Because much of what is said and done on the Web is currently said or done via Google.

What comes after Google? Where will the Web, the Internet, the whole nexus of telecommunications, i-Technology, and the quest for a better world take us?

My strong sense is as follows: if Web 2.0, as the joke goes, is about how we can make money out of Web 1.0, then Web 3.0 is going to be about how we can extract insight out of Web 2.0.

Those who know me professionally – and a few have had to weather that particular storm for well over a quarter of a century already – will recognize my (to them) familiar refrain: that "insight capture" is the key to the 21st century, just as it was the key to the 19th, or the 20th…or the 14th, or the 16th, for that matter.

Unless we can first capture and thereafter harvest – asynchronously, as and when it is most needed and most relevant – the collective wisdom of our time, how can it be deemed "wisdom"? None of us has time any longer to attend all the conferences we'd like to, or to join all the societies or support all the causes that appeal to us for attention, time, and money. What we need above all is to be able to act co-intelligently. While co-intelligence is what we need, our actual opportunities for meaningfully interacting with our peers are in some respects growing in inverse proportion to the variety of ways in which we can execute the interaction.

We send e-mails about phone calls, make phone calls about e-mails, send IM messages about videos, write blogs about IM messaging…and send videos about there being too many ways to communicate – because, let's face it, do we have time to keep up with each other's communication stream? On a good day, barely; on a regular day, heck no!

Welcome to the World Beyond Google. In this post-Google world that I am positing, the responsibility for extracting the good from the rich new seams of inter-communication would pass in part from the individual to the collective – not quite the "Wisdom of Crowds" idea, which is more like a "broadband" version of this vision, but certainly the wisdom of many, on the basis of "none of us is a smart as all of us."

How does it work, co-intelligence? It's almost easier to say how it doesn't work. Co-intelligence begins when trying to outsmart the other guy ends. When we are proud to bring our pebble to the building site and help build the tower of perspective; we don't need to insist on being the chief architect if all that the titular folly-swaddle achieves is that what gets built instead is not a tower but a small woodshed.

Small is powerful, less is more. We need fewer ways to communicate, not more, and better ways to distill what's being communicated. There will most certainly be Life Beyond Google, but it will be insightful only if we plan for insight right now, in every piece of software we develop and every single communication and/or networking application that we build.

Social networking without some kind of insight functionality is like mashing up all the world's transport systems – the road network, the railroads, the navigable rivers, the flight paths – and then hoping it will work without the simultaneous invention and development of maps.

**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*

ALTOVA® stylevision® 2006

Transform your appearance

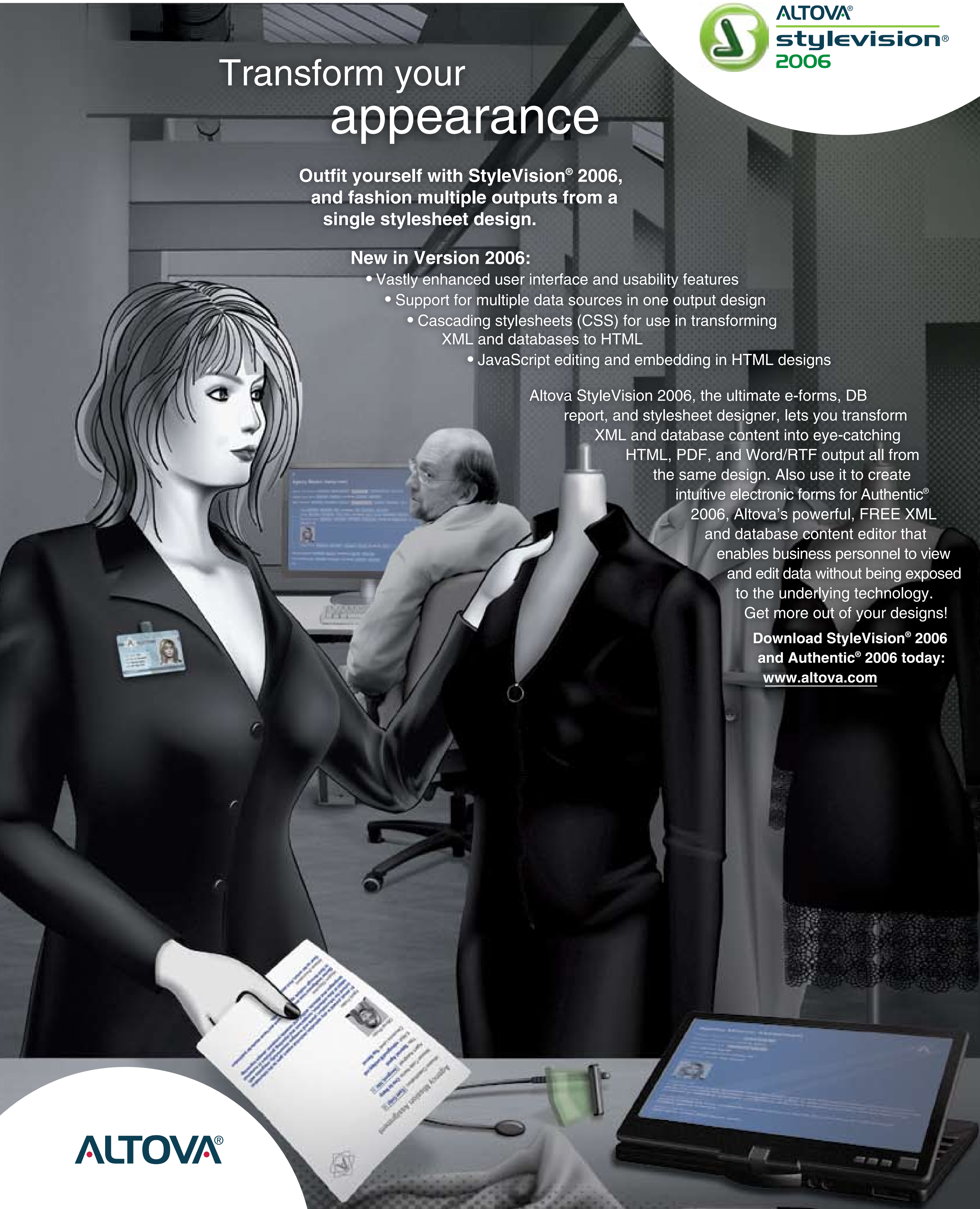
**Outfit yourself with StyleVision® 2006, and fashion multiple outputs from a single stylesheet design.**

**New in Version 2006:**

- Vastly enhanced user interface and usability features
- Support for multiple data sources in one output design
- Cascading stylesheets (CSS) for use in transforming XML and databases to HTML
- JavaScript editing and embedding in HTML designs

Altova StyleVision 2006, the ultimate e-forms, DB report, and stylesheet designer, lets you transform XML and database content into eye-catching HTML, PDF, and Word/RTF output all from the same design. Also use it to create intuitive electronic forms for Authentic® 2006, Altova's powerful, FREE XML and database content editor that enables business personnel to view and edit data without being exposed to the underlying technology. Get more out of your designs!

**Download StyleVision® 2006 and Authentic® 2006 today: www.altova.com**

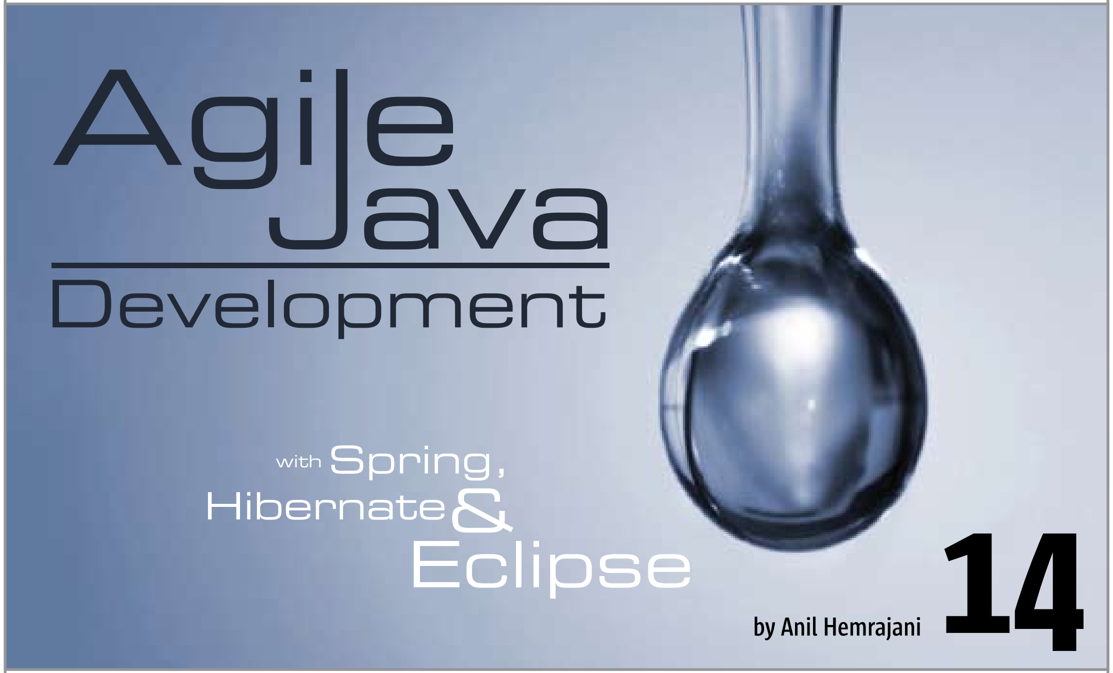


ALTOVA®

JULY 2006 VOLUME:11 ISSUE:7

# JDJ contents

## *JDJ* Cover Story



JDJ (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: JDJ, SYS-CON Publications, Inc., 135 Chestnut Ridge Road, Montvale, NJ 07645.

# Why Is Agile Development Hard?

Jon Kern

I bet you thought agile development was supposed to be easier than a traditional, prescriptive process! That I would wax evangelical that agile development is the answer to everything, and it simplifies your life. Yeah, just like UML and model-driven architecture and XML and SOA and Web services are silver bullets. Uh-huh, r-i-g-h-t.

If you aren't familiar with agile development, you can check out our manifesto here: http://www.agilemanifesto.org/. You can also learn more here: http://en.wikipedia.org/wiki/Agile_software_development. The difficulty with "agile development" is that it is "in the eye of the beholder." That is, even a highly regulated, constrained application can be conducted in an agile manner. That manner will be radically different from the way a five-member team might approach building a small desktop software product that they want to sell. Both projects can be agile, and therein lies some of the issues that make agile seem "hard."

Why, then, is agile development hard?

It might not be so much that agile development is hard, per se, depending on your perspective. The reason I give for agile development being more of a challenge for many teams is simple:

YOU HAVE TO USE YOUR BRAIN!

No, I don't mean to infer that you don't normally use your brain. However, in a highly prescriptive process, it is easy to fall into a trap of doing an activity because… well, just because! Typical reasons are that a specific set of steps are mandated by a process, possibly making sense in some projects, not in others. But often, the process grows old, people no longer remember why they are doing a specific task, but do it anyway. Folks get comfortable building some document without ever asking the recipient if it is enough, too much, useless, perfect.

**– Don't Mistake Activity for Progress**

Developers often jump into the fray by "cherry-picking" specific fun stories and sometimes missing the big picture. For an app that needs to message another app and then do some crunching, I have seen developers working on everything but the core system development needs: the messaging or shelling out of one app to speak to another app. When I asked if they got the basics working yet and are now discussing the pretty frills, they looked around, kind of sheepishly. "No, we don't have the parent app able to message our part yet." If you use your brain and step back a bit, you can see that nothing else matters.

Despite making progress on "stories" they actually had no meaningful progress. Remember, if you can't see it working, it doesn't exist!

**– Show Me Working Features**

Yeah, yeah, yeah, I know, someone didn't prioritize the stories properly. But, you certainly cannot expect a "customer" to figure out that some weird technical messaging thingy had to be in place first. The customers better only talk about the business and features that are (generally) devoid of technology words.

Development teams have to think more broadly than just coding. If a team understands that the client needs to know about some aspects of the technology that have to be implemented before anything else matters, this should be brought to their attention.

Development teams also need to help with the project management aspects. Yes, that's right. You need to learn how to say "No!" – in a gentle manner, of course.

"So, can we add some more features to this iteration?"

"Uh, since we are three days to iteration 'pencil's down,' let me think. NO! It has to wait until the next iteration."

After all, if you know in your gut that there is no way for a disruptive request to be accomplished, why bother trying? You should always maintain your iteration delivery schedule, slipping features instead of dates. You should always maintain your good habits. Someone has to be the adult ;=)

**– Learn to Say "No"**

In agile development, you (and everyone else) are charged with the rather difficult responsibility of always challenging yourself to ensure you are doing the smartest thing possible that will bring about the best solution for the current project (and within its context).

You need to set up your agile team for "running fast" through each iteration's features. To start with, I like to model the problem domain to enough of a level of understanding from which:

- Requirements/features can be written using a consistent language
- Enough of an object model exists to anchor the coding

*– continued on page 10*

SYS-CON MEDIA

President and CEO:
**Fuat Kircaali** *fuat@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*

Advertising Sales Manager:
**Megan Mussa** *megan@sys-con.com*

Associate Sales Manager:
**Kerry Mealia** *kerry@sys-con.com*
**Lauren Orsi** *lauren@sys-con.com*

**Editorial**

Executive Editor:
**Nancy Valentine** *nancy@sys-con.com*

**Production**

Lead Designer:
**Louis F. Cuffari** *louis@sys-con.com*

Art Director:
**Alex Botero** *alex@sys-con.com*

Associate Art Directors:
**Abraham Addo** *abraham@sys-con.com*
**Tami Lima** *tami@sys-con.com*

**Web Services**

Information Systems Consultant:
**Robert Diamond** *robert@sys-con.com*

Web Designers:
**Stephen Kilmurray** *stephen@sys-con.com*
**Wayne Uffleman** *wayne@sys-con.com*

**Accounting**

Financial Analyst:
**Joan LaRose** *joan@sys-con.com*

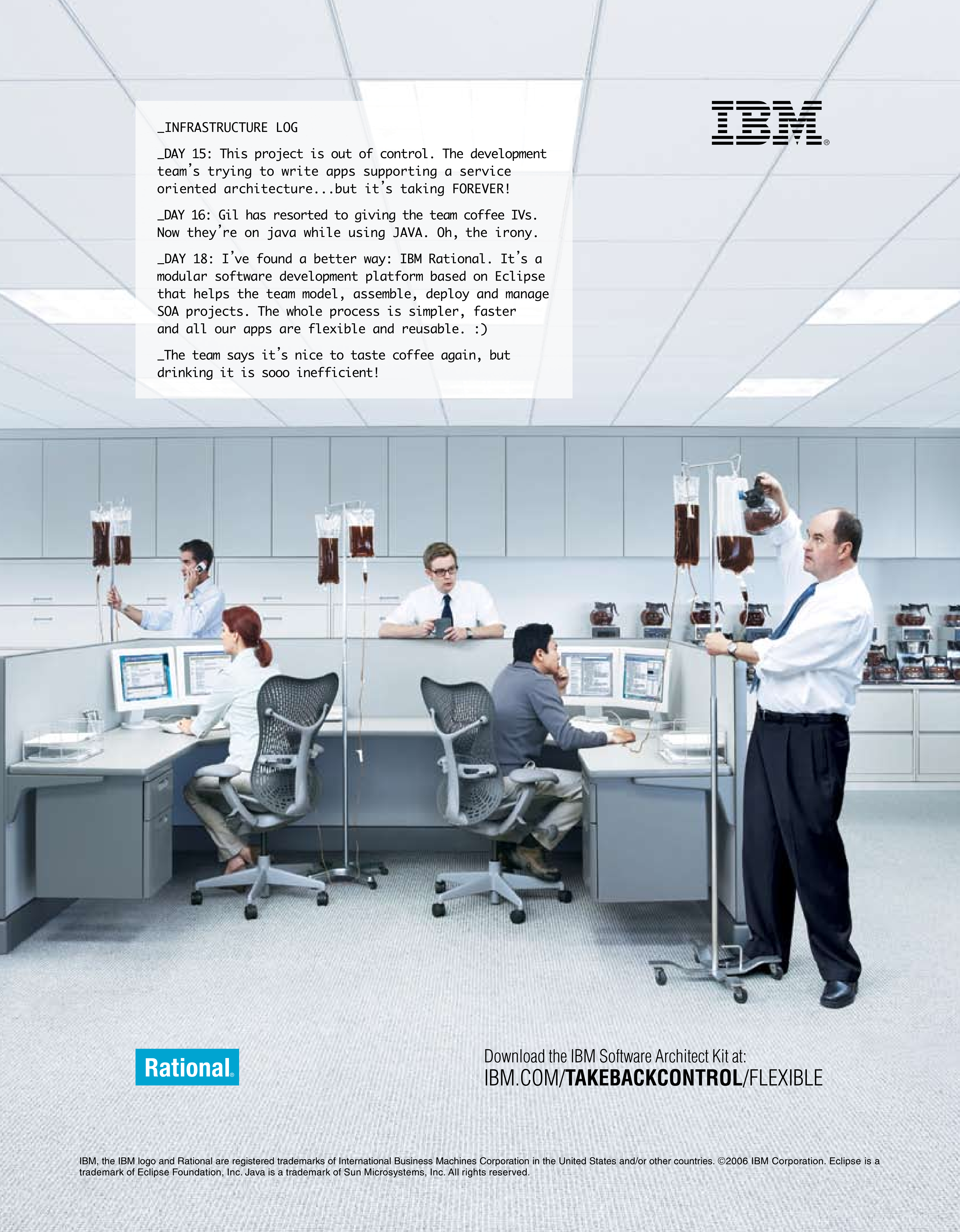
Accounts Payable:
**Betty White** *betty@sys-con.com*

Accounts Receivable:
**Gail Naples** *gailn@sys-con.com*

**Customer Relations**

Circulation Service Coordinator:
**Edna Earle Russell** *edna@sys-con.com*

JDJ Store Manager:
**Brunilda Staropoli** *bruni@sys-con.com*



_INFRASTRUCTURE LOG

_DAY 15: This project is out of control. The development team's trying to write apps supporting a service oriented architecture...but it's taking FOREVER!

_DAY 16: Gil has resorted to giving the team coffee IVs. Now they're on java while using JAVA. Oh, the irony.

_DAY 18: I've found a better way: IBM Rational. It's a modular software development platform based on Eclipse that helps the team model, assemble, deploy and manage SOA projects. The whole process is simpler, faster and all our apps are flexible and reusable. :)

_The team says it's nice to taste coffee again, but drinking it is sooo inefficient!

IBM

Rational.

Download the IBM Software Architect Kit at:
IBM.COM/**TAKEBACKCONTROL**/FLEXIBLE

# Innovative Solutions for **Enterprise Developers**

*Interview with Mike Milinkovich, executive director of the Eclipse Foundation*

**JDJ:** *What's the state of the RCP – are "rich clients" Eclipse's focal point for the future of software development?*
**Milinkovich:** Eclipse RCP is a very important strategy and future for us. We are seeing a lot of uptake, in particular ISVs, from organizations adopting RCP as the platform for building their next-generation products. That being said, Eclipse as a community is focused on a number of different areas including providing innovative solutions for enterprise Java developers. In addition, we have a leadership position in providing the platform for embedded tools development and our SOA and ALM initiatives are coming on strong.

**JDJ:** *How does portability between platforms translate into value for enterprises, assuming that this is what will drive RCP adoption.*
**Milinkovich:** The freedom of choice is the value RCP is providing enterprises and ISVs. If you are building for the .NET platform, you are pretty much committed to deploying on Windows. Eclipse RCP allows you to choose now or in the future the ability deploy on Windows, Mac, Linux, Solaris, or HP-UX, and we are working on embedded platforms like the Nokia Series 60. This is pretty compelling, especially if you are an ISV and want to have a solution for potential Mac and Linux customers.

**JDJ:** *Is SOA still a part of the Eclipse Foundation's vision of the future?*
**Milinkovich:** Absolutely. We have a top-level project called the SOA Tools Platform that is building the frameworks and exemplary tools that enable the design, configuration, assembly, deployment, monitoring, and management of software designed around a service-oriented architecture (SOA).
The interesting thing about this project is that the companies involved in the SCA are also involved in the Eclipse SOA Tools project, so there is going to be very good symmetry between these two initiatives.

**JDJ:** *How about Ajax? Where does that fit in? The Eclipse Foundation joined OpenAjax right from the get-go, for example.*
**Milinkovich:** Ajax is a natural evolution for Eclipse. Lots of people equate Eclipse with being Java but Eclipse is a lot more than just Java. Eclipse is really about being a platform for building and integrating tools. In fact ,we have counted over 20 different language IDEs built on Eclipse.

So developing an Ajax tool chain and frameworks is pretty natural for us. Specifically, we have an Ajax Toolkit Framework (ATF) project and an Ajax framework project called Rich Ajax Platform (RAP). The other interesting thing is that if you look at the wider Ajax and Rich Internet Application (RIA) community, everyone seems to be building their tools on Eclipse – Adobe, Nexaweb, and Laszlo to name just a few.

**JDJ:** *Sun's been rumbling on about possibly joining Eclipse at long last if only you'd ditch the confrontational name. (1) Would you ever change the name and (2) would Sun's joining mean the end of NetBeans?*
**Milinkovich:** I really believe there are two platforms now: Eclipse and Visual Studio .NET. I have said this before; we would love to have Sun join Eclipse. There is lots of room for NetBeans and I can easily see it thriving on the Eclipse platform. Changing the name is really not an option any longer.

**JDJ:** *What do you think are the top three open source issues right now?*
**Milinkovich:** I used to be a product manager in a former life, and to me what the open source community as a whole needs to think of is how to provide enterprises with the "whole product." What I mean by that is just providing the executable bits is not enough. Enterprises need services, technical support, training, knowledge transfer, etc., as well as the source and binary code. In other words, the open source community in its entirety will need to provide all of these various pieces to enterprises before becoming truly mainstream.

Along with this, enterprises want to source their technology from reliable and predictable providers. Open source communities need to demonstrate that they can be transparent and predictable in their technology roadmaps. This is one area that we at Eclipse spend a great deal of time and effort working on.

The third area is that the open source community needs to figure out how to inspire enterprises to contribute back to the community. Not out of altruism, but because the corporations are shown that there are compelling business reasons to do so. To me, that seems like one of the obvious paths for both growth and collaboration by the open source community.

– continued from page 6

- Enough complexity has been uncovered to make the cost/time estimate defensible

In addition, you need to understand the architectural approach (not to mention coding guidelines). You can slowly arrive at the ultimate architecture and call it refactoring. But at what cost? I like to get the bulk of the architecture design/building work out of the way before starting the first iteration. Of course, for some apps, you may already have the architecture style predetermined.

I like to ensure the team can hit the ground running with a list of features in hand, architecture, coding guidelines, automated build scripts, and a domain model from which to hang code.

Sure you can discover all of this piecemeal as you go along, but that is usually slower and less efficient than doing some work up-front to lay the groundwork. No, I am not talking about "Big Stuff Up Front" (BDUF/BRUF) type of an approach. I am talking about using your brain. Do enough up-front work to enable running fast (even with scissors). Maybe: Just Enuf Design Up-front (JEDI – if I substitute "Initially" <g>).

**– You Must Lay the Proper Groundwork to Be Agile**

By a few iterations in, if you are not ripping through the feature list/user stories almost faster than they can be compiled, you are not yet performing at a truly agile level. If it is "disruptive" that a customer changes the stories scheduled for the next iteration because of changing priorities, you are not yet performing at a truly agile level. If you and your team are not constantly using your brains, you are not yet in the agile state of mind.

**– Agile Is a State of Mind!**

Post any comments on my blog: http://www.compuware.com/blogs/jkern/

**Jon Kern** is a software engineering evangelist, Agile Manifesto co-author, speaker, and author. His experience is wide-ranging across varied problem domains and technology platforms. From jet engine R&D (he's an aerospace engineer, after all) to real-time flight simulator design and development, from TogetherSoft's and OptimalJ's commercially successful modeling tools to building IBM's Manufacturing Execution System software – Jon has seen and done a lot in his 20 years.
jon.kern@compuware.com

## Innovations by InterSystems

*Rapid development with robust objects*

*Lightning speed with a multidimensional engine*

*Easy database administration*

*Massive scalability on minimal hardware*

## The Objects Of Your Desire.

Caché is the first multidimensional database for transaction processing and real-time analytics. Its post-relational technology combines robust objects and robust SQL, thus eliminating object-relational mapping. It delivers massive scalability on minimal hardware, requires little administration, and incorporates 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 the license for a full refund if you are unhappy for any reason.** Caché is available for Unix, Linux, Windows, Mac OS X, and OpenVMS – and it's deployed on more than 100,000 systems ranging from two to over 50,000 users. We are InterSystems, a global software company with a track record of innovation for more than 25 years.

InterSystems CACHÉ®

Try an innovative database for free: Download a fully functional, non-expiring copy of Caché, or request it on CD, at **www.InterSystems.com/Cache17P**

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

# A Look at the Eclipse Callisto Release

*Providing a more transparent and predictable development cycle*

by Chris Aniszczyk
& Gunnar Wagenknecht

Callisto is the simultaneous release of 10 major Eclipse projects at the same time. An important thing to note about Callisto is that even though it's the simultaneous release of 10 projects, it doesn't mean these projects are unified. Each one remains a separate Open Source project operating with its own project leadership, its own committers, and its own development plan. In the end, Callisto is about improving the productivity of developers working on top of Eclipse projects by providing a more transparent and predictable development cycle.

## A Quick Tour of Callisto's Projects

In this article, we 'll go through each of the Callisto components. We'll give a brief overview of each and quote an Eclipse committer about what's exciting about his component in the Callisto release. Then we'll discuss some of the challenges that faced Callisto and conclude with the advantages gained by adopting Callisto. As you soak in what the committers have to say, remember that they are from the various corporations working together to make Callisto a reality.

## Platform

The Eclipse Platform component (http://www.eclipse.org/platform) is the heart of Eclipse and has three main pieces:

- ***Java Development Tools (JDT)*** – http://www.eclipse.org/jdt - When most people think of Eclipse, this is the first component they think of. Eclipse provides a world-class Java development environment.
- ***User Interface/Core Tooling*** – This piece encompasses many smaller components in the Platform. It's responsible for all the visuals you see in Eclipse and features like team integration and Ant support.
- ***Plug-in Development Environment (PDE)*** – http://www.eclipse.org/pde - Have you ever used a wizard in Eclipse to create an Eclipse plug-in? If you have, you used the PDE. It's responsible for all the tooling in plug-in development.

Since it's hard to track down all the committers for each of the small Platform projects, we'll focus on what PDE has to offer Callisto:

*"For the Callisto release, PDE provides comprehensive OSGi tooling, which would make it an ideal development environment for component programming, not just Eclipse plug-in development. Other noteworthy features include quick fixes in plug-in manifest files, NLS tooling, and tighter integration with JDT via participation in search and refactoring."*

**- Wassim Melhem,
PDE lead, IBM**

## C/C++ Development Tools (CDT)

*http://www.eclipse.org/cdt*

Did you know Eclipse isn't just for Java development? The CDT project aims to bring a fully functional C and C++ development environment to the Eclipse Platform. One should note that CDT can scale. A famous CDT demo is to import the Mozilla code base and use CDT to develop it.

*"The CDT brings Callisto a development environment for writing C and C++ programs. The JDT sets a high bar as far as Eclipse IDEs go and we are constantly working in catch-up mode. For Callisto, the CDT provides an editor with all your regular text editor features such as language-specific keyword highlighting and content assist. It also provides an index of the user's code to provide search and code navigation features. There's also a framework for integrating build tools and debuggers to complete the edit-build-debug cycle. In this release, we've focused on a faster, more scalable indexing framework as well as a flexible build system that allows for per-resource builds as well as a new experimental internal builder that eliminates the need for MAKE files. We also have the beginnings of a framework for supporting additional compiled languages such as Fortran by the Photran project and hopefully more such as C# and Ada in the future.*

**- Doug Schaefer,
CDT lead, QNX Software Systems**

## Business Intelligence & Reporting Tools (BIRT

*http://www.eclipse.org/birt*

The BIRT project strives to bring a Eclipse-based reporting system that integrates with your application to produce compelling reports for both Web and PDF. BIRT provides core reporting features such as a graphical report designer, data access, and scripting support. BIRT reminds me of Crystal Reports or JasperReports, but tightly integrated with Eclipse.

*"With the Callisto release, BIRT expands on the themes of scaling, broader appeal, and simplicity. Some of the new features include Re-portlet support, which allows elements of a BIRT report to be rendered as partial HTML pages for better integration into dash boarding-type applications, joined datasets for combining disperse data*

*sources into a single table, improved DTP integration, parameterized XML data sources, the ability to template an existing report design, and several chart enhancements. BIRT 2.1 will also provide better tooling to promote developed reports and ancillary files between environments.*

**- Jason Weatherby, BIRT evangelist, Actuate Corporation**

## Data Tools Platform (DTP)

*http://www.eclipse.org/dtp*

DTP project includes extensible frameworks and exemplary tools around data-centric technologies. DTP provides data management frameworks and tools not biased toward any vendor. If you plan to work with databases and use Eclipse, this should be your first stop for database tooling.

*"The Eclipse Data Tools Platform (DTP) brings a number of key data-centric frameworks and tools to the Callisto feature set. Using these DTP frameworks and the examples provided for Apache Derby, the extender community can quickly achieve a high-functionality baseline working with heterogeneous data sources. Once this baseline is attained, specialized offerings for data-centric applications can then be created in the familiar Eclipse Plug-in Development Environment (PDE), allowing developers to leverage existing skills for the data domain."*

**- John Graham, DTP lead, Sybase Corporation**

## Eclipse Modeling Framework (EMF)

*http://www.eclipse.org/emf*

EMF is a modeling framework and code generation tool for building tools and other applications based on a structured model. To put it simply, EMF lets you build models quickly by taking advantage of EMF facilities. For example, one feature EMF provides is support for persisting models to XML (there are options to persist models to databases too).

*"The Eclipse Modeling Framework provides powerful generative and runtime capabilities for applications based on structured data models. From a simple class diagram or XML Schema, you can generate a complete Java implementation of the model, along with an editor for it, and take advantage of EMF's facilities for persistence, notification, validation, and change recording in your application. Callisto includes EMF 2.2, which introduces many exciting new features: a simplified XMLProcessor API for XML persistence; cross-resource containment support; new code generation patterns, allowing, for instance, for all signs of EMF to be suppressed from generated interfaces, or for no interfaces to be generated at all; encryption support in resources; improved XML Schema generation and round-tripping; an extensible model exporter tool; an improved, extensible code generator; and various performance improvements and usability enhancements.*

**- David Steinberg, EMF committer, IBM**

## Graphical Editing Framework (GEF)

*http://www.eclipse.org/gef*

GEF serves as the base for graphical applications in Eclipse. It includes Draw2D (similar to Java2D), which is a lightweight graphical toolkit built on SWT. GEF itself is a framework that extends the Model-View-Controller paradigm to graphical editors. GEF brings your own model to the framework and provides facilities that take advantage of Draw2D to paint your figures.

*"[For the Callisto release] GEF 3.2 is essentially a maintenance release in terms of features and bug fixes. Some minor features that were integrated were for supporting animated layout and general fixes to direct graph layout algorithm..."*

**- Steven Shaw, GEF/GMF committer, IBM**

## Graphical Modeling Framework (GMF)

*http://www.eclipse.org/gmf*

GMF is a new Eclipse project that aims to bridge EMF and GEF to allow for the generation of graphical editors.

*"GMF brings Callisto a more efficient means for Eclipse developers to create graphical editors based on EMF and GEF. Based on model-driven development techniques, GMF leverages a series of models to generate editors targeting the feature-rich GMF diagramming runtime, which can also be used in the absence of the generative framework for the creation of high-quality editors. Follow the GMF Tutorial cheat sheet and online tutorial to get started."*

**- Richard Gronback, GMF lead, Borland**

## Test & Performance Tools Platform (TPTP)

*http://www.eclipse.org/tptp*

TPTP provides an open platform supplying powerful frameworks and services that allow software developers to build unique test and performance tools, both Open Source and commercial, that can be easily integrated with the platform and with other tools. The platform supports a broad spectrum of computing systems including embedded, standalone, enterprise, and high-performance and will continue to expand support to encompass the widest possible range of systems.

*"TPTP provides a rich set of test, profiling, and monitoring tools. However its true value can only be realized by being part of a core typical user use case. By integrating with the WTP project and providing a 'profile on server' action TPTP becomes an easy link to collecting and analyzing your Web application performance characteristics. By further providing the ability to function and load test based on http requests TPTP helps the developer prove the quality of the Web application. Finally by providing customized extended reporting of the rich data TPTP collects with the use of BIRT the user can get the test and performance data they want and need to best manage their own project.*

**- Harm Sluiman, TPTP committer, IBM**

## WebTools Platform (WTP)

*http://www.eclipse.org/webtools*

The WTP Project extends the Eclipse Platform with tools for developing J2EE Web applications. The WTP project includes source editors for HTML, JavaScript, CSS, JSP, SQL, XML, DTD, XSD, and WSDL; graphical editors for XSD and WSDL; J2EE project natures, builders, and models, and a J2EE navigator; a Web Service wizard and explorer, and WS-I Test Tools; and database access and query tools and models.

*"WTP's 1.5 release in the Callisto train will include several new features and a number of stability and performance enhancements. Users of WTP Web Services will appreciate the upgrade to Axis 1.3 and streamlined Web Service and client wizards. XML Schema and WSDL graphical views have also been revamped to make them easier to navigate and read. WTP tackled some major infrastructure work in the Callisto release, moving to the platform's common navigator and undo stacks. The tabbed property support is also transitioning from WTP-only to the platform level in this release. Finally,*

"From the user's perspective Callisto radically changes the way Eclipse **and the participating Eclipse projects get on the desktop"**

*the Dali and JSF projects are planning to do a technology preview around the Callisto timeframe and will provide some exciting 'first looks' at Java EE 5 tooling support that will preview support in WTP 2.0."*

**- Tim Wagner, WTP PMC lead, BEA**

## Visual Editor Project (VE)

*http://www.eclipse.org/ve*

Ever wondered if there was a way to create user interfaces visually, using the simple semantics of drag-and-drop? The Eclipse project provides VE, which is a open development platform for supplying frameworks to create GUI builders. VE has two exemplary implementations of Swing/JFC and SWT/RCP.

*"The Visual Editor project (http://www.eclipse.org/vep/) adds the ability to visually develop SWT and Swing user interfaces in Callisto. The main focus of this release was to add initial support for creating Rich Client Platform (RCP) components with the VE. Towards that goal we've added: the ability to develop Views and Editors visually, support for the Forms UI toolkit, and the ability to work with JFace viewers. Other notable new features in this release include significantly enhanced tooling for SWT's GridLayout and support for VE on the Mac OS X platform."*

**- Jeff Myers, VE committer, IBM**

## Callisto's Challenges

There are two main challenges with Callisto. The first one and for many people the most obvious one is developing Callisto. Aligning 10 large projects for simultaneous release is very challenging. But once you actually get the release, you have to deliver it and that's a challenge on its own.

The method of choice for delivering Callisto is the Eclipse built-in update mechanism. So you only have to download the Eclipse Platform binary for your system and then you start Eclipse, use the Update Manager to visit the Callisto Update Site, and select the Callisto features you'd like to have installed in your environment. The Eclipse Update Manager will do the rest for you.

You can imagine that this will put a burden on a single update site (in terms of bandwidth use). In Eclipse 3.2, the Update Manager and the Eclipse.org infrastructure were enhanced to deliver Callisto. The goal for the Update Manager was to reduce the volume of data that's transferred and the goal for the Eclipse.org infrastructure was to create a reliable mirroring story for the Callisto Update Site.

## Callisto's Advantages

Callisto brings several advantages to users and plug-in developers (adopters) of Callisto projects. Let's start with the user's perspective.

## The User's Perspective

From the user's perspective Callisto radically changes the way Eclipse and the participating Eclipse projects get on the desktop. It takes away the need to read through the requirements sections and collect them manually from several download pages. You just download one Platform binary and select your desired projects from the Callisto Update Site after installing and starting the Platform binary.

**Q: Which projects does WTP depend on?**

*A: Who cares. The Eclipse Update Manager will handle this.*

Callisto also has another great advantage for Eclipse users. It creates some kind of accountability for all participating projects and their committers. Because Callisto creates a reference platform of Eclipse projects that are intended to work together. And if they don't now it's easier to report cross-project issues because you only need to reference the Callisto platform instead of collecting all dependencies.

## The Developer's Perspective

From a developer and adopter's perspective, Callisto introduces stability (in terms of dependencies and investments). Before Callisto, it was up to you to select the projects you'd like to depend on. But often the result was disappointing because of some incompatible dependency conflicts. Now with Callisto the dependencies are clearly defined.

With clearly defined dependencies you get a target platform that will be valid and current for a long time. So Callisto also ensures that the investment you put in your adoptions are well spent in the long term.

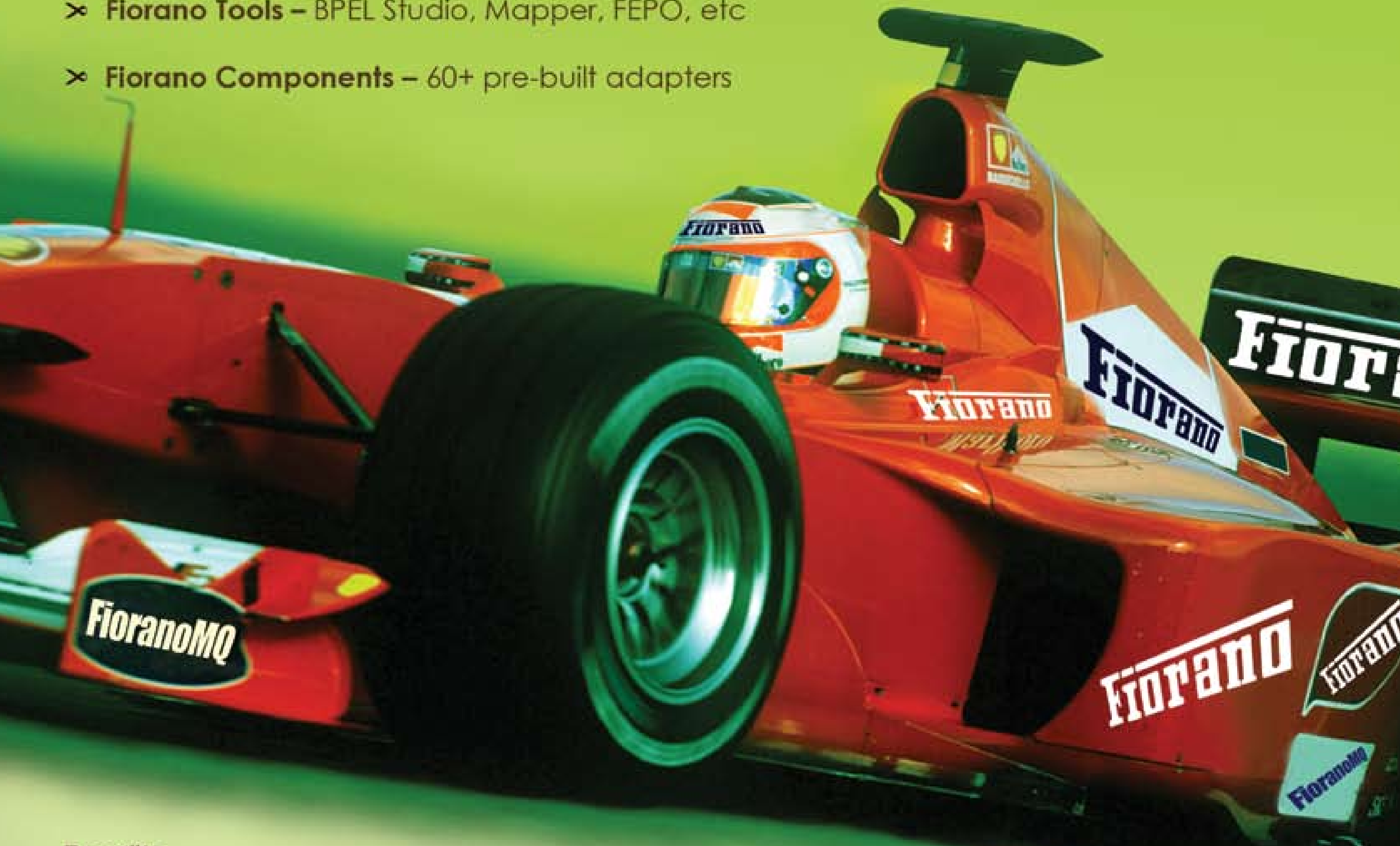
## Conclusion

On the whole, we hope you enjoyed this quick tour of Callisto and some of the challenges Callisto faced. We think Callisto will make it easier for end users to tailor their Eclipse experience by selecting what they want included in their Eclipse installation. Now, the only logical thing to do is give Callisto a try. See http://www.eclipse.org/callisto.

# Fiorano SOA™ 2006

## The Quickest path to an SOA

- **FioranoMQ™ 2006 –** world's fastest, most scalable JMS
- **Fiorano ESB™ 2006 –** CAD/CAM for distributed applications
- **Fiorano BPEL Server –** simplifying business process orchestration
- **Fiorano Tools –** BPEL Studio, Mapper, FEPO, etc
- **Fiorano Components –** 60+ pre-built adapters



**Benefits**

- **Adherence to popular industry standards** - JMS, COM, .NET, JCA, JMX, BPEL, SOAP, etc.
- **Multi-language, Multi-platform, Multi-protocol**
- **Unmatched Scalability and High Performance**
- **Quick, Measurable ROI**

**Download your copy of Fiorano today!**

www.fiorano.com/downloadsoa

Fiorano®

Enabling Change at the speed of thought

# Agile Java Development

## with Spring, Hibernate & Eclipse

*A roadmap for building enterprise-class applications using agile methods and POJOs*

**by Anil Hemrajani**

After getting a head of gray hairs and a quickly receding hairline, I have learned that the simplest solutions are often the best. Having worked with Java since 1995 and various software development lifecycle methodologies over the years, I have seen things grow complex in these areas. Thanks to some new lighter-weight Java tools and agile methods, I can provide a fresh perspective on developing Java applications in an agile manner.

This article is different from typical Java articles for two reasons. First, instead of providing in-depth details on some API or cool tool, it provides a roadmap for building enterprise-class Java applications using agile methods and plain old Java objects (POJOs). Second, it covers a lot of ground, from conceptualization through deployment, so for the sake of brevity, there are minimal code excerpts; however, there's a completely functional sample timesheet application called Time Expression (with source code) built using Spring, Hibernate, Junit, and Ant available at http://visualpatterns.com/resources.jsp.

We have a lot to cover so let's get started.

### Agile Manifesto

In 2001, 17 software experts (including Martin Fowler, Kent Beck, and Jon Kern) got together to discuss lightweight approaches to software development; they jointly defined the term agile. The outcome of this was the "Manifesto for Agile Software Development," a set of values and principles for

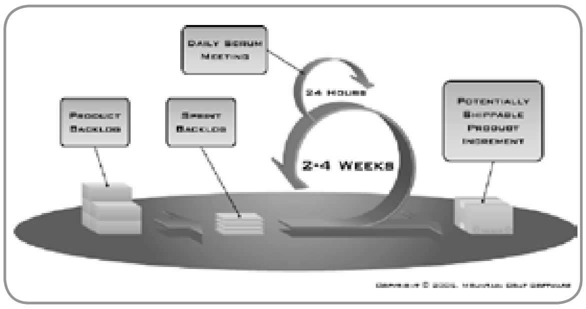


**Figure 1** Scrum (Source: mountaingoatsoftware.com)

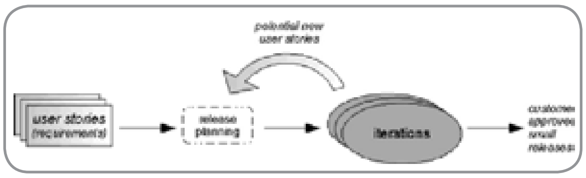

potential new user stories
user stories (requirements)
release planning
iterations
customer approves small releases

**Figure 2**

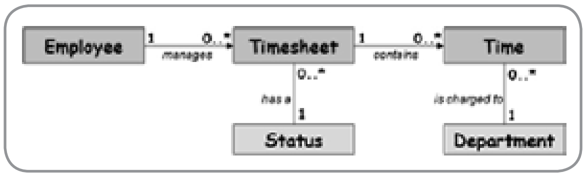
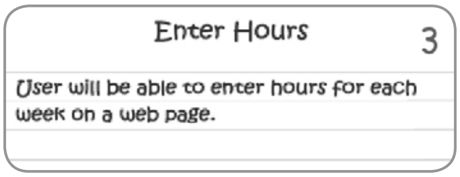


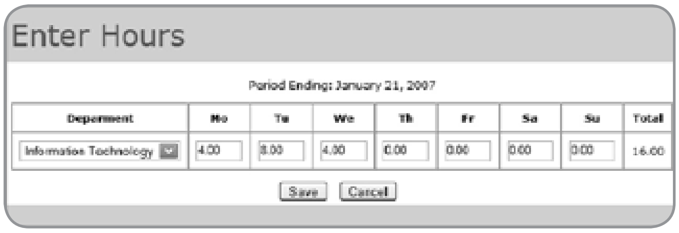
**Figure 3**



Enter Hours 3

User will be able to enter hours for each week on a web page.

**Figure 4**



Enter Hours

Period Ending: January 21, 2007

| Department | Mo | Tu | We | Th | Fr | Sa | Su | Total |
|---|---|---|---|---|---|---|---|---|
| Information Technology | 4.00 | 8.00 | 4.00 | 0.00 | 0.00 | 0.00 | 0.00 | 16.00 |

**Figure 5**

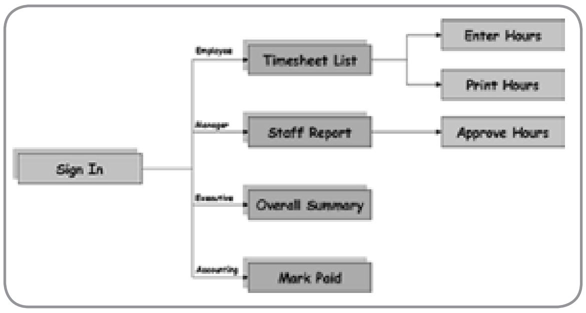


**Figure 6**

these agile methods.

The term agile incorporates a wide range of methods; some of them include Extreme Programming (XP), Scrum, Feature Driven Development, Agile Modeling, and Crystal. Many methodologies tend to include both process and modeling since they often go hand-in-hand; we will look at both next. For details on the Agile Manifesto and various agile methods, visit the agilemanifesto.org and agilealliance.org Web sites.

## Agile Processes

One of easiest agile processes to understand is Scrum. While XP tends to steal the limelight in the agile community, it's a bit more involved than Scrum. However, the two are highly complementary since XP provides a set of excellent engineering practices whereas Scrum is more about product/project management. In fact, these days I tend to recommend becoming "agile" by bringing in Scrum first, then adding XP practices one at a time, as and when needed since moving entirely to XP-based development (from waterfall) is a rude awakening for many organizations and requires a fundamental mind shift that many projects aren't ready for.

So, how does Scrum work? Simple. We gather a list of new features or change requests for an application in a product backlog. For our sample application, Time Expression, these could include:

- Hourly employees will be able to sign in to a Web application and enter their hours for each day of a given week.
- The employee's manager must approve the timesheet.
- After a timesheet is approved or disapproved, notification is sent to the employee indicating the updated status of the timesheet.
- And so on…

From here, we simply take the highest-priority features, move them to a sprint backlog, and implement them in one-month (or shorter) iterations called sprints, and continue having monthly sprints till all the features are implemented. Each sprint (or iteration) contains the entire software lifecycle, in other words, detailed requirements/analysis, design, coding, unit/acceptance testing, and deployment of production-ready code. Scrum also suggests having a planning meeting at the beginning of a sprint and a review at the end of the sprint to discuss lessons learned or the next set of features to implement in the following sprint. Other than that, we have a short daily meeting (say, 15 minutes) to discuss the project's status. Figure 1 depicts the Scrum process. Visit controlchaos.com for details on Scrum.

A common theme of agile processes is iterative development. For example, XP works like Scrum, however, it uses the concept of quarterly releases with weekly iterations as shown in Figure 2. Also the features are provided in the form of user stories, typically written by the customer using one to three lines to describe the feature. My explanation of XP is overly simplified; there's a lot more to it such as pair programming, sit together, and continuous build. Visit extremeprogramming.org for details on XP.

So now we've looked at two agile processes, Scrum and XP.

These help in gathering user feature requests and overall project management. However, as developers, we need to implement features by engineering them into software applications, so let's look at agile modeling techniques next, which can help us bridge the gap between user requirements and coding.

## Agile Design

According to thefreedictionary.com, a model is "a preliminary work or construction that serves as a plan from which a final product is to be made...used in testing or perfecting a final product." So here I'll use the word "model" to describe diagrams and other artifacts.

### *Agile Model-Driven Development*

Agile Model-Driven Development (AMDD) created by Scott Ambler provides guidelines for effective modeling. Instead of creating extensive models, AMDD recommends creating "good enough" models. One of my favorite Scott quotes is "your goal is to build a shared understanding, it isn't to write detailed documentation."

AMDD suggests two categories of models, requirements and architecture. Requirements models could include a domain model (Figure 3), usage models such as user stories or use cases (Figure 4), and UI models such as prototypes and flow map (Figures 5 and 6).

Architecture models could include a freeform model like the one shown in Figure 7.

There really isn't a whole lot more to AMDD since it provides minimal guidelines for agile modeling. Visit agilemodeling.com for more details.

### *Agile Draw*

Before moving to the next topic, let me briefly mention a new and elegantly simple technique called Agile Draw, which was used to draw Figures 3-7. This technique provides an alternative to the heavy-handed Unified Modeling Language (UML) but can also be used to complement UML. Agile Draw provides minimum guidelines for modeling and additional guidelines for adding appeal to these models using graphic design concepts. The core concepts behind Agile Draw include four basic components that make it a virtually notation-free modeling technique; these concepts include circles, boxes, lines, and text. Using them you can draw practically any model by hand or with a drawing program.

Visit agiledraw.org for more details.

### *Refactoring*

One of the core aspects of agile methods is not to do too much design upfront so you can start showing results to the customer quickly by developing actual software versus producing comprehensive documentation that no one actually reads or maintains. Of course, the downside is that it cuts down on the amount of design done for an application. However, this isn't necessarily a bad thing since most programmers find better ways of doing things once they begin coding.

For example, we find cleaner ways of structuring our code after the first pass at it, perhaps by improving our own design or because we learned a better way of using a framework (such as Hibernate or Spring). This code improvement is known as refactoring and is considered a continuous design activity.

Refactoring is more than fluff; it's now appearing as a menu option in integrated development environments such as Eclipse and IntelliJ IDEA. Visit Martin Fowler's Web site, refactoring.com, for more information on refactoring along with a catalog of many refactoring techniques.

### *Other Design Considerations*

While refactoring can help improve code, there might be other things you should consider upfront or in the first couple of iterations. Some of these include schemes for transaction management, exception handling, clustering, and application security (authentication, authorization, and encryption). Any enterprise-class project that doesn't at least consider these "big picture factors" upfront is asking for trouble later. One common problem found in XP projects is that a lot left for refactoring later never happens. Drawing bare-minimum architecture models like the one shown in Figure 7 upfront can help with general discussions about the important design considerations I've mentioned here.

## Agile Java Development

Now that we've discussed agile processes and modeling, we are ready to begin coding. However, before coding can begin, getting the environment set up is important, so let's look at that first before discussing Hibernate, Spring, and other technologies.

### *Environment Setup: JDK, Ant, JUnit, and Version Control*

Before Java application development can begin, some minimal tools are required such as the Java Standard Edition Development Kit, a build tool like Apache Ant, and an important tool for agile development, a unit-testing framework such as JUnit.

#### Apache Ant

Ant is commonly used to build Java applications, however, it's much more than a build tool. For example, some commonly found Ant tasks include javac, copy, delete, move, junit, cvs, ftp, mail, exec, and sleep – these can be used for everything from file management to code compilation to e-mailing. You can even write your own custom tags and it should be no surprise that there are many Open Source and commercial Ant tasks available.

For details on Ant, visit the ant.apache.org.

#### JUnit and Test Driven Development (TDD)

Erich Gamma (Gang of Four, Design Patterns book) and Kent Beck originally wrote JUnit. JUnit classes provide various assert methods (for example, assertTrue and assertNull) that let you test the expected results. JUnit is a simple framework but a powerful unit-testing tool. When combined with Test-Driven Development (TDD; testdriven.com), a

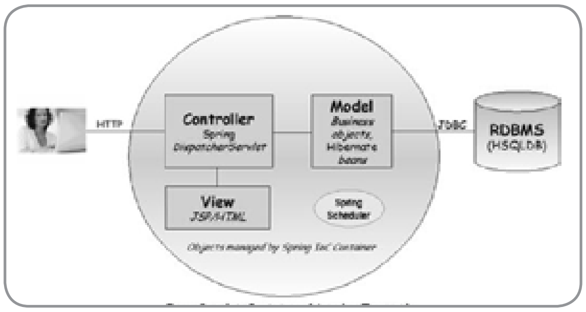
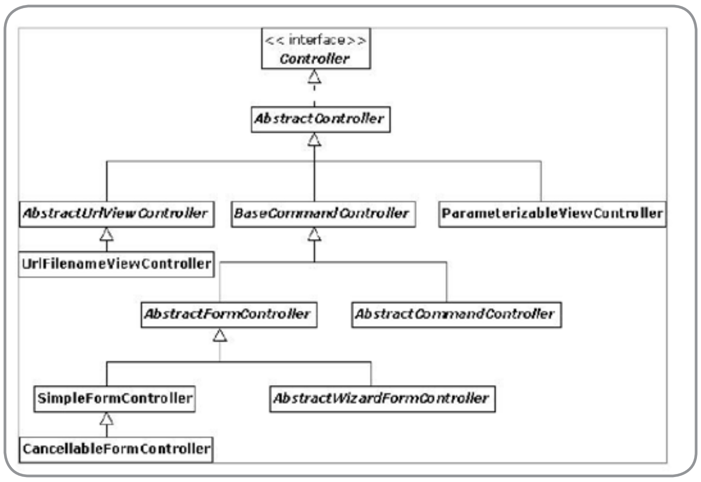


Figure 7


<< interface>>
Controller
AbstractController
AbstractUrlViewController
BaseCommandController
ParameterizableViewController
UrlFilenameViewController
AbstractFormController
AbstractCommandController
SimpleFormController
AbstractWizardFormController
CancellableFormController

method created by Kent Beck, it can significantly help you write better, cleaner, stabler code. TDD recommends that you write your test code before writing your actual code – this is a fundamental mind shift but one I recommend you investigate further.

For details on JUnit, visit the junit.org.

## Version Control, Naming Standards and More

Finally, getting a development directory structure established, class/file naming conventions defined, and version control software in place such as Concurrent Versions System (CVS) are crucial steps that a development team should take to be highly effective.

## Developing Our Data Tier with Hibernate

Relational databases and object-oriented technologies have been with us for a while now and it appears they are here to stay for the foreseeable future. Given the fact that Java developers typically work with both technologies, JDBC is often used to write the mapping code in data access objects (DAOs) that can be used to fetch and save the data – passing data back and forth is typically handled using data transfer objects (DTOs). An alternate approach to using JDBC is to use EJB entity beans, however, before EJB 3.0, they would have been considered heavy-handed since they were remote objects. So, what do you do if you want an agile approach to Java persistence using POJOs? One answer is an ORM framework such as Hibernate.

Object-relational mapping (ORM) code eliminates the need for writing JDBC calls by hand, lets you do the mapping in XML files and then simply work with database records as POJOs. Hibernate is a popular ORM framework widely used in the Java community. In fact, due to its popularity, the EJB 3.0 specification follows the Hibernate model very closely.

Hibernate supports a dozen or so relational databases (through its dialect classes). To use it, we first need to configure the database connection, typically in a file called hibernate.cfg.xml. Then, for each table to be used, we'd typically map the table to a Java class. For example, the following line shows a sample mapping for a table called Department (in a file called Department.hbm.xml) that maps to a Java class called Department as well.

```
<class name="com.visualpatterns.timex.model.Department"
       table="Department">
```

This next line shows mappings for a couple of the columns, with departmentCode being a primary key in database terms and a unique object identifier in Hibernate/ORM terms:

```
<id name="departmentCode" column="departmentCode">
<property name="name" column="name"/>
```

Once the database and mappings have been properly configured, we can simply obtain a Hibernate Session (essentially a JDBC database connection) from a SessionFactory and work with the record as a Java object, as demonstrated in the code below, which fetches a Department record with a departmentCode of "IT":

```
Session session = sessionFactory.getCurrentSession();
department = (Department) session.get(Department.class, "IT");
```

As you might guess, Hibernate also provides methods to save and delete database records (objects). Some of these methods include save, load, get, update, merge, saveOrUpdate, and delete.

One of Hibernate's most powerful features is its Hibernate Query Language (HQL). This is a SQL-like language and is extremely robust since it supports such things as joins, aggregate functions, parameter substitution, expressions, and sorting. The extremely simple example below demonstrates how we can fetch a java.util.List of objects from the Department table:

```
departmentList = session.createQuery("from Department ORDER BY
name").list();
```

We've merely scratched the surface here since Hibernate provides many more features; some of them include record locking, associations, native queries, stored procedure sup-

Register for an Online Webinar
www.opnet.com/pinpoint

STILL SEARCHING ?

MAKE ANSWERS TO PERFORMANCE PROBLEMS COME TO YOU.

**OPNET Panorama** offers powerful analytics for rapid troubleshooting of complex J2EE/.NET applications. Panorama quickly identifies how application, web, and database servers are impacting end-to-end performance. With Panorama, you can pinpoint the source of a problem, so time and money aren't spent in the wrong places.

*The world's most successful organizations rely on OPNET's advanced analytics for networks, servers, and applications.*

OPNET Panorama™
Real-Time Application Analytics

**www.opnet.com/pinpoint**

OPNET®
Making Networks and Applications Perform™

**OPNET Technologies, Inc.** 7255 Woodmont Avenue, Bethesda, Maryland 20814 phone: (240) 497-3000 • e-mail: info@opnet.com • NASDAQ: OPNT

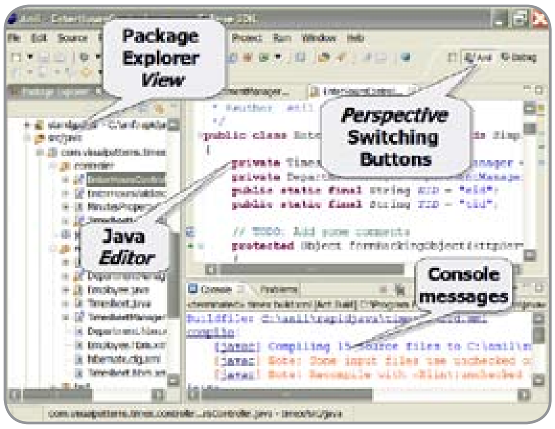


Figure 9

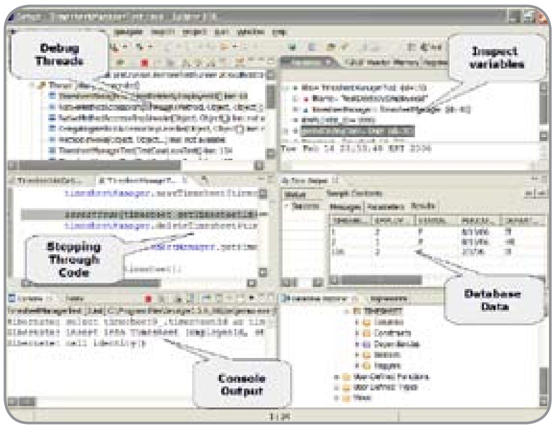


Figure 10

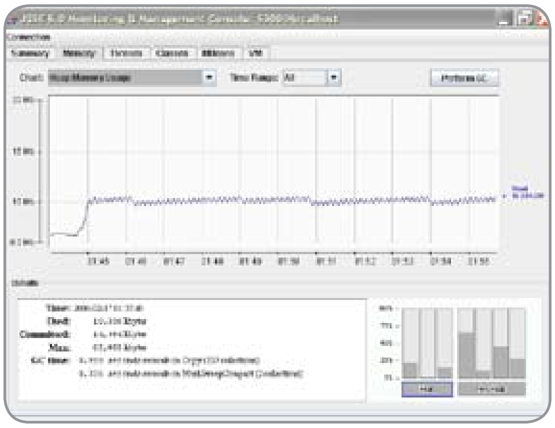


Figure 11

port, scrollable iterations, interceptors, and filters.

Visit hibernate.org for more details.

## Developing Our Web Tier with Spring

Spring (springframework.org) is one of those framework that is almost impossible to describe in one short sentence because it does so much. For example, it supports IoC or inversion of control (martinfowler.com/articles/injection.html), a complete Web MVC framework, JDBC, ORM, JEE/Web Services, aspect-oriented programming (AOP), declarative transaction management, job scheduling, mail, and more.

Using Spring provides several benefits like easier and cleaner unit testing, the ability to use POJOs in lightweight containers (say, Apache Tomcat) with enterprise services such as declarative transaction management, convenient data access, and consistent data exception handling through ORM and JDBC integration, and job scheduling in a Web/application server.

### *The Spring Web MVC Framework*

The Spring Web MVC Framework (or simply Spring MVC) is a robust, flexible, well-designed framework for rapidly developing Web applications using the MVC design pattern. The benefits to using this Spring module are similar to those you get from the rest of the Spring Framework; however, one additional and very key benefit is the ability to bind directly to business objects unlike other frameworks that require you to extend special sub-classes. Let's review some Java and configuration-related concepts for this framework.

#### Spring MVC Java Concepts

The key Java concepts in Spring MVC are:

- Controller
- Model and view object
- Command (form-backing) object
- Validator object
- Tag libraries

One of the good features of Spring MVC is that it provides a large number of controller classes to choose from (see Figure 8). Of course, this can be a bad thing when you're trying to learn this framework because deciding which to use can be a minor challenge. For example, I tend to use SimpleForm-Controller for HTML forms and UrlFilenameViewController when I don't need a controller. In some cases, I simply implement the Controller interface when I want a no-forms controller.

Many of the key GET- and POST-related Spring controller methods return a ModelAndView object that can contain model-related data and the name of a view (or reference to a view object). For controller classes that support HTML forms, we can have optional command and validator objects to bind the HTML form fields to Java objects and validate the input data, respectively. As for the view itself, Spring supports a variety of view technologies including JSP, Velocity, and JasperReports. Let's look at how we might use JSP for our views.

Figure 5 shows a sample forms screen that can be developed in JSP using Spring's bind tag library. The Spring bind tag library is simple yet powerful. It's typically used in JSP files via the <spring:bind> tag that essentially binds HTML form fields to the command object. Furthermore, it provides access to special variables in JSP that can be accessed using JavaServer Pages Standard Tag Library (JSTL) expressions such as ${status.value}. The code excerpt below demonstrates how the spring:bind tag library works – notice how we bind directly to the Department domain (business) object that we looked at in the Hibernate section:

```
<spring:bind path="command.departmentCode">
 <input
   name='<c:out value="${status.expression}"/>'
   type="text" size="10"
   maxlength="30">
</spring:bind>
```

Besides spring:bind, Spring 2.0 introduces some new tag libraries that ease working with individual HTML form elements. Some of these include form:input, form:textarea, and so on.

### Spring MVC Configuration Concepts

Till now we've only looked at Java-related concepts for Spring MVC. Of course Spring also has configuration aspects. For starters, its DispatcherServlet class has to be configured in the Web server's Web.xml file, so files matching a certain extension (like .htm) can be processed by Spring MVC. Once this is configured, we're in the world of Spring MVC. From here, we configure view resolvers and handler mappings in a Spring application context file. View resolvers map incoming URLs to actual view names. Handler mappings map incoming URLs to controller classes.

### *Spring ORM*

One of the beautiful things about Spring is its support for third-party APIs such as JDBC, JAX-RPC, Hibernate, and many others." For example, if we use Spring with Hibernate, we can eliminate the code required to manage Hibernate's sessionFactory, session and programmatic transaction management. The benefits of using Spring with Hibernate is that it cuts down the Hibernate-related code by almost a half and provides additional benefits such as easier testing, consistent exception hierarchy, and management of Hibernate resources.

Visit springframework.org for more details.

## Effectively Developing Java Code with Eclipse

In my book I have a chapter dedicated to the Eclipse SDK. Initially I planned to use a generic title but later I changed it to "The Eclipse Phenomenon!" because that's the best way to describe what's happening in the Eclipse community. No matter how good another Java IDE might be, the sheer number of plug-ins available for Eclipse is hard to match. If you do a search for the words for "eclipse plugins" on the Web, you'll literally get millions of matches. In other words, the Eclipse community is exploding!

The Eclipse platform is essentially a framework that provides a set of services that other plug-ins can build on. Each plug-in is developed to the same platform, which translates into a set of highly integrated tools. The Eclipse Web site currently has many sub-projects underway including everything from support for various programming languages to modeling plug-ins to reporting, testing, and performance to almost everything else required for software development.

The core concepts of Eclipse include a workspace, essentially a directory for your projects. The first main screen in Eclipse is known as the workbench (see Figure 9). The workbench contains a set of editors and views organized as perspectives. Perspectives are task-specific layouts of editors and views.

One of the core Eclipse plug-ins is the Java Development Tool (JDT). It's an extremely robust plug-in with support for Java development such as managing Java-related files (.java, .class, and .jar), Java views, compilation, code formatting, debugging, refactoring, and syntax highlighting – in fact, the JDT plug-in is a full-blown product in itself.

Another important plug-in is the Eclipse Web Tools Platform (WTP), intended for developing JEE Web applications. It provides editors like JSP, HTML, CSS, JavaScript, and WSDL. It also provides extremely handy database query and model tools to explore the database, run queries, and analyze the data. Of course, the ability to create and test Web Services easily is another major feature of this plug-in. JDJ published a series of articles by Boris Minkin on using WTP (see http://java.sys-con.com/author/minkin.htm).

Apart from the plug-ins provided on the Eclipse.org Web site, there's no shortage of plug-ins available for Eclipse on the Web. Sites such as eclipseplugincentral.com, eclipse-plugins.2y.net, and myeclipseide.com have a large number of plug-ins.

Other Eclipse features include team support via tight integration with CVS, a robust help system, a large number of preferences, and shortcut keys.

In short, Eclipse provides tools to work on all tiers of an applications, that is, data, Web, and business.

## Beyond the Basics

If we lived in a perfect world, we would simply gather user requirements, code them, and deliver perfectly stable applications that would run smoothly without intervention. However, as developers, we know it doesn't quite work that way and that there are times to troubleshoot problems or monitor the "health" of our applications; so, let's review some techniques that can help.

### *Debugging*

Debugging is typically a process of locating and fixing a defect, although it can also be used to step through code to ensure the logic works right. Eclipse's JDT plug-in provides a powerful Java debugger that lets us debug local Java programs or ones running on a remote Java server. Like most debuggers, the Eclipse JDT debugger can step through code (one line at a time or by jumping to a breakpoint) and inspect variables. It also provides a very useful feature known as Hotswap that lets us change code on-the-fly, recompile, and continue debugging in the same session. This is a handy feature since setting up a debugging session just the way you want it can take time. Figure 10 demonstrates how we can debug our Java code, see the data in the database, and see the console output – all in a highly integrated fashion using two completely different plug-ins JDT and WTP.

*Profiling*

Java profilers have been around for almost as long as Java. Among other things, they let us analyze the heap for memory usage and leaks, CPU utilization, trace objects, and methods, and determine performance bottlenecks. A variety of Open Source profilers are available out there, as well as commercial ones (like YourKit Java Profiler and Quest's JProbe Suite). Some run as standalone Java programs; others can be deployed to a servlet container; and still others are available as Eclipse plug-ins. So, if you're looking for an Open Source profiler,: www.manageability.org/blog/stuff/open-source-profilers-for-java/view/ lists a dozen of them.

One other profiler that's supposedly one of the best is the NetBeans Profiler, however, I haven't tried it out yet but the screenshots look sleek. Visit profiler.netbeans.org to learn more about this.

*Logging*

Logging is an important aspect of software development and varies from print statements to complex database-based logging. Logging types can include audit logging, tracing, and error reporting.

Two logging frameworks commonly found in the Java world are Apache Log4J (logging.apache.org/log4j/) and JDK logging (java.sun.com). Another option is to use Apache's Jakarta Commons Logging (jakarta.apache.org), which provides a thin bridge between various logging frameworks, including Log4J and JDK logging. While we can use simple print statements to output messages from your programs, logging frameworks let us control the output of our messages according to destination (files, database, remote), levels (fatal, error, warning), and format (date and time). In addition, logging frameworks provide benefits such as automatically rolling over log files when they reach a certain length.

*Monitoring*

Java Platform Standard Edition (JSE) 5.0 provides built-in remote monitoring, management, and the JConsole Swing-based utility (see Figure 11) to monitor applications that run using JSE 5.0 or later versions. These tools can be used to view the resource utilization of Java applications. For example, it can help detect memory issues, class loading, and garbage collection, control JDK logging levels, and manage an application's Managed Beans (MBeans). Furthermore, Spring's JMX support lets us automatically register POJOs, which gives us a powerful paradigm because we could easily write business-type objects that can be monitored (instead of the typical low-level technical stuff). For example in our sample application, this could include the number of timesheet records fetched and the number of logins.

## Conclusion

We've covered a lot of ground in this article. As I mentioned at the beginning, this is a road map for one way of doing agile Java development. However, what would an article in a Java magazine be without some Java code? So I have a completely functional sample timesheet application, downloadable (and a deployable war file) at http://visualpatterns.com/resources.jsp. The resource section below also provides a summary of links specified throughout the article.

I hope this article has provided some guidelines for developing Java in an agile manner. Cheers!

## Resources

- *Agile Data:* agiledata.org
- *Manifesto for Software Development:* agilemanifesto.org
- *Agile Modeling:* agilemodeling.com
- *Scrum:* controlchaos.com
- *Eclipse Foundation:* eclipse.org
- *Extreme Programming:* extremeprogramming.org
- *Hibernate:* hibernate.org
- *Martin Fowler:* martinfowler.com
- *The Spring Framework:* springframework.org
- *Test Driven Development:* testdriven.com
- *Author's Web site:* visualpatterns.com

**Anil Hemrajani** is the author of the book Agile Java Development with Spring, Hibernate and Eclipse. He has 20 years of experience in IT working with Fortune 100 companies and smaller organizations. He is the founder of Isavix Corporation , a successful IT service company, and DeveloperHub.com, formerly isavix.net, an award-winning online developer community. He has published numerous articles in well-known trade journals and gotten several awards. Anil can be reached via his Web site, VisualPatterns.com.

"If we lived in a perfect world, we would simply gather user requirements, code them, **and deliver perfectly stable applications that would run smoothly without intervention"**

# IT'S THE END OF THE LINE FOR SPAM.

## PUT IT IN THE CAN.

**Ads, fraud schemes, porn, viruses, phishing attempts… if it's offending e-mail, Roaring Penguin Software's CanIt™ solutions will stop it before it reaches your mail server.**

- Simple administration
- Automatic spam and virus updates
- Compatible with all e-mail servers
- Available as software or hardware appliance
- Flexible end-user controls

**Throw the book at spam today! Request a free 20-day CanIt evaluation and stop spam on YOUR terms!**

ROARING PENGUIN SOFTWARE INC.

www.roaringpenguin.com +1 (613) 231-6599 sales@roaringpenguin.com

# Web Services Security in Java EE

*The present and future*

Andrei Iltchenko

In my earlier article "Moving to SOA in J2EE 1.4" published in the February issue of *JDJ* I introduced you to the new object distribution model based on Web Services that became available to Enterprise Java applications with the advent of Java EE 1.4. In this article I want to look at the security features available in Java EE SOA.

Here you'll get thehands-on knowledge of Web Services security in Java EE that we acquired when adding security support to OptimalJ-generated SOA applications. It's based on the J2EE 1.4 specification itself as well as on what is actually supported and it works in three major J2EE 1.4 application servers — JBoss 4.0.4, WebSphere 6.0.2.x, and WebLogic 9.1. You'll also learn about the new mandatory security features available to Web Service endpoints in Java EE 5.0.

## Overview of Security in Java EE

Java EE comes with a mature security model that provides for the guaranteed features that have to be supported by all compliant application servers: authentication, authorization, confidentiality, and integrity. Though not yet required by the specification, most high-end application servers also support some sort of auditing of security-related events and non-repudiation — in other words a way of preventing an invocation sender from denying responsibility for the action — for communicating with Web Service components.

Authorization is based on logical security roles that are simple names defined by the component provider or application assembler in XML deployment descriptors. The code underneath all Java EE components — JSPs, servlets, and Enterprise JavaBeans — can be restricted declaratively based on logical security roles. In the case of EJBs, access can be limited on an Enterprise Bean's method level, whereas access to JSPs and servlets is enforced based on their URL and the HTTP method utilized (e.g. POST, GET, etc.). Besides declarative authorization, programmatic authorization is also supported so that a component's code can dynamically inquire whether the security context of the current user is associated with a particular logical security role and make a decision based on this analysis. How a given principal is actually mapped to a set of security roles depends on the Java EE notion of a security domain and the principal authentication mechanisms associated with the domain.

The confidentiality and integrity requirements are met at the transport layer with the help of the Secure Sockets Layer (SSL 3.0) protocol and the related IETF standard Transport Layer Security (TLS 1.0) protocol. For SSL and TLS only X.509 certificates are supported for authenticating principals. Kerberos-based authentication mechanisms in TLS are presently regarded as optional and aren't implemented by the application servers this article concentrates on.

The authentication security requirement is by far the most difficult to explain since it requires understanding the Java EE notion of a security domain, which is essentially a security mechanism used to authenticate the user. Here are the three arbitrary examples of security domains:

1. A security domain where users are authenticated based on their X509 certificates presented during an SSL handshake. In this case the protocol used by the client for communicating with the application server can be HTTPS, IIOP/SSL, or JRMP/SSL.
2. A security domain that uses the SRP protocol in communicating a user's name and password to the server in a secure fashion. Here the communications protocol that the client uses can be JRMP.
3. A security domain that uses the HTTP Basic Authentication in communicating a user name and password to the server. Such a security domain will use either HTTP or HTTPS as the supported communications protocol.

Different security domains entail different types of principals for representing users. In the first security domain presented above, a principal will be derived from an X509 certificate or a certificate chain that the user presented during an SSL handshake. In the second example, a principal will be taken from the user name specified by the client. Here's a code sample taken from JBoss that shows how a certificate chain can be mapped to a principal:

```
public Principal  toPrinicipal(
        X509Certificate[] certs) {
   Principal   subject =
           certs[0].getSubjectDN();
   return  subject;
}
```

Thus a security domain deals with a set of principals of a particular kind (e.g., based on X509 certificates, Kerberos tickets, plain user names, etc.). This set is termed a principal realm. For each principal realm, there's mapping between its principals and the one or more logical security roles

**Andrei Iltchenko** is a development lead at Compuware Corporation where he works on the MDA product OptimalJ and is responsible for the business logic area of OptimalJ-generated J2EE applications. He is also a Sun certified Java developer for Java Web Services, a Sun Certified Business Component Developer, a Sun Certified Developer, and a Sun Certified Programmer.

that are used in Java EE applications. Application servers offer a plethora of ways to represent a principal realm, the most common of which are a local OS user registry, an LDAP server, an RDBMS schema, a Kerberos KDC, or a simple .properties files.

Modern Java EE application servers support different security domains or let users define their own based on the JAAS login modules available. See the sidebar "What is JAAS?" for more information on using JAAS in Java EE.

When a Java EE application is deployed, the deployer assigns the application modules to the security domains that have been configured in the targeted application server installation. Typically, the components of a Java EE module (an EJB .jar module or a Web .war module) are all assigned to the same security domain; some application servers let the components of a given module be assigned to different security domains, but this practice is generally avoided since it can easily lead to confusion. Java EE doesn't standardize the scope of a security domain and leaves it up to vendors. At the moment all high-end application servers let a security domain span multiple application server installations (which typically form a cluster).

## Security Context Propagation and Single Sign-on

A Java EE application server features three different containers (there's also an applet container that is typically embodied by a Web browser program): a Web Container that hosts JSPs and servlet components, an EJB Container where EJB components are deployed, and an Application client container (see the sidebar "Application client containers" for more details on this concept). EJB and Web Containers are typically collocated, and components running in the Web Container can access EJBs of the corresponding EJB container. Figure 1 depicts the relationships between the three containers and various ways in which a client can access a Java EE application. For simplicity's sake I depicted all the enterprise components as running in the same application server on a single node, but it doesn't have to be this way; modern application servers let them be distributed among multiple nodes.

The following are the two typical usage scenarios shown in Figure 1 involving access to an enterprise Java application:

1. A user accesses a JSP or a servlet component deployed in a Web Container with a Web browser. He authenticates himself to the Web Container using either 1) a username and password that his Web browser prompts him to enter (Basic HTTP Authentication) or 2) an X509 certificate that the browser lets the user choose from a pre-installed set of user certificates. The servlet component carries out the presentation-related activities and invokes an EJB Session component (using a local invocation in the same JVM or RMI-based protocol) to carry out the business logic-related tasks. To fulfill the business logic task the session bean can invoke an Entity EJB, call on an EIS with a help of a JCA resource adapter, or carry out some JDBC-based data access. After completing its work, the session component returns the processing results to the servlet component, which in turn renders them to the user in HTML.
   The user can then invoke the servlet or some Web component or JSP again.The application server maintains a session with the user's browser and doesn't require re-authentication.
2. A Java client application uses either RMI-IIOP or RMI-JRPM to access the server. The application prompts the user for a name and credentials and authenticates itself to the server with the help of JAAS and one or more JAAS the login modules provided by the vendor. For RMI-IIOP, the CSIv2 SAS protocol will most likely be used to communicate authentication data to the server. The client application accesses an EJB deployed in an EJB Container. Like the first scenario, the invoked EJB can call other EJBs or enterprise services.
   The client application then goes on to invoke another EJB without having to re-authenticate the user. Listing 1 is an example of such a client application for WebSphere.

A lot can be gathered from these scenarios and from Figure 1.

First, they show that external clients can access components running in the

### What Is JAAS?

JAAS stands for Java Authentication and Authorization Service. It provides a Java implementation of the Pluggable Authentication Module (PAM) framework that was pioneered in the Solaris operating system.

Modern application servers use JAAS to authenticate principals accessing resources running in the server. It is also used heavily by Java clients running in an application container as a way of authenticating themselves to the application server and benefiting from single sign-on.

The article "JAAS in the Enterprise" gives a pretty good idea of the future direction that this specification is likely to take in upcoming releases of Java Enterprise Edition.

### Application Client Containers

Application client containers are a way of giving remote J2SE clients access to the components and services of a Java EE application server.

Despite its rather imposing name, an application client container can be nothing more than a set of .jar libraries that let a standalone Java application access the JNDI tree of an Java EE application server, whereby gaining access to the Enterprise Beans and other enterprise services such as JMS, container-managed JDBC data sources, and JavaMail.

For JBoss AS the set of .jar libraries is all that's required to set up a client container on a host where a Java SE runtime is installed (see http://wiki.jboss.org/wiki/Wiki.jsp?page=J2EEClient for more information).

For WebSphere and WebLogic, the setup is more involved — both require that a client host have access to the AS installation and provide an application client launch program that must be used to execute a client program.

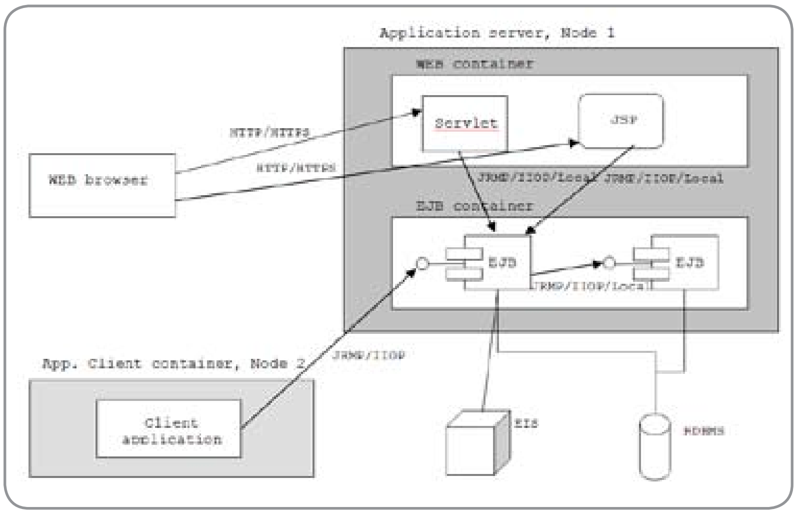


Figure 1

WEB container by using either HTTP or HTTPS and components hosted in the EJB container with RMI-IIOP or RMI-JRMP. They also show that components can use 1) local invocations in the same JVM, 2) RMI-IIOP, or 3) RMI-JRMP for inter-component communication. Which of the three is used depends on the vendor and the configuration of the application server.

Second, in both examples the clients authenticated themselves to the container before being able to use a component, and the application server propagated the established client security context when the component invoked the other EJBs.

Third, the samples demonstrate Java EE support for single sign-on (frequently abbreviated as SSO), thanks to which needless re-authentications are avoided for subsequent application are avoided server access. The propagation of the client security context and single sign-on are two important security characteristics of Java EE.

Application servers let the client security context be propagated if local JVM invocations, RMI-IIOP, or RMI-JRMP are used as inter-component communication transports and the component targeted belongs to the same security domain. A client security context typically consists of a principal object (whose type depends on the security domain of the Java EE application) and zero or more associated credentials presented during authentication. Java EE specifies RMI-IIOP and the accompanying CSIv2 OMG spec as the only interoperable way of propagating a client security context that must be understood and supported by all compliant application servers (a security context propagated with RMI-JRMP is only meaningful if the targeted component runs in an application server from the same vendor). Using CORBA-related standards for interoperability among disparate application servers reflects the CORBA-oriented nature of the early Java EE specifications that holds to this day.

The way single sign-on capabilities are gained depends on the client. For Web browser clients, the Web Container uses either HTTP cookies or URL rewriting to track a session. If the browser accesses the container through HTTPS then SSL Sessions can also be used. Which of the three mechanisms is available depends on the application server and its configuration. Some servers such as WebSphere support all three, others don't.

With a Java application client, user authentication credentials are established during the JAAS login and are then kept in a thread local variable of the Java application thread that executes the code in Listing 1. The credentials will then be used for each subsequent application server access by the thread until the logout statement has been executed.

Besides the default mode in which an established client security context is propagated during inter-component communication, Java EE lets a given enterprise component specify another identity (a so-called run-as identity) that will be in effect when the component accesses other enterprise resources. The run-as identity mode is typically used only when there's no client security context (e.g., with message-driven beans and ejbTimeout callback methods of Enterprise Beans that implement the TimedObject interface). For instance, in OptimalJ-generated J2EE applications, users are given a warning whenever they model a message-driven bean or an enterprise component that uses the timer service but doesn't specify a run-as identity.

## Security in J2EE 1.4 for Web Service Endpoints

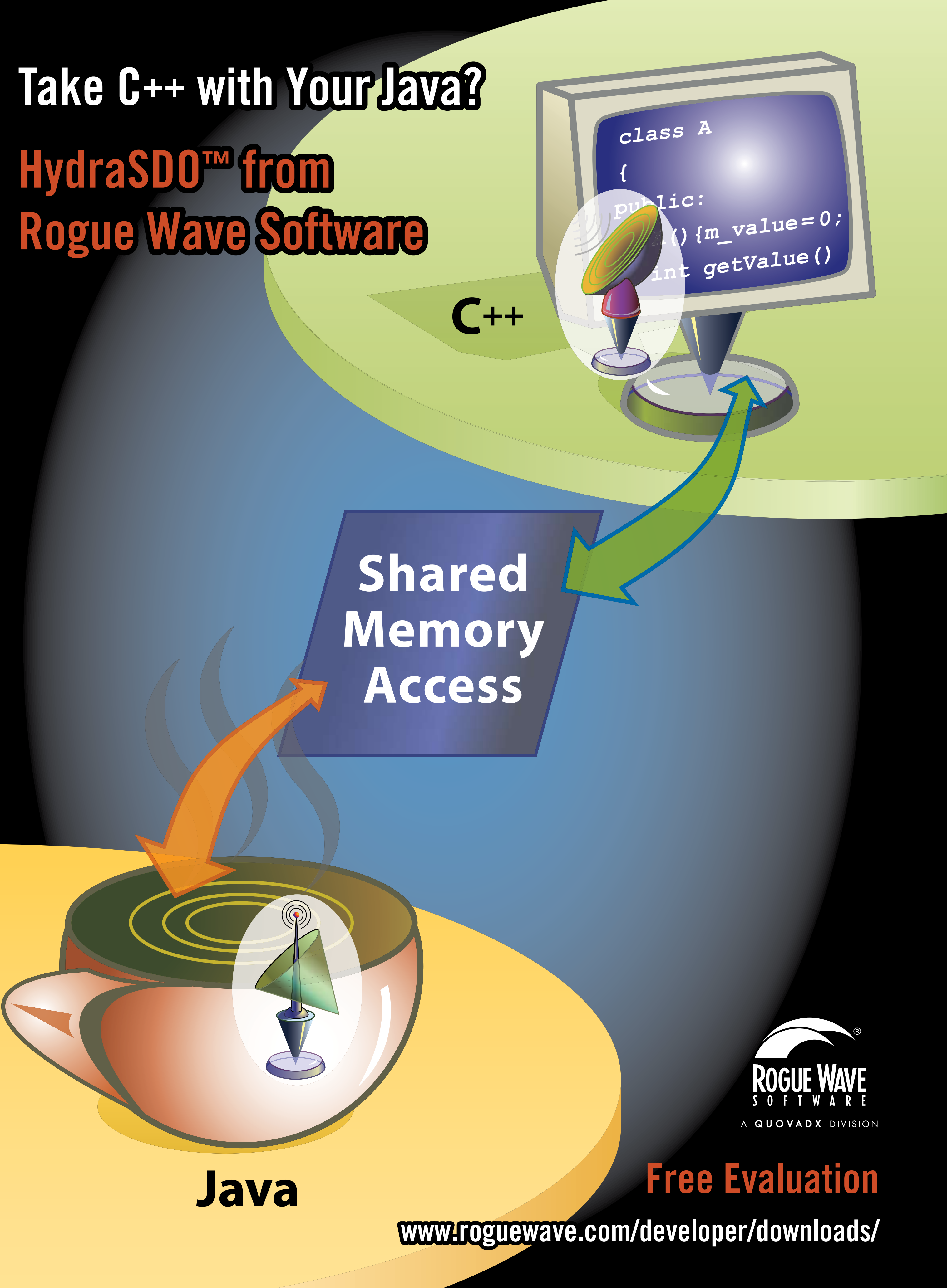
By far, the most visible change in J2EE 1.4 is the introduction of Web Service endpoints, which effectively provides a viable alternative model for component distribution and interoperability that can compete with CORBA. The Web Service endpoint is a term used to describe Web Service components deployed in a J2EE container. As I explained in my earlier article, a service endpoint can be implemented using a stateless session bean, in which case it runs in the EJB container, or as a Java class that's registered as a servlet, in which case it runs in the WEB container and is called a JAX-RPC endpoint. Associated with each service endpoint is its service endpoint interface (SEI). As prescribed by the WS-I Basic Profile, J2EE limits SOAP to HTTP and HTTPS as its only interoperable underlying transport protocols.

Figure 2 shows the component landscape in J2EE 1.4. It's analogous in intent to Figure 1 and complements it by emphasizing the entries to the application server through Web Service endpoints. There are a number of important differences between Figure 1and Figure 2. For instance:

1. A Web Service component running in the WEB container can be accessed from a client program running in the application client container, which wasn't possible in J2EE 1.3;
2. A Web Service component hosted in

# Take C++ with Your Java?

## HydraSDO™ from Rogue Wave Software



ROGUE WAVE SOFTWARE®
A QUOVADX DIVISION

**Free Evaluation**

www.roguewave.com/developer/downloads/

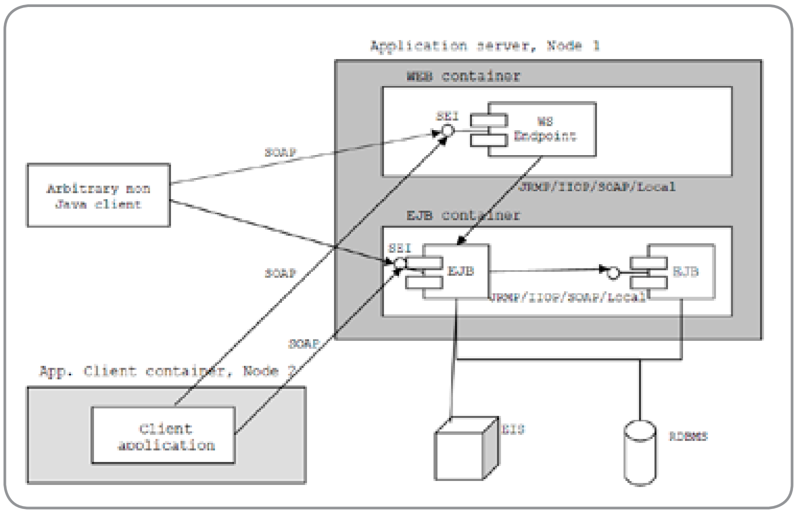


**Figure 2**

the EJB container can be reached by an external non-Java client not only via the CORBA IIOP protocol, but also using the lighter-weight SOAP protocol;

3. J2EE components can now use four different transports for inter-component communications: 1) local invocations in the same JVM, 2) RMI-IIOP, 3) RMI-JRMP, or 4) SOAP.

Let's look at the ramifications of these changes and the mandatory Web Services security-related features supported by all complaint application servers.

The ability to reach an application server with the new SOAP protocol impacts authentication and J2EE requires that the following two authentication mechanisms be supported:

1. HTTP Basic Authentication and
2. HTTPS Mutual Authentication, which uses the certificate presented by a Web Service endpoint client during a SSL/TLS handshake.

It's fairly obvious that HTTP Basic Authentication provides no security unless combined with HTTPS. It's also apparent that with these two choices authentication occurs in the transport, which has consequences for inter-component communication that I'll discuss later.

HTTPS Mutual Authentication is a very viable authentication scheme. Unfortunately we learned that many applications servers, for example, JBoss and WebSphere (see http://jira.jboss.com/jira/browse/JBAS-3019), don't allow one to check the client certificates presented during SSL/TLS handshakes against the CRLs (Certificate Revocation Lists), which severely limits the security of this authentication method in large public enterprise applications.

To achieve integrity and confidentiality when communicating with Web Service endpoints, J2EE fully supports SSL/TLS, giving it the same level of security with regard to integrity and confidentiality as whencommunicating via RMI-IIOP or RMI-JRMP.

Using SOAP for inter-component communication in J2EE might present some surprises since no client security context propagation is supported on most application servers (JBoss, WebSphere) (http://jira.jboss.com/jira/browse/JBWS-679 explains the situation in JBoss).

You might also run into security breaches similar to http://jira.jboss.com/jira/browse/JBWS-675 — a security vulnerability I discovered in JBoss 4.0.3 and older versions that enables you to create a J2EE component (an EJB or a servlet) that would send the credentials of a user accessing it to a non-authorized party whenever the component being accessed communicates via SOAP with another one.

For J2EE applications generated by our OptimalJ product, we decided to simply prohibit our users from modeling inter-component communications via the Web Service endpoints, and I strongly recommend that you don't use Web Service endpoints for this purpose either. If you need to use SOAP for inter-component invocations, you'll have to explicitly configure security parameters for each { invokingComp/invokingModule, invokedCompWithServiceEndpoint } pair, called a Web Service Reference, in a vendor-specific deployment descriptor.

The situation with single sign-on isn't much better. Listing 2 shows a piece of Java EE application client code that accesses a Web Service endpoint. The code does the same thing as Listing 1, but I use SOAP instead of IIOP or JRMP to reach the application server.

You can see that you don't benefit from JAAS for authentication and SSO, because you have to specify the authentication data for each invocation. And uncommenting JAAS login statements would be futile — even though it will create a login session with the server — because no attempt will be made to put the established credentials in the SOAP messages generated by the calls to the enterprise components. Fortunately most application servers let you remove authentication-specific data from your code and put it into a deployment descriptor, but that still falls short of proper SSO support.

Example 2 shows you a piece of a JBoss client deployment descriptor with the necessary authentication data. The advantage of the deployment descriptor approach is that it enables you to specify security parameters for each { invokingComp/invokingModule, invokedCompWithServiceEndpoint } pair only once and does not needlessly clutter the accompanying Java code for each invocation.

**Example 2**

```
<service-ref>
  <service-ref-name>
service/Bean1Service
  </service-ref-name>
  <port-component-ref>
    <service-endpoint-interface>
Bean1ServiceEndpoint
    </service-endpoint-interface>
    <call-property>
      <prop-name>
```

```
javax.xml.rpc.security.auth.username
      </prop-name>
      <prop-value>Name</prop-value>
    </call-property>
    <call-property>
      <prop-name>
javax.xml.rpc.security.auth.password
      </prop-name>
      <prop-value>Passw</prop-value>
    </call-property>
  </port-component-ref>
</service-ref>
```

## Security for Web Service Endpoints — The Future

If you're familiar with recent developments in Web Services security standards, you may wonder why SSO and security context propagation are still a problem. The answer is simple. When Java EE 1.4 was finalized in November of 2003 there was no approved WS specification that addressed the problem of security context propagation and SSO at the SOAP message level. The OASIS Web Services Security (WS-Security) standard was then in draft, which precluded its incorporation in the spec. Since then WSS has become an OASIS standard and has moved from version 1.0 to 1.1. Currently, the standard features two approved profiles, each of which allows you to achieve SSO: SAML Token Profile and Kerberos Token Profile. There are also two new non-OASIS specifications that potentially address this problem: WS-Trust and WS-SecureConversation. Both are currently in a public draft state and their support in existing Java EE application servers is largely absent.

It's logical then that Java 2 Platform, Enterprise Edition, v5.0, which has just been blessed by the Java Community Process executive committee, should offer some support for WS-Security to address the problem of security context propagation. Surprisingly Java EE 5.0 contains almost no changes in Web Services security and doesn't address the problems I delineated in the previous section.

Fortunately IBM, BEA, and JBoss support the WS-Security standard in their WebSphere 6.0.x, WebLogic 9.1, and JBoss 4.0.4. WebLogic even uses it for security context propagation and SSO while WebSphere and JBoss limit themselves to authentication, encryption, and digital signing.

WebSphere, WebLogic, and JBoss implement version 1.0 of the WS-Security standard and support the following specifications:

- SOAP Message Security
- UsernameToken Profile
- X.509 Certificate Token Profile

WebLogic also implements the SAML Token Profile so it can offer Web Services SSO and a security context propagation experience.

Beside the SSO qualities in WebLogic, the servers achieve the following additional security characteristics with the help of WS-Security:

1. Authentication at the SOAP message level using plaintext name passwords and X.509 Version 3 certificates;
2. On WebLogic, authentication at the SOAP message level using SAML tokens;
3. On WebSphere, authentication at the SOAP message level using LTPA tokens;
4. Encryption of SOAP messages or parts thereof using symmetric cryptography. Secret keys can be encrypted and put in messages too. Essentially, the application servers support most of the required and some of the optional encryption-related algorithms specified in the XML Encryption Syntax and Processing specification that the WS-Security standard builds on.
5. Digital signing and verifying SOAP messages or portions thereof. This is an important item because it achieves non-repudiation — something that's not possible with RMI-IIOP- and RMI-JRMP-based transports and wasn't possible in the Java EE security model in the past. The application servers implement most of the required portions of the XML-Signature Syntax and Processing specification, which also underlies the WS-Security standard.
6. Means of defeating replay attacks by using nonces and timestamps in SOAP headers.

As I mentioned earlier, Java EE v 1.4/5.0 is silent on the subject of the WS-Security standard. One unpleasant consequence of that silence is that you need to use container-specific deployment descriptors to specify all the WS-Security related information in your application, which obviously limits the portability of your application. WebLogic is an interesting exception. BEA chose to adopt WS-Policy as a standard means of specifying WS-Security-related configuration of its Web Service endpoints (Microsoft has done the same thing in its .NET Framework.) And JBoss is moving toward embracing WS-Policy (see http://jira.jboss.com/jira/browse/JBWS-856 for more information).

The lack of WS-Security support in Java EE 1.4/5.0 means that Sun Microsystem's technology conformance kits for Java EE (which an application server must pass for it to be declared compliant) exclude any related testing and so vendors can deviate from one another in their WS-Security implemetations.

As is the case with most crucial Web services specifications, the WS-I Consortium (the producer of the Basic Profile – a specification that ensures interoperability among Web service components today) – is defining a subset of the WS-Security standard and its constituent specifications that all the vendors will have to support in the same way. This effort is known as the Basic Security Profile and is now in draft. Until it's completed and all the vendors have incorporated it in their products, interoperability issues are inevitable (largely because of the extensiveness the WS-Security standard and the plethora of decisions that a vendor has to make when implementing it). (The following article will give you an idea of what problems you might run into if you use products from different vendors: http://www-128.ibm.com/developerworks/webservices/library/ws-was-net/index.html?ca=drs-%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20Articles.)

## Conclusion

Clearly the new Web Services object distributed model in Java Enterprise Edition 1.4/5.0 could supplant RMI-IIOP and RMI-JRMP in Enterprise Java as the object distribution protocol that offers the same or better security services. At the moment, however, the level of support that the Java Enterprise Edition v. 1.4/5.0 specification required is clearly insufficient for that to happen over night. Still, given the current industry trends and with more and more vendors committing themselves to WS-Security and its Basic Security Profile counterpart, it is more a question of when than if.

The fact that support for WS-I Basic Security Profile isn't prevalent yet in Java EE applications servers and that the profile is still subject to change has consequences for those who develop portable Java EE applications. If the portability of your application is a concern, you'd be better off limiting yourself to the guaranteed Java EE 1.4/5.0 Web Services security features and avoiding WS-Security for a time being. This is the approach we took in our forthcoming OptimalJ product since we needed to shield our users from the specifics of any particular vendor implementations.

## References

- Moving to SOA in J2EE 1.4: http://java.sys-con.com/read/180362.htm
- Java 2 Platform Enterprise Edition Specification, v1.4: http://java.sun.com/j2ee/j2ee-1_4-fr-spec.pdf
- Java 2 Platform, Enterprise Edition, v5.0: http://jcp.org/aboutJava/communityprocess/pr/jsr244/
- The SSL Protocol, Version 3.0: http://home.netscape.com/eng/ssl3/draft302.txt
- RFC 2246: The TLS Protocol, Version 1.0: http://www.ietf.org/rfc/rfc2246.txt
- RFC 2459: Internet X.509 Public Key Infrastructure, Certificate and CRL Profile: http://www.ietf.org/rfc/rfc2459.txt
- RFC 2712: The Addition of Kerberos Cipher Suites to Transport Layer Security (TLS): http://www.ietf.org/rfc/rfc2712.txt
- RFC 2945: The SRP Authentication and Key Exchange System: http://www.ietf.org/rfc/rfc2945.txt
- Making Login Services Independent of Authentication Technologies: http://java.sun.com/security/jaas/doc/pam.html
- JAAS in the Enterprise: http://jdj.sys-con.com/read/171477.htm
- Common Secure Interoperability, Version 2: http://www.omg.org/technology/documents/formal/omg_security.htm#CSIv2
- Certificate and Certificate Revocation List (CRL) Profile: http://www.ietf.org/rfc/rfc3280.txt
- OASIS Web Services Security (WSS): http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss
- Web Services Trust Language (WS-Trust): ftp://www6.software.ibm.com/software/developer/library/ws-trust.pdf
- Web Services Secure Conversation Language (WS-SecureConversation): ftp://www6.software.ibm.com/software/developer/library/ws-secureconversation.pdf
- XML Encryption Syntax and Processing: http://www.w3.org/TR/xmlenc-core/
- XML-Signature Syntax and Processing: http://www.w3.org/TR/xmldsig-core/
- Web Services Policy Framework (WSPolicy): http://specs.xmlsoap.org/ws/2004/09/policy/ws-policy.pdf
- Basic Security Profile: http://www.ws-i.org/deliverables/workinggroup.aspx?wg=basicsecurity

**Listing 1**

```
final static InitialContext   iniCtx = new InitialContext();
// Will prompt the user for a name and credentials
// using a GUI dialog box.
CallbackHandler   handler = new WSGUICallbackHandlerImpl();
LoginContext       logCtx = new LoginContext("WSLogin", handler);

// Single signon allows to access the container without
// needing to reauthenticate.
logCtx.login();
Subject   subject = logCtx.getSubject();

PrivilegedAction   bean1Action = new PrivilegedAction() {
public Object run() {
    try {
        Object   homeProxy = iniCtx.lookup("ejb/bean1");
        Bean1Home   bean1Home = (Bean1Home)
            PortableRemoteObject.narrow(homeProxy, Bean1Home.
class);
        Bean1Remote   bean1 = helloHome.create();
        bean1.businessMethod1(...);
        ...
    }
    catch (CreateException ce)  { ... }
    catch (RemoteException re)  { ... }
}
};
PrivilegedAction   bean2Action = new PrivilegedAction() { ... }

// Access components in the application server on behalf of 'sub-
ject'
com.ibm.websphere.security.auth.WSSubject.doAs(subject, bean1Ac-
tion);
com.ibm.websphere.security.auth.WSSubject.doAs(subject, bean2Ac-
tion);

// End the logon session.
logCtx.logout();
```

**Listing 2**

```
InitialContext   iniCtx = new InitialContext();
// No use logging in, authentication will occur in the transport
// anyway and no effort to propagate credentials established in
// the lines commented out below will be made!
//
// UsernamePasswordHandler   handler =
//          new UsernamePasswordHandler(username, password);
// LoginContext   logCtx =
//          new LoginContext("client-login-module-name", handler);
// logCtx.login();

Service   srv = (javax.xml.rpc.Service)
    iniCtx.lookup("java:comp/env/service/Bean1Service");
Bean1ServiceEndpoint   bean1Stub = (Bean1ServiceEndpoint)
    srv.getPort(Bean1ServiceEndpoint.class);

// Can use either 1) Java EE standard stub properties to specify
// authentication data for HTTP Basic Authentication, or
// 2) Vendor specific extensions to specify a cerificate
// for HTTPS Mutual Authentication

Stub   stub = (Stub) port;

// Portable code for HTTP Basic Authentication.
stub._setProperty("javax.xml.rpc.security.auth.username", "Name");
stub._setProperty("javax.xml.rpc.security.auth.password",
"Password");

// JBoss specific code for HTTPS Mutual Authentication.
// stub._setProperty("org.jboss.webservice.keyStore", keyStore);
// stub._setProperty("org.jboss.webservice.keyStorePassword",
//                    "keyStorePassword");
// stub._setProperty("org.jboss.webservice.keyStoreType", "JKS");
// stub._setProperty("org.jboss.webservice.trustStore", trust-
Store);
// stub._setProperty("org.jboss.webservice.trustStorePassword",
//                    "trustStorePassword");
// stub._setProperty("org.jboss.webservice.trustStoreType", "JKS");

bean1Stub.businessMethod1(...);
...
// End the logon session.
// logCtx.logout();
```

# REALWORLD FLEX SEMINAR

www.flexseminar.com

## Learn How to Build the Next Generation of Web Apps **from the Experts!**

**August 14, 2006**
The Roosevelt Hotel
New York City

Adobe Flex 2 is a complete, powerful application development solution for creating and delivering cross-platform rich Internet applications (RIAs), within the enterprise and across the web. Not until now has there been a way for enterprise programmers and architects to work with existing tools of choice, familiar programming models and integration with existing systems and infrastructure. Multi-step processes, client-side processing, direct manipulation and data visualization are all key factors in the Flex solution.

The "Real-World Flex" One-Day Seminar will delve deep into the central workings of Flex so that the seminar delegates can integrate this timely new technology into their applications, creating powerful interactive content. Delegates will learn from the experts who not only created the product but also those who are using it to the massive benefit of their employers, their customers and the Web.

## "Go Beyond AJAX with Flex."

***The list of topics at the Real-World Flex Seminar includes:***

- ✓ ***The Flex Approach to RIA Development***
- ✓ ***Bridging Flex and AJAX***
- ✓ ***Integrating The SPRY Framework***
- ✓ ***Using Flex Builder 2***
- ✓ ***Flex for Java Developers***
- ✓ ***ActionScript 3.0 Tips and Tricks***
- ✓ ***How To Use ColdFusion with Flex***
- ✓ ***Leveraging Flash***
- ✓ ***Preparing for Adobe Apollo***
- ✓ ***MXML Master Class***

***Who Should Attend?***

- ✓ ***CEOs and CTOs***
- ✓ ***Senior Architects***
- ✓ ***Project Managers***
- ✓ ***Web programmers***
- ✓ ***Web designers***
- ✓ ***Technology Evangelists***
- ✓ ***User Interface Architects***
- ✓ ***Consultants***
- ✓ ***Anyone looking to stay in front of the latest Web technology!***

**Early Bird:**
(Hurry for Early Bird Pricing) ........ $395*

**Conference Price:**
(Register Onsite) ........ $495*

OFFER SUBJECT TO CHANGE WITHOUT NOTICE, PLEASE SEE WEBSITE FOR UP-TO-DATE PRICING

* Golden Pass access includes Breakfast, Lunch and Coffee Breaks, Conference T-Shirt, Collectible Lap-Top Bag and Complete On-Demand Archives of sessions in 7 DVDs!

Sponsored by Adobe   Web Developer's & Designer's Journal

Produced by SYS-CON EVENTS The World's Leading i-Technology Event Producer www.events.sys-con.com

# By Invitation **Only!**

*Effective page authorization in JavaServer Faces*

**Frank Nimphius** **Duncan Mills**

Application security — the art of applications defending themselves — represents an important line of defence in an overall in-depth security strategy. Web applications that follow the Model-View-Controller (MVC) architecture can, and should, have security implemented on all three layers. Normally it's the controller component that handles page authorization in MVC, the view layer that hides controls and information based on user authorization, and the model that enforces the business rules and input validation. However, it's up to the developer, based on an individual security policy and the programming technology used, to decide where to put security. Using pluggable validator components in JavaServer Faces (JSF), for example, developers may decide to verify user input on the view layer as well as on the model layer.

JavaServer Faces, the new J2EE standard for building feature-rich Web applications with JEE 1.5, has few integrated security features. JSF generously delegates the task of implementing application security such as page authorization to the application developer, leaving many developers pondering where to start, where best to put security, and which security technology to choose.

This article aims to answer such questions for authorization in JavaServer Faces, demonstrating how a custom PhaseListener that uses J2EE container-managed security can be used to implement access control for JSF pages. Besides page authorization, the security PhaseListener supports protocol switching between HTTP and HTTPS, a common requirement of applications that work with sensitive data on a Web page.

## The J2EE Security Choices

The J2EE platform provides two built-in security technologies for the application developer to use: the Java Authentication and Authorization Service (JAAS) and container-managed security, also known as J2EE security.

The Java Authentication and Authorization Service is a J2SE 1.4 security standard designed for the Java desktop that's also used as an implementation technology for security in J2EE. The JAAS authentication infrastructure is built as a Java version of the Pluggable Authentication Module (PAM) architecture that allows one or more authentication providers to be used for user identification. Before JAAS, the Java 2 security platform was code-centric, determining access privileges based solely on the location of the Java sources. Using JAAS, Java security now also looks at the authenticated user when evaluating access control to resources. JAAS's benefit is its ability to implement fine-grained access control through external Java permission classes, which associate users with a list of resources and allowed actions.

Authentication and authorization in J2EE security is configured declaratively in the application's web.xml deployment descriptor and handled by the J2EE container at runtime. Working with APIs defined in the J2EE servlet standard, application developers don't have to worry about the implementation of security in a container. In container-managed security, authorization is enforced on URL patterns, which are absolute or relative URLs. This however also means that authorization is only enforced on requests that are initiated by the client, not as server-side forward requests.

Ease of use, the clean separation of security definition and application code, and portability across application servers are the main reasons for the wide adoption of container-managed security among business application developers. J2EE security is sufficient to implement many common security use cases. As a reflection of its popularity and its portability, container-managed security is used in the code examples of this article to illustrate effective page authorization in JavaServer Faces.

## Container-Managed Security in J2EE

In container-managed security, a user is granted access to protected URL resources through security roles defined in the web.xml deployment descriptor. Security roles in J2EE are logical names used in Web applications that are mapped during or after deployment to user groups that exist on the target application server platform.

**Listing 1 Web.xml excerpt granting the app_user security role access to the protected URL resource /faces/protected/***

```
<security-constraint>
   <Web-resource-collection>
   <Web-resource-name>Members</Web-
resource-name>
   <url-pattern>/faces/protected/*</url-
pattern>
   </Web-resource-collection>
   <auth-constraint>
      <role-name>app_user</role-name>
   </auth-constraint>
</security-constraint>
...
<security-role>
   <role-name>app_user</role-name>
</security-role>
```

To access a protected application resource, Web application users must first authenticate, which in container-managed security is handled by the J2EE container. Either the application developer or the application deployer configures the type of authentication in the web.xml deployment descriptor.

**Listing 2 Basic authentication defined for the jazn.com realm**

```
<login-config>
    <auth-method>BASIC</auth-method>
    <realm-name>jazn.com</realm-name>
</login-config>
```

**Frank Nimphius** is a principal product manager for application development tools at Oracle Corporation. As a conference speaker, Frank represents the Oracle J2EE development team at J2EE conferences world wide, including various Oracle user groups and the Oracle Open World conference.

**Duncan Mills** is a Java evangelist and product manager at Oracle Corporation. He's been working in the IT industry for the last 17 years in various development and DBA roles and now works on the team responsible for the JDeveloper IDE. You can follow Duncan's life on the GroundBlog at http://www.groundside.com/blog.

To check authorization programmatically in an J2EE application – like in JavaServer Faces – application developers use the isUserInRole method in the servlet API. This isUserInRole method is also exposed via a convenience method in JavaServer Faces through the static FacesContext class. Role names referenced in application code ought to be mapped to roles defined in the web.xml file using the <security-role-ref> element if the role names don't match. Using the <security-role-ref> element, developers don't have to be aware of the security role names that exist in the web.xml descriptor when developing an application.

**Listing 3 Mapping the "user" role name used in the application code to the security role name "manager_role" defined in the web.xml file**

```
<security-role-ref>
  <role-name>user</role-name>
  <role-link>app_user</role-link>
</security-role-ref>
```

If the authenticated user isn't authorized to access the requested URL resource, the J2EE container responds with HTTP error 403, indicating a bad request. A HTTP error 401 is returned if a user cancels the authentication process. HTTP error codes and Java exceptions are handled declaratively in the web.xml file using the <error-page> element.

**Listing 4 Redirecting a request in response to unauthorized page access handling error code 403 and 401**

```
<error-page>
   <error-code>403</error-code>
   <location>Error.jsp</location>
</error-page>
<error-page>
   <error-code>401</error-code>
   <location>Logon_cancelled.jsp</location>
</error-page>
```

If SSL is required to ensure secure communication when accessing a specific Web resource, the <security-constraint> element added for a protected resource contains an additional <user-data-constraint> element. Setting the transport guarantee to "confidential" indicates that SSL is required.

**Listing 5 Indicating that a Web resource requires SSL**

```
<user-data-constraint>
    <transport-guarantee>CONFIDENTIAL</
transport-guarantee>
</user-data-constraint>
```

Though J2EE Web resources can be configured to require HTTPS this way, the J2EE specification doesn't demand that Web containers respond by automatically switching protocols. Instead, J2EE containers usually return an HTTP error message to indicate a failed user request. Individual J2EE containers like Apache Tomcat provide native support for switching between the HTTP and HTTPS protocol. However, you ought to be aware that such solutions aren't portable to other J2EE containers.

## Page Navigation in JavaServer Faces

So we've seen how basic container-managed security is configured through the use of logical application roles protecting a URL pattern. Now let's look at how JSF manages page navigation and how the principles of container-managed security can be applied.

Page navigation in JavaServer Faces is defined in the WEB-INF\faces-config.xml file, where the JSF NavigationHandler component uses it.

The application developer configures navigation in JSF either as a server-side forward or, if the <redirect/> element is included in the navigation case, as a browser redirect.

**Listing 6 JSF navigation case issuing a redirect request for page navigation**

```
<navigation-case>
   <from-outcome>success</from-outcome>
   <to-view-id>/Departments.jsp</to-view-
id>
   <redirect/>
</navigation-case>
```

Server-side forwards are the de facto default implementation of navigation cases in JSF, but such navigation harbors two side effects: first, the URL won't change, and so pages in the application aren't bookmarkable; and secondly, there's no new submission of a URL pattern for container-managed security to be applied to. As a consequence, if you've implemented container-managed security, you'll have to use <redirect/> explicitly on any navigation cases that have to trigger a security check across role boundaries.

It should be clear by now that authorizing pages and implementing a secure channel for communication isn't easy to achieve in JavaServer Faces. And using JAAS instead of container-managed security offers no better solution. The answer lies in a combined approach: a reusable custom security implementation specifically designed to work with JSF based on JAAS or container-managed J2EE security.

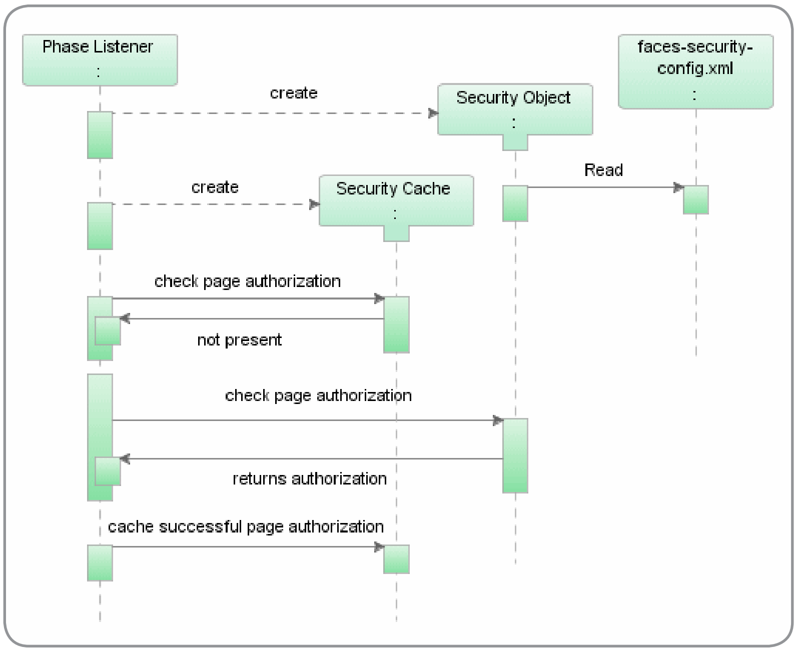


**Figure 1** Security PhaseListener sequence

## Where Does Security Belong in JavaServer Faces?

As we've mentioned, JSF has no real security infrastructure built into it and this leaves the developer to wonder where and how to implement security. The examples that address user authentication in JSF (using JAAS, for instance) don't cover the important task of page authorization.

The authorization enforcer security pattern demands that authorization be handled in a central location. For many Web applications, this central location is a ServletFilter associated with the application front controller. In JSF, though, security is best implemented using the built-in extension points provided by the JavaServer Faces architecture.

The two candidate approaches for implementing security in JSF are:

- A custom ViewHandler that decorates the default ViewHandler and also adds security checks to the createView and restoreView methods
- A PhaseListener that adds security evaluation to the restoreView and invokeAction phases

Adding security to a Viewhandler doesn't appear to be an ideal solution because there's no guarantee that a security ViewHandler will be executed before any other custom ViewHandlers that have been registered. If another ViewHandler is present and writing to the servlet response stream then security-related actions such as redirecting a request to the HTTPS port would result in illegal state exceptions. So it seems that a custom PhaseListener is the better approach to adopt.

## Developing a JavaServer Faces PhaseListener for Security

JavaServer Faces executes in a rich request lifecycle composed of a sequence of individual phase events: Restore View, Apply Request Values, Process Validation, Update Model Values, Invoke Application, and Render Response. PhaseListeners in JavaServer Faces are Java classes configured in the faces-config.xml file and notified when a specific phase event of interest occurs.

To execute custom logic before and after a specific event, custom application code is added to the beforePhase and afterPhase methods of a PhaseListener. Both methods accept an input argument of a PhaseEvent to provide information about the calling phase, for example, a phase ID. The third method developers have to implement is getPhaseId. The getPhaseId method is used to declare which phases the listener is actually interested in being notified about.

**Listing 7 Custom PhaseListener listening to any event and implementing the JSf PhaseListener interface**

```
public class J2EESecurityPhaseListener
implements PhaseListener
{
  public SecurityPhaseListener() { }
  public void afterPhase(PhaseEvent phas-
eEvent) { }
  public void beforePhase(PhaseEvent phas-
eEvent){ }
  public PhaseId getPhaseId()  {return
PhaseId.ANY_PHASE;}
}
```

An application can have more than one PhaseListener configured. PhaseListeners are configured in the <lifecycle> element of the faces-config.xml configuration file.

**Listing 8 PhaseListener configuration in faces-config.xml**

```
<lifecycle>
    <phase-listener>
         com.groundside.jsf.J2EESecurityP
haseListener
   </phase-listener>
</lifecycle>
```

Java IDEs like Oracle JDeveloper provide visual editors to simplify this configuration of the faces-config.xml file.

## A J2EE security PhaseListener — J2EESecurityPhaseListener

The J2EESecurityPhaseListener is a custom PhaseListener that implements page authorization for the JSF restoreView and invokeAction phases using container-managed J2EE security roles. JSF doesn't let PhaseListener register for multiple selected events. Either a PhaseListener registers for one specific event or it registers for all events. So the J2EESecurityPhaseListener listener is registered to listen to any phase, but responds only to the RESTORE_VIEW and INVOKE_APPLICATION phases.

**Listing 9 The J2EESecurityPhaseListener listens to any event but responds to RestoreView and InvokeApplication only**

```
public void afterPhase(PhaseEvent phas-
eEvent)
  {
    PhaseId phaseid = phaseEvent.getPha-
seId();

    if ( phaseid == PhaseId.RESTORE_VIEW||
         phaseid == PhaseId.INVOKE_
APPLICATION )
    { … }
}
```

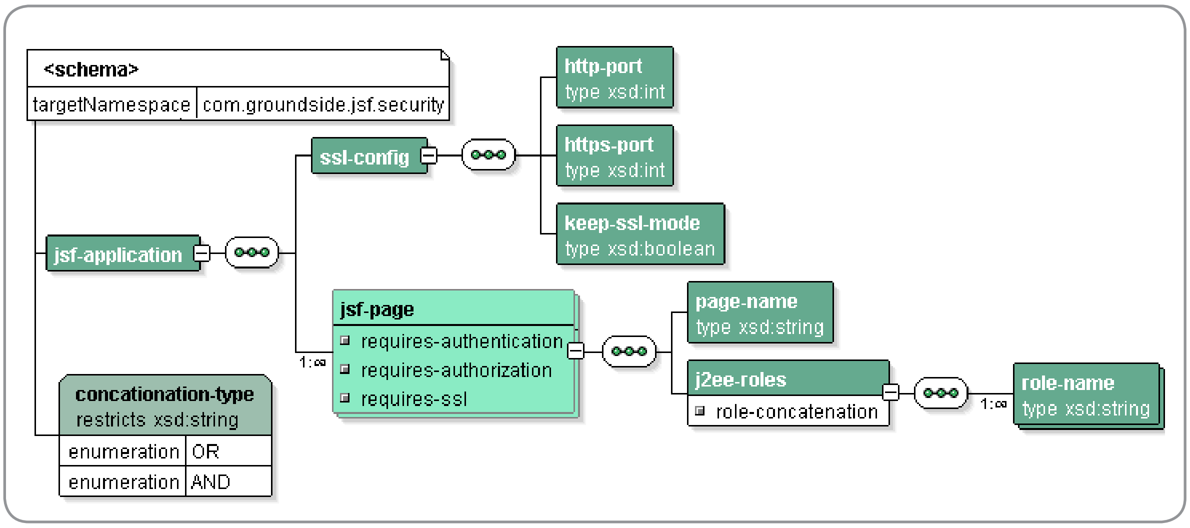


**Figure 2** XML schema of the faces-security-config.xml file

# #1 Rated AJAX and Rich Internet Application toolkit* TIBCO General Interface

**Build 100% Pure Browser Rich Internet Apps with TIBCO General Interface™**

“TIBCO General Interface is a friendly, capable toolkit for building sophisticated JavaScript Web applications that run in a browser.”

Download today at **http://www.tibco.com/mk/gi**

TIBCO®
The Power of Now®

The J2EESecurityPhaseListener uses an extra XML configuration file, faces-security-config.xml, to define JSF pages that require authentication, authorization, or secure communication with SSL. The faces-security-config.xml file is located in the WEB-INF directory like the JavaServer Faces faces-config.xml file.

The security PhaseListener parses the faces-security-config.xml file with the Apache Digester [DIGESTER] to create a Java security object that is subsequently cached in the application scope. The security object contains security information about the configured HTTP and HTTPS ports, whether or not to keep SSL after it's used the first time, and individual authentication, authorization, and SSL requirements for each page.

The PhaseListener determines page authorization only once per session for each JSF page (viewId) that a user requests. At successful authorization, a reference to the JSF page is cached in the session to improve performance on subsequent requests for the same resource.

A user who requests a protected page without first having been authenticated is redirected to an authentication servlet. The authentication servlet is configured in the web.xml deployment descriptor and protected by a J2EE security role that all users are members of. All JSF pages configured in the faces-security-comfig.xml file to require authorization implicitly also require authentication. If an unauthenticated user tries to access a page that requires SSL, a container-managed logon form is launched over HTTPS.

The faces-security-config.xml file consists of two parts:

- A global configuration section to provide HTTP and HTTPS port information and whether SSL communication should be kept once established. The keep-SSL-mode information overrides the individual page configuration for SSL.
- Multiple jsf-page elements to configure page and directory authorization. The page is identified by its viewId, which starts with a leading slash followed by the relative URI, not excluding the faces virtual mapping (for example, /protected/main.jsp). Directory names are indicated by an appended wildcard character '*' (for example, /protected/*).

For each protected page or directory, the developer defines the authentication, authorization, and SSL requirements. If a page requires SSL, but the current request protocol is HTTP, the J2EESecurityPhaseListener redirects the request to a configured HTTPS port. Similarly, if the protocol is currently HTTPS but the page doesn't require a secure channel and the keep-ssl-mode is set to false, the PhaseListener redirects the request to the HTTP port.

JSF pages and directories that require authorization have to reference the name of one or more J2EE security roles defined in the web.xml file. The role-concatenation attribute lets developers specify whether a user has to be a member of all the configured roles (AND) or only a single role (OR).

Because the security configurations are stored in an XML file, the page authentication strategy and the authorization definitions can be changed at any time without recompiling or redeploying the application.

To use the J2EESecurityPhaseListener in custom JSF applications, developers do the following

- Deploy the jsfj2ee-security-util.jar file as a library with the application.
- Create and configure the faces-security-config.xml file in the application WEB-INF directory.
- For protocol switching, configure the HTTP and HTTPS ports and ensure that the application server is set up to share a session between the two ports.
- Configure the authentication servlet in the web.xml file.
- Configure container-managed authentication and J2EE security roles in the web.xml file.

Once set up, the J2EESecurityPhaseListener lets developers apply authorization to any JSF page navigation – a forward or a redirect – while still using standard container mechanisms to handle authentication and role definition.

To improve default container-managed security, the J2EESecurityPhaseListener also lets developers configure page authorizations that use multiple J2EE roles, rather than having to define multiple roles to achieve fine-grained authorization in conventional container-managed security.

## Completing the Page Authorization Picture with a Custom Property Resolver

The "limited view" security design pattern defines what users can view and access and lets them access accordingly. Protecting a page from unauthorized access alone isn't enough. UI components that initiate page navigation should be hidden if the user is not allowed access the navigation target. To set a component's rendered property to false, expression language (EL) can be used. The EL can be sourced from handcrafted managed beans defined as part of the application or more generically through a security-aware custom variable resolver. A sample resolver of this type is available for download at jsf-security.sourceforge.net.

## Summary

Without a doubt, JavaServer Faces is a great step forward for J2EE application development. Application security is, however, an important component in the development lifecycle and, unfortunately, this is where the current JavaServer Faces specification falls short. In the future, we would hope and expect that security integration, perhaps of a nature similar to that discussed in this article and the associated code, will be added to the specification.

The custom J2EESecurityPhaseListener developed for this article uses container-managed security for the sake of simplicity, but it can be adapted to use either JAAS or a custom security provider. The source code for this PhaseListener based solution can be downloaded with this article.

## References

- E. Burns, C. Schalk. *JavaServer Faces: The Complete Reference.* ISBN 0072262400
- D. Alur, J. Crupi, and D. Malks. *Core J2EE Patterns: Best Practices and Design Strategies.* 2d ed. ISBN 0-13-142246-4
- http://jsf-security.sourceforge.net/
- DIGESTER: http://jakarta.apache.org/commons/digester/

SAN FRANCISCO'06

Register by July 14
to save up to $500
Enter priority code D0120

# 15 Years of BREAKING the rules...
## Now we're Finally **MAKING** the Rules!

**Join LinuxWorld** as we celebrate the 15th Anniversary of Linux—and its amazing evolution from a kernel to industrial-strength corporate applications. Come to San Francisco—to the main event for the Linux community—and dive into the open technologies that are still transforming IT all these years later.

**Go Deep** in technical training
**Brainstorm** with the best
**Check Out** the latest products and services
**Feed Your Mind** in visionary keynotes
**Feel the Energy** of the entire Linux and open source community

**Don't Miss the Biggest Linux Event Ever!**
Hundreds of products, services, and training sessions all under one roof!

## KEYNOTE SPEAKERS

| | | | | |
|---|---|---|---|---|
| **Lawrence Lessig** | **Guru Vasudeva** | **Peter Levine** | **Greg Besio** | **Richard Wirt** |
| *Professor of Law, Stanford Law School* | *AVP & Enterprise Chief Architect, Nationwide* | *CEO, XenSource* | *Corporate VP, Mobile Devices Software, Motorola* | *VP, Senior Fellow, General Manager, Software and Solutions Group, Intel* |

## Go Deep in 100+ Sessions on Key Linux and Open Source Topics

- Iron-Clad Security for Open Environments
- Managing Mixed Environments
- Virtualization
- Scalable Open Source Applications
- Desktop Linux
- Grid Computing
- Mobile Linux
- Web Services and SOA
- VoIP
- IT / Line-of-Business Alignment
- High-Performance Open Source
- The Business of Open Source
- Kernel and System Development

## Check out the latest products and services from hundreds of key exhibitors

| | | |
|---|---|---|
| AMD | HP | Oracle |
| CA | IBM | Unisys |
| Dell | Intel | VMware |
| EMC | Novell | and more! |

**Discover the Full Power of Today's Linux**
**Get the Skills, Solutions, and Insight Your Business Needs**

**Feed Your Mind** in Visionary Keynotes by Lawrence Lessig and other Linux and open source luminaries

**Feel the Energy** of a full slate of 15th Anniversary Events

**Register by July 14 and save**
Enter priority code D0120 and save up to $500
www.linuxworldexpo.com/register

## Brainstorm with Top Open Source Experts

- **Alan Boda,** Software Consultant, HP
- **Fabrizio Capobianco,** CEO, Funambol, Inc.
- **Dave Dargo,** Senior VP for Strategy and CTO, Ingres
- **Chris DiBona,** Open Source Programs Manager, Google
- **Scott Handy,** VP of Worldwide Linux & Open Source, IBM
- **Greg Kroah-Hartman,** Linux Kernel Subsystem Maintainer, SuSE Labs / Novell
- **Dan Kusnetzky,** Executive VP of Marketing, Open-Xchange Inc.
- **Eben Moglen,** Chairman, Software Freedom Law Center
- **Bernard Traversat,** Director of Advanced Development, Sun Microsystems

Get complete details at www.LinuxWorldExpo.com
and sign up for the event where Linux history is made.
Register by July 14 with priority code D0120 and save up to $500.

LINUXWORLD® CONFERENCE & EXPO

Conference: August 14 – 17, 2006
Expo: August 15 – 17, 2006

Moscone Center, San Francisco

Plus expanded content with OpenSolutions WORLD™

hp invent PLATINUM SPONSOR · IBM PLATINUM SPONSOR · MOTOROLA PLATINUM SPONSOR · Novell PLATINUM SPONSOR · ORACLE PLATINUM SPONSOR · palmsource PLATINUM SPONSOR · coverity SILVER SPONSOR · SAP SILVER SPONSOR · WYSE SILVER SPONSOR · OpenSource MEDIA SPONSOR

LinuxWorld is open to business professionals only. No one under 18 years of age will be admitted. D0120

IDG WORLD EXPO

# Configuring the WebLogic-Eclipse Plug-in

*Designed to run the WebLogic Server from the Eclipse IDE*

by Deepak Vohra and Ajay Vohra

The WebLogic-Eclipse plug-in is designed to run the WebLogic Server from the Eclipse IDE. With the WebLogic-Eclipse plug-in, the WebLogic Server gets started and stopped from Eclipse. An application deployed in the WebLogic Server can be debugged from Eclipse with the plug-in. By installing the WebLogic plug-in in Eclipse the WebLogic Server can be configured and administered from the Eclipse IDE by setting the server classpath and JVM options in Eclipse.

## Overview

A J2EE developer is commonly required to administer the WebLogic Server and debug applications deployed in the WebLogic Server. While the WebLogic Server administration console can start and stop the WebLogic Server, the administration console doesn't provide for setting the JVM options and the server classpath. The JVM options and server classpath have to be set in the startWebLogic script. And to debug an application deployed in the WebLogic Server, an IDE with a remote debugger is needed. With the WebLogic plug-in the WebLogic Server can be administered from the Eclipse IDE. In this tutorial we'll develop a J2EE application consisting of a Session EJB and a servlet, deploy the application in the WebLogic Server from the Eclipse IDE, and debug the application in Eclipse.

## Preliminary Setup

- ***Download and install the Eclipse 3.0 IDE:*** www.eclipse.org
- ***Download and install the WebLogic 8.1 Server:*** www.bea.com/framework.jsp?CNT=index.htm&FP=/content/products/weblogic/server

## Installing the WebLogic-Eclipse Plug-in

Now we'll install the WebLogic-Eclipse IDE. In the Eclipse IDE select Help>Software Updates>Find and Install. The Install/Update frame gets displayed. Select Search for new features to install and click on the Next button. The Install frame gets displayed. Click on the New Remote Site button to specify an update Web site from which to install a plug-in. In the New Update Site frame specify a name and the URL from which the WebLogic-Eclipse plug-in is installed. The URL for the WebLogic-Eclipse plug-in is https://eclipse-plug-in.projects.dev2dev.bea.com/update. An update site configuration gets added. Select the checkbox for EclipseWebLogic for "Sites to include in search" and click on the Next button. In the features to install frame select the WebLogic-Eclipse Feature and click on the Next button.

Select the license terms and click on the Next button. In the Install location frame the directory in which the WebLogic-Eclipse plug-in will be installed is specified. Click the Finish button to complete the configuration of the WebLogic plug-in. The JAR Verification frame is displayed. Click the Install button to install the WebLogic-Eclipse plug-in. Restart the Eclipse workbench for the plug-in to get installed. The WebLogic-Eclipse plug-in gets installed in the Eclipse IDE. The Run>Start WebLogic and Run>Stop WebLogic features get added to Eclipse.

## Configuring the WebLogic-Eclipse Plug-in

After installing the WebLogic-Eclipse plug-in we'll configure the plug-in in the Eclipse IDE. First, create a project with which the WebLogic plug-in is to be configured. Select File>New>Project. In the New Project frame select Java>Java Project and click the Next button. In the Create a Java project frame specify a project name and click the Next button. In the Java Settings frame add a source folder for the project. Click the Add Folder button. In the New Source Folder frame specify a folder name. A message frame prompts to set the bin folder as the build output folder. Next, add the libraries required for the project. The example application requires the J2EE JAR in the classpath. Select the Libraries tab and click on the Add External JARs button.

Add the J2EE 1.4 j2ee.jar file to the project. The j2ee.jar gets listed in the project Libraries. Click the Finish button to complete the configuration of the project. A project gets added to the Eclipse IDE Package Explorer view.

Next, we'll specify a WebLogic Server configuration. Select Window>Preferences. The Preferences frame gets displayed. Select the WebLogic node. In the WebLogic preference page select the version of the WebLogic Server to be configured. Specify the different field values, which are listed in Table 1. The values may vary depending on the directory in which the server is installed and in which the domain is configured. Click on the Apply button to apply the specified values.

If any JAR files have to be added to the server classpath, select the WebLogic>Classpath node. JAR/Zip files or directories can be added before the WebLogic libraries or after the WebLogic libraries. Select the WebLogic>JavaVM Options node to specify JavaVM options. For example, modify the weblogic.ProductionModeEnabled property. Set the property

**Deepak Vohra** is a Sun Certified Java 1.4 programmer and Web developer.
dvohra09@yahoo.com

**Ajay Vohra** is a senior software engineer with Compuware.
ajay_vohra@yahoo.com

value to false to start the server in development mode. Click on the Apply button to apply the JavaVM options.

Next, specify the projects to be debugged with the WebLogic Server configuration. Click on the Add button. Select the projects to be added to the plug-in configuration. To debug a project, the project has to be in the plug-in configuration. Click the OK button.

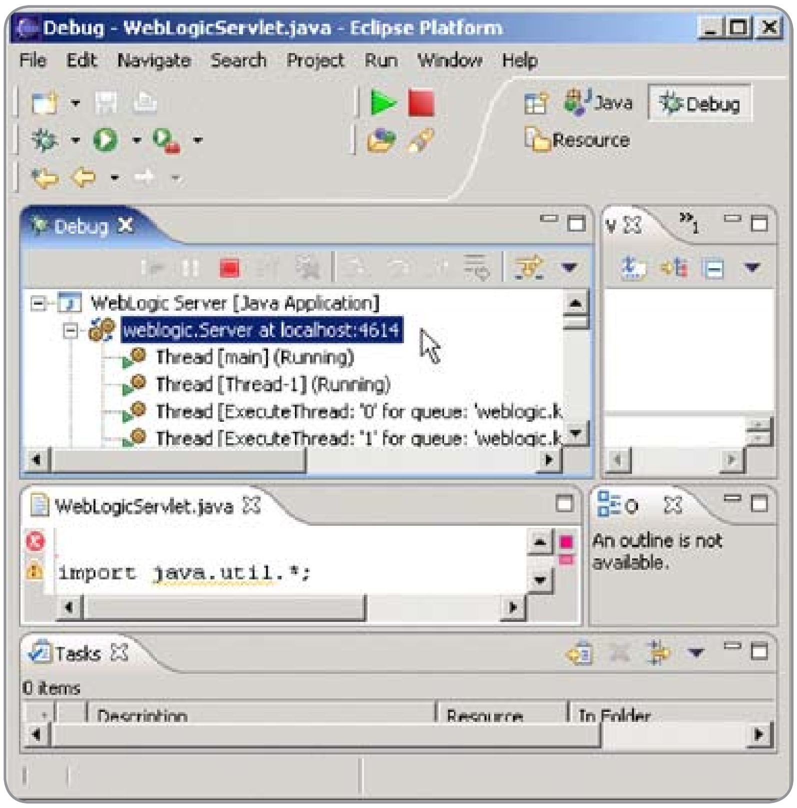
The selected projects get added to projects list. Click on the Apply button and then the OK button. The WebLogic plug-in gets configured with a project and the WebLogic Server.

## Developing and Debugging a WebLogic Application

After configuring the WebLogic plug-in, develop a J2EE application to deploy and debug in the WebLogic Server. The example J2EE application consists of a Session EJB and a client servlet. The J2EE application is available in the resources zip file (source code for this article can be found by viewing the article online in the WLDJ archives, http://wldj.sys-con.com/read/issue/archives/, Vol. 5, iss. 2). Extract the resources zip file to a directory. In the Eclipse project EclipseWebLogic, which was configured in the previous section, import the src directory of the J2EE application with File>Import. In the Import frame select the File System node and click the Next button. In the File system frame select the directories/files to add to the project and click the Finish button.

The example J2EE application files get added to the project. Build the project with the Ant build.xml file. Right-click on the build.xml file and select Run>Ant Build. The J2EE application gets built and deployed in the WebLogic Server applications directory. Next, start the WebLogic Server in the Eclipse IDE with Run>Start WebLogic. The Session EJB/Servlet application gets deployed in the WebLogic Server as listed in the applications node.

Run the WebLogicServlet in the browser with the URL http://localhost:7001/weblogic/webLogicPlugin. The output from the servlet is displayed in the browser. Next add an exception (a NullPointerException) to the client servlet to demonstrate the debugging feature of the WebLogic plug-in. In the WebLogicServlet servlet replace


Debug - WebLogicServlet.java - Eclipse Platform
File Edit Navigate Search Project Run Window Help
Java
Debug
Resource
Debug
WebLogicServlet.java
Tasks

```
WebLogic Server [Java Application]
    weblogic.Server at localhost:4614
        Thread [main] (Running)
        Thread [Thread-1] (Running)
        Thread [ExecuteThread: '0' for queue: 'weblogic.k
        Thread [ExecuteThread: '1' for queue: 'weblogic.k
```

```
import java.util.*;
```

An outline is not available.

0 items

| | Description | Resource | In Folder |
|---|---|---|---|

Figure 1

```
out.println(sessionEJB.getEclipsePlug-
in());
```

with

```
String str=null;
out.println(str.toString());
```

Add a breakpoint to the servlet with Run>Add Java Exception Breakpoint. In the Add Java Exception Breakpoint frame select the NullPointerException. Delete the previous build directory and build the application with the build.xml. Select the Debug perspective. In the Debug perspective the WebLogic Server is shown to be running at localhost host.

Run the example servlet (with the NullPointerException) in the browser. Because the servlet has an exception the server gets suspended and the Debug perspective displays the NullPointerException. The application can be debugged with the debug features in the Run menu item.

## Conclusion

Thus the WebLogic Server gets administered from the EclipseIDE with the WebLogic plug-in and applications deployed in the server are debugged from the Eclipse IDE. A limitation of the WebLogic plug-in is that debugging JSPs isn't supported. The 2.0 version of the plug-in will have additional features.

| Field | Description | Value |
|---|---|---|
| BEA Home | The BEA installation directory | C:/BEA |
| WebLogic Home | The WebLogic Server installation directory | C:\BEA\weblogic81 |
| Domain Name | The WebLogic domain | mydomain |
| Domain Directory | The WebLogic domain directory | C:\BEA\user_projects\domains \mydomain |
| Server Name | The WebLogic Server name | myserver |
| User | User to login to the WebLogic Server | weblogic |
| Password | Password to login to WebLogic Server | weblogic |
| Hostname | WebLogic Server hostname | localhost |
| Port | WebLogic Server port | 7001 |

**Table 1** WebLogic Eclipse plug-in

# Java Techie to Manager

**Benjamin Garbers**

*You've got the job now what do you do?*

You're six-feet, 190 pounds and can type System.out.println faster than most people can say AJAX. You're a person who dreams about the Milwaukee Brewers winning the World Series and the correct data structure to be used when talking about a baseball player. You've spent five years of your life writing Java code and leading Java development teams. You consider yourself an expert in Swing, Struts, XML, and XSL-FO and feel comfortable talking about any other buzzword in the Java world such as JSF, Portal, and AJAX. You've had experience as development lead on a team with anywhere from three to seven people where Java applications were rolled into production well within the scheduled deadline. Now you have received a management position on an internal Java development team. Where do you start? What things do you look at from day one? What's your role going to be as a manager? What would you like to see happen within your team? Do you want to keep your technical skills? How do you rate your employees at the end of the year?

These are just some of the question's that you'll have to answer.

Fortunately, I'm the Brewers fan who just got a new first-line management position. The team that I'm managing consists of 18 employees with skillsets ranging from Java Swing development to J2EE Web development. The main point of our existence is to create, support, fix and build tools inside IBM for a number of platforms. A number of small tools have already been developed that use Swing technology for the front-end. The small tools end up communicating with DB2 systems on the back-end and start a number of native back-end processes depending on the back-end servers' platform. The team has also created a Web application that lets internal developers create a fix pack of a particular product. These are examples of just a couple of the many Java tools that my department is responsible for.

Now back to the questions at hand. Where does a manager start when taking over a Java development team? These are just a couple of the things that concerned me when coming in as manager of a Java development team.

## Who's Doing What?

Every manager has to understand what the main responsibility of the team is. Once that's understood then the next question to answer is, who is working on achieving that goal. What positions have been defined in the department to carry out the team's primary responsibility? For instance, do you have developers working on a single application from the beginning to end or do you have each software development process task broken down among different employees. Once you understand the tasks that everyone is working on, does it matter how they're done? For example, the team that I'm managing has application owners who are responsible for the entire development process lifecycle for a particular application. An application owner would have to gather the new requirements that come in, create a design that fits into the existing application design, develop, unit test, and do the production test. And if an external customer discovers a problem with the tool it's their responsibility to fix it.

Some things I've heard from the group is that testing all our small tools is quite expensive. Every small tool is dependent on each other. New functionality added to one of them may have an impact on another, thus causing all application owners to test their code before it's released.

From a resource perspective this really scares me. You wouldn't like your most experienced developers spending a lot of time on testing. Some would disagree with me on this and say that this person has the application domain experience and should be involved in production testing. However, I feel that testing something like this should be documented in a test plan and tested by a separate group. Test cases could be written by this separate group cross-referencing the requirements. That way a different set of eyes could manually test the application outside of the application owners who should only do unit testing.

## Is There a Development Process?

As the manager of any software department I would hope so. Hardcore software developers hate processes. I know this from past experience. When I was given an assignment, I wanted to complete it as fast as I could by writing code. If you wanted to know my progress all you had to do was ask. I felt the information in my head was sufficient. However, this kind of thinking makes things very hard when working on a team that's larger than one person. Information has to be communicated from one person to another. The memory of what someone said lasts only so long. Having documentation helps remind an employee of what's required. It helps for reviews and lets an employee hand his work off if something happens and he's pulled from the project.

Without a development process

**Ben Garbers** is currently a 1st line manager at IBM where the department he manages creates and maintains Java standalone applications and dynamic Java web applications run on Websphere. Before his management position he was the lead developer on a number of teams that developed standalone Java applications.
garbersb@us.ibm.com

it's even harder to rate employee performance. Who is your best designer? Who is your best coder? By defining a development process, the strengths and weaknesses of each employee can be measured at particular stages of the development process. Running a tool suite that does metrics throughout a development process can be used to measure performance. Tracking and monitoring this kind of information will also help you understand the task force needed for a particular project. For instance, if a manager knows how long it took for an application to be finished with a particular number of employees, it makes it easier to estimate how long it will take those employees on the next project.

The team that I've inherited has an ad hoc development process. There's no standardized format of what's required in each development phase. For instance, Team A could have a requirements document that looks different from Team B's requirements document. Does something like this need to be standardized throughout the development process? Some would argue that as long as there's documentation for each development stage it shouldn't matter. They'd also argue that the format of each document should be up to the project lead. However, if you have employees switching from one team to another, this may become an issue. It may take an employee some time to understand a format that's different from what they used in a prior project. From a management perspective it's always nice to standardize the format in a tool that can run some kind of metrics. For example, if a requirements document is submitted with a tool, metrics could be run on how good the document actually is. When a review is held for the requirements document, the number of problems found in the requirements document could be traced and analyzed by a manager. This could be a perfect way to isolate the employees who have strong requirements-gathering skills. As a manager, I feel it's a priority to make sure our development team has a standardized format for all development process milestones.

## Are Swing Applications Old?

First of all why would a manager even care about Swing applications? As long as the development lead knows when to change from Swing to a more Web-centric application, why should a manager even care? The reason I ask this is that you have to remember I come from a technical background. I feel that if a strategic decision has to be made on which technology we should use, I'd like to be part of it. If I were the type of manager who thought Swing was something for my two-year-old son then of course you wouldn't want me in the discussion at all.

We have a number of Swing-based applications that are used by our internal customers and by administration. The Swing-based applications follow a fix process required by every internal developer who wants to create a fix. This fix process is very complicated and requires an internal developer to run a number of the Swing applications so a fix can be created, tested and deployed to external customers. There have been a number of developers who have implemented additional functionality within the Swing applications. Over time, this has made some of the code hard to read. There is logic that is duplicated because a developer was not aware of particular methods that already existed. There are also a number of classes that were implemented that do not fit within the old design because of the changing functionality. Instead of enhancing the old design, now a new design and old design exist within the application. This, of course, has nothing to do with the debate over whether Swing-based applications are old but does create additional work if you were to migrate the applications from Swing to a Web-based tool. Time would have to be spent to understand the differences between the old design and new design. Eventually, a design bringing both of them together would have to be created.

From a manager's perspective I see a couple of questions that have to be answered when looking at migrating Swing applications to Web applications. First, does my customer need this? Currently, a number of different commands are run on an AIX or Windows command line to run the Swing applications. The internal customers feel it's easier to go to a browser instead of understanding command-line syntax to run the applications.

Do I have the skillsets on my team to transfer the Swing application to a Web-based tool?

My team is very skilled in Java and has experience in creating Web applications. I worry that there'll be problems transferring all the logic from the Swing-based applications to the Web-based applications correctly because of the current design problems that exist. However, this would be a perfect time to analyze the problems and correct them.

Are there enough people on my team to do this? This is the extremely hard part of being a manager. Estimating how long a project can take is not a fine art. If the estimation isn't done properly, time is wasted or not enough employees are allocated. The migration requirements must be gathered and sized. Once sized, a manager must support his development leads with the resources that they need.

These are just a couple of the things that I've chewed on my first month of experience. Perhaps, other first-time managers have had the same things happen in their department. Making management decisions is not an exact art but hopefully the situations I've described give you an idea of one approach.

"Where does a manager start when **taking over a Java development team?"**

# Write Right Java Faster Using Test-Driven Development

*The benefits of embracing TDD*

by Richard Cariens
& John Evans

Testing Java code is increasingly a task taken on by developers rather than separate teams to which the programs are handed. Many Java developers are now familiar with JUnit and know the different between unit tests and integration tests. This has been driven largely by the focus on test-driven development (TDD) in extreme programming (XP) and other agile software development methodologies. While the industry-at-large has recognized the value of unit tests and has a new outlook on testing in general, for the most part, actual TDD (meaning, the tests are written first) is not usually practiced outside of hardcore agile shops.

In this article, we'll present a specific example (based on a real-world scenario that we recently faced) and walk step-by-step how to take a pure TDD approach and hopefully show the benefits of embracing TDD completely in this scenario. (For a clear and concise explanation of some of the major benefits of TDD in general, see http://www.extremeprogramming.org/rules/testfirst.html.)

## The Scenario

This scenario is modeled closely on one we faced at a client site recently. In short, we were a pair on a development team working on a project with typical issues:

1. A deadline/delivery date had been set
2. Little or no requirements existed and
3. It didn't look like we'd be getting requirements any time soon (due to limited staffing, etc.).

The project goal was to build a marketing Web site around the client's existing feed management product. At a high level and besides product marketing, the Web site should include basic information and some rudimentary services related to Web feeds (RSS, Atom, etc.). The list of services included:

1. A "feed finder" service: the user must be able to enter a URL somewhere on the site that will produce a list of candidate feed URLs that it found at that URL.
2. A "feed validation" service: the site will analyze a user-provided URL and inform the user if the document found at that URL is a valid RSS or Atom feed.

To mimic the last-minute changes in requirements we'll imagine that a business stakeholder stopped by our cubicles and provided some additional information this morning: the site must have a "coolness" factor, i.e., make the site an active Web 2.0/AJAX site. A new person has been hired to handle the user interface, HTML and JavaScript; our job is to build remote components that will implement the services listed above.

## Constraints

The client's standard production platform is Java 5, JBoss 4.x, and MySQL 4.1 on Red Hat Linux. We're supplied with a workstation running Windows XP Pro, Eclipse 3.1, Java 5, and JBoss 4.0.1.

## Decisions

We decided to assess the risk level of each service. RSS and Atom standards are well known and there are a variety of tools that we can probably use to implement the "feed validation" service, which doesn't feel that risky. The "feed finder" service feels much riskier since there are many ways to detect a candidate feed for a supplied URL, some of which are:

1. We can try to discover the feed from the HTML document at the supplied URL using <link> elements in the HTML <head> section;
2. We can spider the URL's Web site for common feed file names like rss.xml, atom.xml, rdf.xml, feed.rss, etc.;
3. We can try to get the feed URL by using a Web Service like Syndic8's XML-RPC services.

We're not too worried about the service interfaces with the AJAX pages; we know we'll probably have to write a servlet that accepts a GET or POST request with a URL parameter. And while we're not sure if we'll use XML or plain text in the response, we think that's a straightforward problem. We proceed to tackle what we think are the risky unknowns (like how to find feed references from a document) and start by writing a test for the discovery method of "feed finding."

**Richard Cariens** is an independent software consultant in the Washington D.C. area (www.jpevans.com). He has over 10 years of experience testing, developing, designing, and architecting Internet technology and financial systems. Rich holds an MS in computer science from George Mason University in Fairfax, Virginia.

**John Evans** is the founder and president of JPEvans, Inc. (www.jpevans.com), a small independent computer consulting company based in Northern Virginia just outside of Washington D.C. John has over 10 years of professional experience in software development. He has successfully developed and deployed large-scale software systems for several large multi-national corporations.

## Getting Started

We think it will be easier to write the tests if we break up the service into two discrete steps:

1. Download the HTML document from the supplied URL;
2. Parse/search the document for <link> tags with "type" attributes of "application/rss+xml," "application/atom+xml," "application/x.atom+xml," or "application/x-atom+xml."

First we create a simple empty Eclipse project for our feed finder and add a new JUnit Test Case (see Figure 1). Then we call the test "FeedAutoDiscovererUnit".

For starters, we add a test for detecting RSS links that automatically fails (since we don't have anything to test yet) and we're ready to start defining success and failure criteria.

Note that even though we haven't written any application code, we know that we're probably headed towards creating a class called FeedAutoDiscoverer and that we're expecting this class to be able to find an RSS link in a document.

## Writing the Test

Now we have to generate some input HTML that contains a <link> tag with a type of "application/rss+xml" (the Atom test will come later). We add a simple in-line HTML document that contains the expected link to our "testFindsRssLink" method.

Now we're ready to introduce the component that will implement the discovery logic. We replace the "fail" statement with a call to an instance of a class called FeedAutoDiscoverer. We decide that this class should implement a method called "discoverLinks" that accepts a string and returns a list of strings. We also add assertions that help us know if the FeedAutoDiscoverer discovered the RSS link type correctly:

1. We know that our test input only has one <link> tag in it so we assert that the FeedAutoDiscoverer returns a list with one element;
2. We know that our test input contains a <link> tag with a specific "href" attribute and we want to see that expected href value in the list (see Figure 2).

Note the red decorations on the test source. The FeedAutoDiscoverer class doesn't exist yet so let's use Eclipse's "quick-fix" capabilities to create it for us from this test (press Ctrl + 1 while the red-underlined code has cursor focus).

We tell Eclipse we want this class to live under the "src" source folder instead of the "tests" folder and then let the wizard generate the class for us.

Once the empty FeedAutoDiscoverer is generated, we still find we can't compile our test because the method "discoverLinks" method doesn't exist. We let Eclipse generate this for us as well (see Figure 3).

Eclipse generates an empty method for us with a handy "TODO" reminder that we'll eventually have to change this method.

The test will now compile and is ready for its first run. Of course it will fail, but that's to be expected since our FeedAutoDiscoverer just returns null.

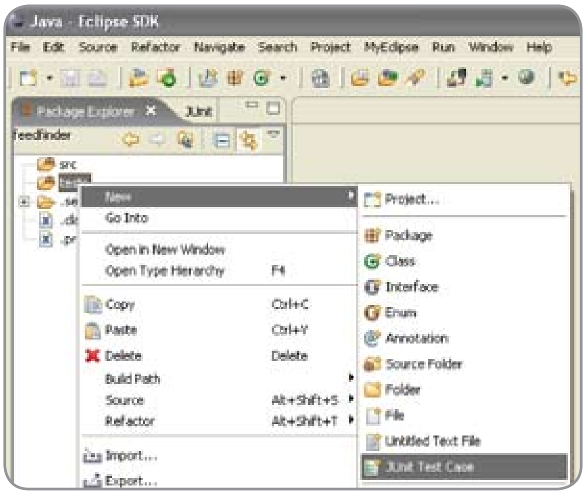
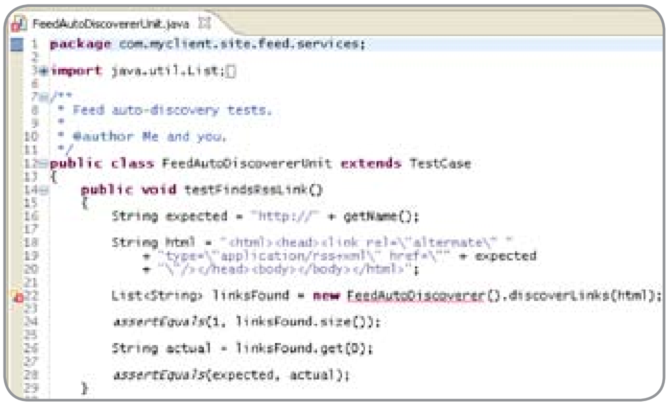


**Figure 1** Creating the first test



FeedAutoDiscovererUnit.java

```
package com.myclient.site.feed.services;

import java.util.List;

/**
 * Feed auto-discovery tests.
 *
 * @author Me and you.
 */
public class FeedAutoDiscovererUnit extends TestCase
{
    public void testFindsRssLink()
    {
        String expected = "http://" + getName();

        String html = "<html><head><link rel=\"alternate\" "
            + "type=\"application/rss+xml\" href=\"" + expected
            + "\"/></head><body></body></html>";

        List<String> linksFound = new FeedAutoDiscoverer().discoverLinks(html);

        assertEquals(1, linksFound.size());

        String actual = linksFound.get(0);

        assertEquals(expected, actual);
    }
```

**Figure 2** Test-driven design

Now that we've written the test, we can focus on making the test pass.

## Making the Test Pass

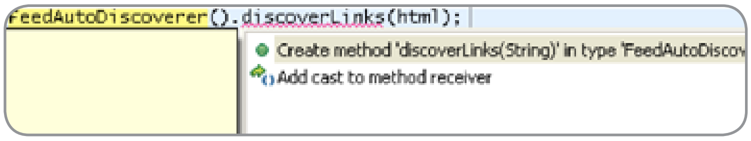
We're going to try to write the simplest code that will make the test pass. We know there are several ways to detect a <link> tag in an HTML document:

1. We can sub-class the HTMLEditorKit.ParserCallback from the javax.swing.text.html package;
2. We could use a third-party HTML/XML parsing library like TagSoup, HotSax, NekoHTML, JTidy, etc;
3. We could use regular expressions and other "brute force" techniques.

We decide to go with option 3, specifically Java regular expressions, since there's some resistance to introducing a new and uncertified framework at the client site. We start fleshing out the FeedAutoDiscoverer by making sure it returns a list of strings. We know that the regular expression must match <link> tags with a "type" attribute of "application/rss+xml." We'll also want to extract the "href" attribute from all matching tags, so we should be sure to wrap the "href" attribute value in a capturing group. Our first version of the FeedAutoDiscoverer looks similar to Figure 4.

Lucky for us, the test passes!

After congratulating ourselves on our success, we realize



```
FeedAutoDiscoverer().discoverLinks(html);
```

Create method 'discoverLinks(String)' in type 'FeedAutoDiscov
Add cast to method receiver

**Figure 3** Generating the method

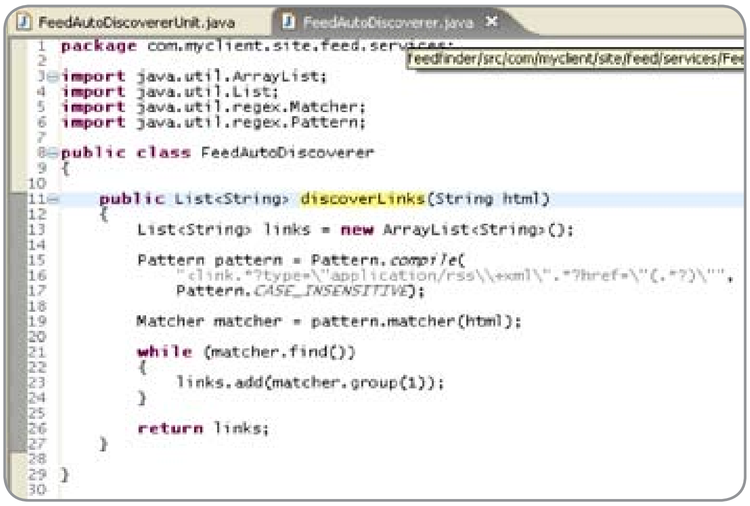
that the <link> attributes might appear in any order and with embedded new lines. We decide to write a second test to handle this new scenario. This test case switches the "type" and "href" attribute ordering and adds a new line between them.

Sure enough, we find that our FeedAutoDiscoverer doesn't handle this scenario; the new test fails.

We open up the FeedAutoDiscoverer and tweak the regular expression so that it'll hopefully handle the new scenario:

1. We add an OR operator to the original expression and switch the order of the "type" and "href" attributes in the new half;
2. We add the DOTALL flag to the expression, which allows the '.' character to match new lines.

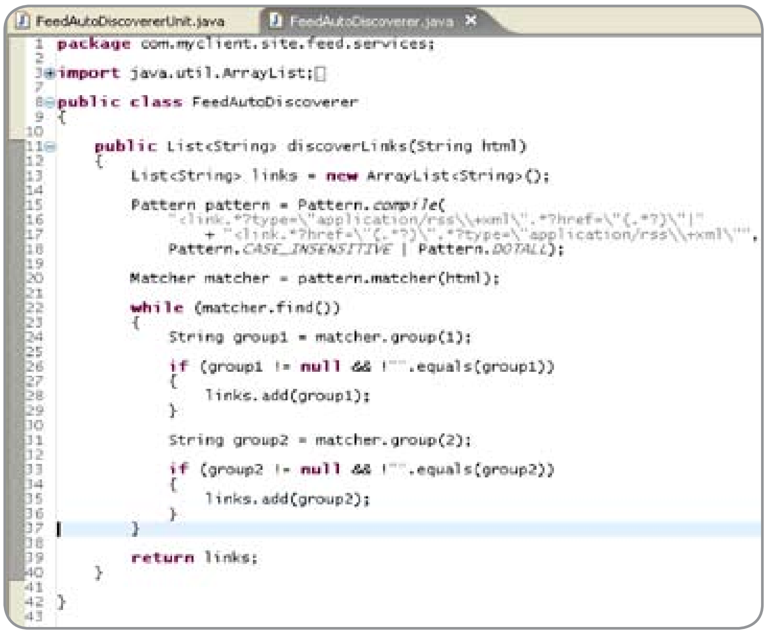
Now the FeedAutoDetector looks like Figure 5.



FeedAutoDiscovererUnit.java
FeedAutoDiscoverer.java
feedfinder/src/com/myclient/site/feed/services/Fe

```
package com.myclient.site.feed.services;

import java.util.ArrayList;
import java.util.List;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class FeedAutoDiscoverer
{

    public List<String> discoverLinks(String html)
    {
        List<String> links = new ArrayList<String>();

        Pattern pattern = Pattern.compile(
            "<link.*?type=\"application/rss\\+xml\".*?href=\"(.*?)\"",
            Pattern.CASE_INSENSITIVE);

        Matcher matcher = pattern.matcher(html);

        while (matcher.find())
        {
            links.add(matcher.group(1));
        }

        return links;
    }

}
```

**Figure 4** A good start



```
package com.myclient.site.feed.services;

import java.util.ArrayList;

public class FeedAutoDiscoverer
{

    public List<String> discoverLinks(String html)
    {
        List<String> links = new ArrayList<String>();

        Pattern pattern = Pattern.compile(
            "<link.*?type=\"application/rss\\+xml\".*?href=\"(.*?)\"|"
                + "<link.*?href=\"(.*?)\".*?type=\"application/rss\\+xml\"",
            Pattern.CASE_INSENSITIVE | Pattern.DOTALL);

        Matcher matcher = pattern.matcher(html);

        while (matcher.find())
        {
            String group1 = matcher.group(1);

            if (group1 != null && !"".equals(group1))
            {
                links.add(group1);
            }

            String group2 = matcher.group(2);

            if (group2 != null && !"".equals(group2))
            {
                links.add(group2);
            }
        }

        return links;
    }

}
```

**Figure 5** Capture group handling

Running the test again results in failure, but it looks like the regular expression may have worked since it's the contents of the list that are wrong. Reviewing our changes shows us that we forgot to update our capture group handling. We have two capture groups in the regular expression and have to add the proper group contents depending on which half of the regular expression worked, so we change our logic.

We find that the test now passes and we're ready to move on to adding more tests for auto-discovering Atom link references, etc.

There's a lot more to do, but we're well on our way to delivering the site "Feed Finder" service. If we wanted to, we could stop here with auto-detection and start building out the downloader component and the servlet, or we could extract an interface from the FeedAutoDiscoverer and start work on fitting it into an IoC container like Spring. Whatever we decide to do, we now have two tests that we can use to sanity-check changes we make from this point forward that might impact feed auto-discovery. If these tests continue to pass as we change the system (and we should run our tests often), we'll become more confident and comfortable with handling change.

(As an aside, at this point in real life, we discovered that there was actually a spec for ATOM-based feed auto-discovery at http://philringnalda.com/rfc/draft-ietf-atompub-autodiscovery-01.html. So, we took each of the examples in the RFC and coded them up as unit tests, and made sure that our FeedAutoDiscoverer successfully discovered them.)

## Conclusion

Test-driven development helped us in many ways on this project:

1. It forced us to translate our ambiguous requirements into verifiable test criteria;
2. The test criteria helped us focus on doing just what was needed to pass the test, and thus hopefully satisfying the requirements;
3. We avoided the heavy front-loading of design documentation and focused on getting some working code.

While we'll ultimately write more code with this approach we'll have fewer defects in the end product. Testing early will also help us uncover requirements errors or changes that would be expensive to address in the later stages of the project.

## References

- http://extremeprogramming.org/rules/testfirst.html
- http://extremeprogramming.org/stories/testfirst.html
- http://en.wikipedia.org/wiki/Test_driven_development
- http://www.testdriven.com/
- http://www.aaronsw.com/2002/feedfinder/
- http://www.syndic8.com/web_services/
- http://java-source.net/open-source/html-parsers
- http://mercury.ccil.org/~cowan/XML/tagsoup/

Welcome to the Future of Video on the Web!

**REGISTER NOW!**
**www.iTVcon.com**

CALL FOR PAPERS NOW OPEN!

**LIVE SIMULCAST!**
**AROUND THE WORLD ON SYS-CON.TV**

iTVCON.COM INTERNET TV CONFERENCE & EXPO 2006

Coming in 2006 to New York City!

"Internet TV is wide open, it is global, and in true 'Web 2.0' spirit it is a direct-to-consumer opportunity!"

**For More Information, Call 201-802-3023 or Email itvcon@sys-con.com**

## Welcome to the Future!

Did you already purchase your ".tv" domain name?
You can't afford not to add Internet TV to your Website in 2006!

2005 was the year of streaming video and the birth of **Internet TV**, the long-awaited convergence of television and the Internet. Now that broadband is available to more than 100 million households worldwide, every corporate Website and every media company must now provide video content to remain competitive, not to mention live and interactive video Webinars and on-demand Webcasts.

20 years ago the advent of desktop publishing tools opened the doors for the creation of some of today's well-known traditional print media companies as well as revolutionized corporate print communications. Today, with maturing digital video production, the advent of fully featured PVRs, and significant advances in streaming video technologies, **Internet TV** is here to stay and grow and will be a critical part of every Website and every business in the years to come.

It will also very rapidly become a huge challenge to network and cable television stations: **Internet TV** is about to change forever the $300BN television industry, too.

The Internet killed most of print media (even though many publishers don't realize it yet), Google killed traditional advertising models, and **Internet TV** will revolutionize television the way we watch it today. You need to be part of this change!

Jeremy Geelan

***Jeremy Geelan***
*Conference Chair, iTVCon.com*
*jeremy@sys-con.com*

PRODUCED BY SYS-CON EVENTS

**List of Topics:**
- > Advertising Models for Video-on-demand (VOD)
- > Internet TV Commercials
- > Mastering Adobe Flash Video
- > How to Harness Open Media Formats (DVB, etc)
- > Multicasting
- > Extending Internet TV to Windows CE-based Devices
- > Live Polling During Webcasts
- > Video Press Releases
- > Pay-Per-View
- > Screencasting
- > Video Search & Search Optimization
- > Syndication of Video Assets
- > V-Blogs & Videoblogging
- > Choosing Your PVR
- > Product Placement in Video Content
- > UK Perspective: BBC's "Dirac Project"
- > Case Study: SuperSun, Hong Kong

| | |
|---|---|
| **Track 1** | Corporate marketing, advertising, product and brand managers |
| **Track 2** | Software programmers, developers, Website owners and operators |
| **Track 3** | Advertising agencies, advertisers and video content producers |
| **Track 4** | Print and online content providers, representatives from traditional media companies, print and online magazine and newspaper publishers, network and cable television business managers |

**Joe Winchester**
Desktop Java Editor

# Who does Business Logic ?

One of the phrases that has always puzzled me is "business logic". It seems to crop up a lot in presentations, articles, sales pitches and so forth. The one I saw it in most recently was a talk about how great web servers are because they keep all of the business logic on the server where it can be robust, secure, and logged. By analogy the client is a poor place for business logic because, while it can do richer things with the user interface, all of the core rules must be kept on the server.

It's not the first time I've come across server heads who use this argument, that their box has to be the gatekeeper for all of the hard and important corporate logic. By using the adjective "business" they're sort of belittling the desktop in any client server equation to be good for nothing more than fancy editing controls and salad dressing the user experience.

The problem is that often when you push people for what business logic really means, it boils down to something like "this value can't be larger than the sum of these values" or "this date can't be before this date for this kind of transaction". It's an answer that more often than not sounds to me like something the GUI is not only perfectly capable of doing, but is probably most sensibly done on the desktop. After all, it can notify errors instantly on mouse and keyboard events, and provide completion assistance and help without incurring the latency of an expensive server round trip.

Wikipedia describes "business logic" with the following sentence:

*"Take a spreadsheet, for example. The spreadsheet in itself is a generic tool and embodies no business logic as such. When you use the spreadsheet by encoding formulas which calculate values of importance to your organization, then you are encoding business logic"*

For any server guy reading this, a spreadsheet is a desktop application. However, the key phrase in the definition above is "importance to your organization, then you are encoding business logic".

From that definition I think that all code any of us have ever written is business logic. I assume of course none of us have ever written stuff that wasn't important to whichever organization was paying our salary at the time.

Why then is there such a mystique about the phrase ? I think it's because as soon as the adjective "business" is placed around something it means that it's more important to the organization and therefore attracts the attention of managers, accountants and analysts. Business modeling is something done by analysts (proper analysts, not people who write specs for programs that developers have to stay at work late and write) where they take apart the mechanics and structure organization of an organization in attempt to apply change management and restructure its processes to be more efficient and cost effective in future. A Masters of Business Administration studies for three years or more to understand this in depth, hoping for a destiny in the echelons of senior management to perfect and apply their skills. There are even executive MBA programs for those who are aiming even higher up the corporate ladder. I wonder whether MBAs drill into people a subliminal Pavlovian association that make its graduates salivate each time the word "business" is used to prefix an otherwise boring task, such as coding spreadsheet cells.

It's not just business logic that one can dissect in this way, but there are a slew of terminologies such as "business process execution language", "business event publishing", or "business process modeling notation". If you dig hard enough behind the sea of white papers and PowerPoint charts surround these however, you'll find that at the core of each is some plain old-fashioned, unfashionable, boring old code. "When value foo reach values a limit moo write value foo*100 to buffer boo that program goo reads and updates database yoo with".

There is benefit in abstracting lines of code to higher level units. Both from the benefits of modularity and re-use, while object-oriented programming further reifies blocks of work to become recognizable tasks and functions around anthropomorphic functions. What troubles me though, is when just because someone has grabbed a trendy name for what's basically just code, and then denigrates those who aren't using their coding technique as being fat, thick, poor, or whatever other insult they can dream up, allowing them smugly preaching the benefits of the new "business logic application hardware" (BLAH) technique they created with impunity.

We all write business logic. From games programmers, to COBOL guys, through Java, Visual Basic, and spread sheet macro heads. A good rule of thumb I think is to always apply the wikipedia test, which is when coding or designing, to continually question the importance of what you're doing to the organization for whom the program is being built.

Business logic can, and does, run anywhere, in any language, on any platform. Next time you see an over the top presentation being given by someone who dresses up their newfangled architecture with the "business" adjective start questioning them hard and peel back the layers of their onionware. You'll find that behind the robes there's just some code served up in an alphabet soup of acronyms to make it current and confusing. Then question whose benefit this is for. The customer for who the application is going to work, or the company whose consulting services are behind the presentation. Seems pretty logical to me.

**Joe Winchester** is a software developer working on WebSphere development tools for IBM in Hursley, UK.
*joewinchester@sys-con.com*

# Eclipse: a Solid Desktop, Rich-Client, or Embedded Application Framework

Todd Williams

*A general purpose platform*

By now, you've probably heard about Eclipse as "the Open Source Java IDE" (http://www.eclipse.org). Today, several companies have looked past the Java IDE plug-ins provided as part of Eclipse, and are creating products that use Eclipse as a tool integration platform, both inside and outside of the Java arena. But what about using royalty-free, Open Source Eclipse technology as a general-purpose application framework for your next desktop, fat client, or embedded application? With the support provided by the Eclipse Rich Client Platform (RCP) and the embedded version of the same (eRCP) the idea is certainly not as strange as it first sounds. So we'll explains why Eclipse is a solid desktop, rich-client, or embedded application framework with the potential to greatly simplify and accelerate development as well as forever change the way developers think about writing Java applications.

Software development is often about compromises. Currently, one of the most visible debates centers on the tradeoffs between productivity applications and browser-based UIs. In spite of what current media coverage might lead one to believe, the industry hasn't decided to throw away all its desktop applications in favor of browser-based UIs rendered with some combination of HTML/XML/XSLT/Javascript. The reason can be summed up in three words: "the user experience." Form follows function… is the key criteria for judging usability. In practice, high user interactivity or complex data relationships make delivering user interfaces as a desktop application less of a choice and more of a requirement.

In today's computing environments it's important to deliver user interfaces that can run on a wide variety of platforms. The range is broad – including small handheld devices as well as server consoles. When users interact with applications in the window management environments they're most familiar with, using the application must feel natural and predictable.

Building a productivity application means starting with a good design and a supportive architecture. Since there's no universally accepted application framework, most developers design their own architecture and then build it into a framework. However, the cost of this approach is considerable expense, time, debugging, support, and aggravation expended on solving a problem that's peripheral to building the functionality of the intended application.

A much better approach than "rolling your own" application framework would be to find one that could fulfill the design requirements while simplifying and accelerating project development. A "wish list" for such a framework would likely contain the following:

- Implements a clear, consistent, and cohesive architecture
- Supports development and execution on all the major desktop platforms (Windows, Mac OS X, Linux, QNX Photon, Pocket PC, HP-UX, AIX, Solaris)
- A snappy UI response that maintains the platform's native look-and-feel
- Provides a large variety of widgets, both standard (i.e., button and checkbox) and extended (i.e., toolbar, tree view, and progress meter)
- Provides extensive text processing that includes editors, position/change management, rule-based styling, content completion, formatting, searching, and hover help
- Supports using platform-specific features (i.e., ActiveX) and legacy software, if desired
- Enables branding the application
- Contains an integrated help system
- Manages user configuration and preferences
- Supports remote discovery and installation of application updates
- Created and backed by respected software companies experienced in creating object oriented frameworks
- Supports internationalization and national language translation
- Designed for flexibility with natural features for adding new functionality
- "Pay" only for what you need – base frameworks can be easily reduced as well as extended to tailor capabilities to precise requirements

To complete our "wish list" we might as well add that the technology be used and supported by a multi-industry charitable foundation, created and maintained by an Open Source community, royalty-free and licensed to provide worldwide redistribution rights. Although these requirements may sound like a pipe dream, it's likely that Java application developers already have this incredible application framework installed. It's Eclipse.

## Can Eclipse Be Used as an Application Framework?

The Eclipse Project FAQ say "The Eclipse Project is an Open Source software development project dedicated to providing a robust, full-featured, commercial-quality industry platform for the development of highly

**Todd Williams** is Genuitec's VP of technology and leads its Eclipse Technology Consulting Practice. He has over 20 years of industry experience in developing computing infrastructures, large-scale distributed software architectures, and optimizing development processes, techniques, and tools. Todd has been Genuitec's representative to the Eclipse Foundation since 2002 and currently holds an elected seat on the Eclipse Foundation's board of directors.

integrated tools." So by definition, Eclipse is an open platform for tools integration, not an IDE. The issue has been confused because a complete industrial-strength Java IDE is available in the Eclipse Platform in the form of plug-in components that extend Eclipse's basic framework facilities.

Eclipse provides the framework for combining disparate tools into a single integrated application with a seamless user interface. New tools are integrated into the Eclipse Platform and its user interface through plug-ins that extend Eclipse's facilities and provide new functionality. Eclipse plug-ins can also extend other plug-ins. When an Eclipse-based application initializes, it discovers and activates all of the plug-ins that have been configured for the environment. An Eclipse application is quite literally the sum of its parts since it's capable of performing any function that has been added to it by the plug-ins it currently contains.

Since being able to write and test such plug-ins is essential to the success of Eclipse, the Eclipse Platform is bundled with a plug-in development environment (PDE) and a set of Java development tools (JDT) to support it. Eclipse's developers clearly trusted the power of the frameworks they created. The entire development environment is just another set of tools integrated into the platform using the standard plug-in techniques. The Eclipse Platform itself was itself created by developers using the Eclipse-based Java IDE (initially in beta form). And, since it's Open Source, anyone can inspect the code and understand in great detail exactly how the frameworks are supposed to be used.

It's this practice of packaging the development tools with the platform that causes some people to be confused about the nature of Eclipse. The JDT components are so effective that they're attractive to all Java developers, not just those writing plug-ins. On the surface, Eclipse appears to be just an excellent Java IDE. But instead of thinking about Eclipse simply as a Java IDE, try to think about it as a productivity application that happens to include a Java IDE built using the underlying Eclipse Platform as an application framework.

## Eclipse Framework Features

Eclipse embodies an extensible design that maximizes its flexibility as an architectural platform. At its core, the Eclipse Platform contains an efficient implementation of the OSGi R4 core framework specification known as Equinox, which is used to bootstrap the application. Up from that, the Eclipse architecture defines sets of layered subsystems that allow it to be used as a framework for a portable application (or suite) that's not an IDE at all. And, since the frameworks are layered and coupled only at distinct architectural interfaces, an application can be built by combining only the frameworks it needs, while eliminating those that it doesn't.

The following sections describe the primary Eclipse features that make it attractive as a general application framework.

### Extensibility Model

Requirements change over time so developers often expend considerable effort designing applications so that they're flexible and extensible. Eclipse is built around a highly flexible and extensible plug-in model so any type of capability can be added to the platform. If an application can be thought of as a tool, or set of tools, it immediately becomes apparent that its functionality can be added to an Eclipse-based framework as a set of plug-ins just as Eclipse's native Java IDE capabilities have been.

### Content Model

Eclipse provides a content model built around the concept of a workbench in which tools (capabilities) can be installed. The tools operate on resources organized into projects in the workspace. Projects contain a tree structure of resources, which are folders and files containing any type of content. The core platform provides a large number of extension points that allow the customization of all aspects of resource lifecycle management.

The hierarchical, categorized nature of the content model lends itself to many kinds of productivity applications with a bit of thought. For example, a simple e-mail client could be built on a workspace that contains a single project associated with the user's e-mail account. The user's project could contain folders for common functional e-mail elements such as inbox, outbox, and sent items. Each of these folders could contain the corresponding set of e-mail messages as project resources.

### Native Widgets

The Eclipse platform contains a standard widget toolkit, SWT, implemented natively on all supported Eclipse platforms. SWT contains a large set of events, layout managers, and widgets. When a supported platform doesn't contain a native widget supported by Eclipse, such as a toolbar on Motif, an emulated widget for that platform is provided. SWT also interacts with native desktop features, such as drag-and-drop. SWT can also use OS-specific components, such as Windows Active/X controls, if such functionality is more desirable than full platform portability. So far,

### About MyEclipse

An innovative, comprehensive, and affordable Integrated Development Environment (IDE) for Java, J2EE, and open standards technologies. MyEclipse Enterprise Workbench is a full-featured enterprise-class platform and tool suite for developing software applications and systems supporting the full lifecycle of application development. Facilities and features usually found only in high-priced enterprise-class products are included in MyEclipse, which extends the best practices and technology available from the latest Eclipse 3.1 SDK. Based on open standards and the Eclipse platform, MyEclipse redefines software pricing, support, and delivery release cycles by providing a complete application development environment for J2EE, WEB, AJAX, XML, UML, and databases and the most comprehensive array of application server connectors (25 target environments) to optimize development, deployment, testing. and portability.

### About Genuitec

Genuitec, LLC is an Eclipse-based company offering innovative Java and J2EE development tools. It offers training and expert consulting and development services for the Eclipse SDK and rich client platforms. A sponsor of Eclipse Plug-in Central, Genuitec joined the Eclipse Foundation early in 2003 and is currently on the board of directors, actively participating in the organization's strategic development and direction. Genuitec was founded in 1997 and is headquartered in Plano, Texas.

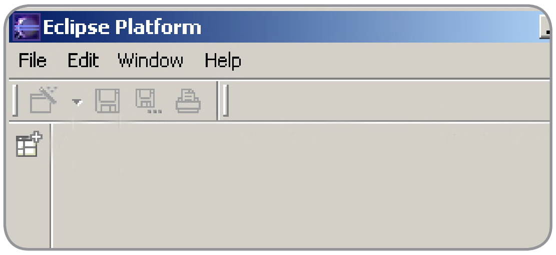


**Figure 1** An "empty" Eclipse-based application

SWT has been proven on the Windows Win32 and Pocket PC, Photon, Motif, and GNU window managers, covering deployment platforms from high-end workstations to embedded devices.

Although the Java language already contains two widget toolkits, AWT and Swing, the Eclipse group still chose to implement their own. The detailed reasons for this choice can be found in the Eclipse Overview white paper. However, to prove that this was the right decision, all one needs to do is compare the look-and-feel of a Swing or AWT application of your choice with that of Eclipse. Eclipse looks, feels, and responds like a native application on whatever platform it's running on.

## User Interface Framework

To build a graphical interface, SWT can either be used directly or through JFace, the user interface framework of the Eclipse platform. JFace includes dialog, preference, progress reporting, and wizard frameworks as well as image and font registries that make user interface creation very straightforward.

The Eclipse platform supports a multi-window, MDI-like user interface presentation. On top of JFace and SWT the Eclipse workbench provides a framework for building perspectives, editors, and views that provide the structure for user interaction. Editors handle resource lifecycle interactions such as creating, editing, saving, and deleting. Views are used to provide supplementary information about an object with which the user is interacting. Examples include outline, pending tasks, and property views. A perspective is a stacked, tiled, or detached arrangement of views and editors. Only one perspective is visible in a window at a time but you can open multiple windows to view multiple perspectives simultaneously.

The Eclipse user interface framework is extensive, flexible, and powerful. And, even if it doesn't do everything you need, it can easily be extended in less time and with fewer resources than designing and building your own.

## Update Manager

Historically one of the biggest problems associated with applications is the support cost incurred to package, distribute, maintain, and upgrade the application as new versions are released. This cost increases when a large and dispersed user community uses the application. With an offering's success and broad deployment, support after the sale can become time-consuming and expensive.

Component maintenance and upgrade facilities were part of the design of Eclipse from the beginning. To control ongoing cost and remove maintenance issues that could become barriers to project development and deployment, the Eclipse platform contains a flexible update manager. The update manager can be configured to initially install new components or updates to existing components from a remote server. As you release new versions of your application or add-on components, distribution can be as

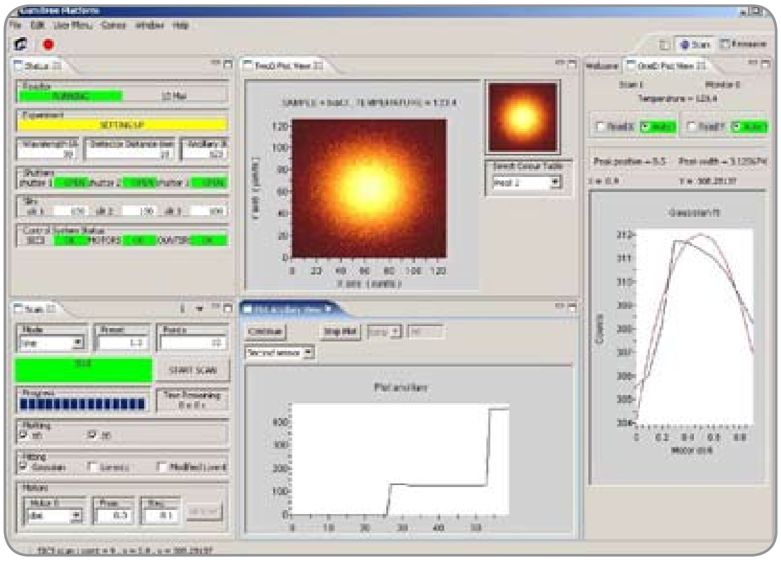


**Figure 2** GumTree

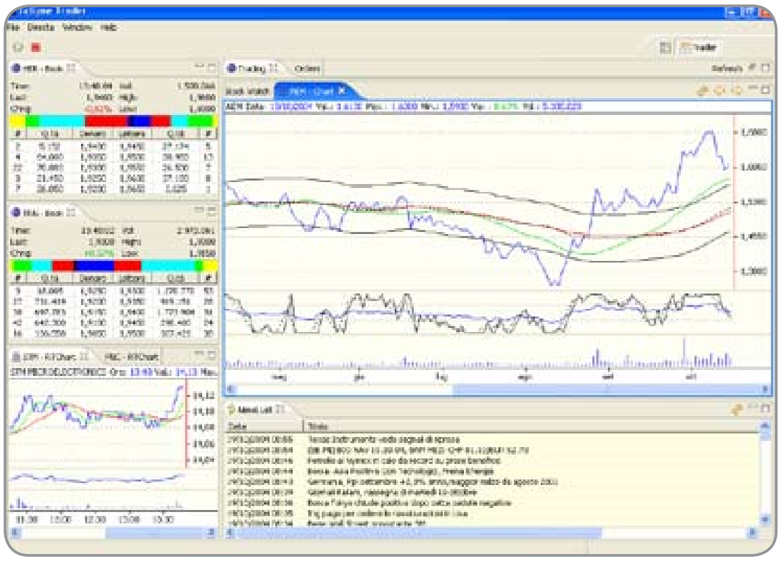


**Figure 3** Eclipse Trader

**SAVE $200! Register by August 18 For Early Bird Savings!**

**The industry's independent conference for enterprise software developers and IT development managers using Eclipse tools and technologies**

**eclipse WORLD**
**The Enterprise Development Conference**

**Hyatt Regency Cambridge**
**Boston, MA**
**September 6-8**

**At EclipseWorld, you will...**

- Learn how to save money and improve developer productivity with Eclipse.
- Improve team collaboration using Eclipse-based tool sets.
- Go beyond the IDE to understand the wide range of Eclipse technologies.
- Get deep inside Eclipse's open-source architecture.
- Discover the best, most effective Eclipse add-ins and plug-ins.
- Learn how to build better applications using the Eclipse RCP.

**"It's a really good place to get those questions answered which are hard to find in books."**
Farha Azaz, Senior Software Engineer, TouchNet Information Systems

**"Great instructors and very good environment, very casual and enjoyable."**
Tim Condron, Business Analyst, Spectra East

**"Technology being discussed in the conference was very leading edge. I got to familiarize myself with projects I hadn't had a chance to research."**
James Pitts, Director of Database Programming, Embarcadero Technologies

**"If you are looking for both high and low level information regarding Eclipse, both as an IDE as well as an RCP, you'll find it at EclipseWorld. The presenters are knowledgeable and approachable. In addition, the networking with other users of the Eclipse platform is invaluable."**
Dan Colbert, Product Manager, CompassCom

**"EclipseWorld had a great collaborative atmosphere. It was exciting to learn about the different uses of Eclipse, and see what people were creating with it."**
Justin Stern, Software Engineer, Auspice

**"EclipseWorld was dynamic, thought provoking and cutting-edge."**
Robert Rothman, Senior Developer, Morgan Stanley

**"Well organized and enjoyed the presentations."**
Alice Chan, Software Engineer, CyberAccess

**"Good sessions on the basics as well as plug-ins."**
Patricia Timms, Senior Software Engineer, Kodak

**"A good opportunity to meet Eclipse developers of all kinds."**
Suzanne Yoakum-Stover, Senior Computational Scientist, SAIC

**"Great place to find out the latest Eclipse info."**
Kelly Hart, Manager, Special Projects, IFMC

**Over 60 great classes to choose from!**

**Keynote Speaker**
Mike Milinkovich
Executive Director
Eclipse Foundation

**Register Now For Early Bird Savings!**

A BZ Media Event

BZ Media is an Associate Member of the Eclipse Foundation.

Diamond Sponsor
COMPUWARE
www.compuware.com

Platinum Sponsor
SYBASE

Gold Sponsors
openmake a product of catalyst
PARASOFT
genuitec
ACTUATE

Silver Sponsor
instantiations

Media Sponsors
SD Times
Software Test & Performance
ECLIPSE review
EclipseSource
queue
Software
eclipse
Methods & Tools
//javareference

For sponsorship or exhibiting information, contact Donna Esposito at 415-785-3419 or desposito@bzmedia.com.

**www.eclipseworld.net**

easy as packaging them using Eclipse facilities and putting them on your update server.

## Help System

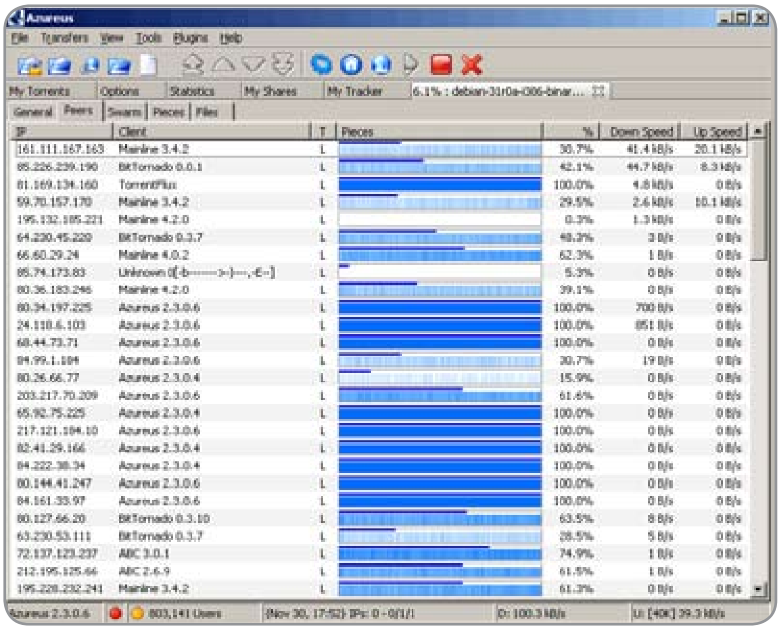
Every professional desktop application has a help system for end users and Eclipse is no different. However, Eclipse's help system isn't simply built from a static group of HTML files that document Eclipse. Rather, it's a framework for providing both searchable and context-sensitive help that's open to extension by documentation plug-ins. As a result, for any application built on Eclipse, everything's available for constructing, packaging, and shipping a complete, custom, context-sensitive help system without buying third-party tools.

## Using Eclipse as an Application Framework

So starting with the underpinnings of a Java IDE as an application framework may at least sound possible, but why would anyone do it? Well, Eclipse satisfies the full function and facilities wish list mentioned earlier, while providing the program development environment for building the project as a series of Eclipse plug-ins. At the outset the frameworks provide an empty, featureless application that is architecturally sound, extensible for future enhancements, and can upgrade itself remotely.

The main question then becomes how much of Eclipse is required? Simply stated, an application can be built on the Eclipse framework by removing functionality that's not important and adding functionality that is. The more challenging part is where to begin. The easiest cases are in the extremes. For example, when building a commercial IDE, like we do with MyEclipse Enterprise Workbench, it makes sense to start with the complete Eclipse Platform download, as well as a few other Eclipse projects, and build on top of them. At the other extreme, when building an application for an embedded device or any other environment where size constraints are paramount, then either Equinox or eRCP would make a more reasonable starting point. If the deployment target has a few more resources, but still don't require the vast majority of the platform's features then using the RCP (available from the platform download page) as the primary framework is likely the right starting point. With a little configuration on the base RCP you can quickly set up an "empty" application, as shown in Figure 1, and then concentrate on adding only what adds value, rather than infrastructure.

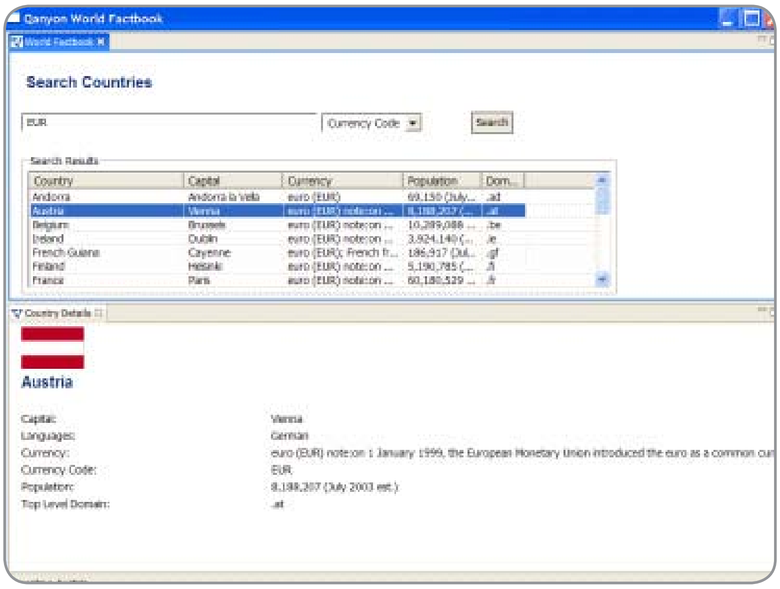
Once the starting platform has been determined, building an application is simply a matter of writing plug-ins to add features to the basic Eclipse framework and branding them appropriately for the intended audience. For example, a large application is typically written as multiple custom perspectives and supporting views using many plug-ins. Alternatively, to integrate a suite of small applications, perhaps each one can be a single perspective in its own plug-in. Along those lines, Eclipse can also be used as a portal to integrate all of a company's homegrown applications. The possibil-



| IP | Client | T | Pieces | % | Down Speed | Up Speed |
|---|---|---|---|---|---|---|
| 161.111.167.163 | Mainline 3.4.2 | L | | 30.7% | 41.4 kB/s | 20.1 kB/s |
| 85.226.239.190 | BitTornado 0.0.1 | L | | 42.1% | 44.7 kB/s | 8.3 kB/s |
| 81.169.134.160 | TorrentFlux | L | | 100.0% | 4.8 kB/s | 0 B/s |
| 59.70.157.170 | Mainline 3.4.2 | L | | 29.5% | 2.6 kB/s | 10.1 kB/s |
| 195.132.185.221 | Mainline 4.2.0 | L | | 0.3% | 1.3 kB/s | 0 B/s |
| 64.230.45.220 | BitTornado 0.3.7 | L | | 48.3% | 3 B/s | 0 B/s |
| 66.60.29.24 | Mainline 4.0.2 | L | | 62.3% | 1 B/s | 0 B/s |
| 85.74.173.83 | Unknown 0[-b------>-)---,-€--] | L | | 5.3% | 0 B/s | 0 B/s |
| 80.36.183.246 | Mainline 4.2.0 | L | | 39.1% | 0 B/s | 0 B/s |
| 80.34.197.225 | Azureus 2.3.0.6 | L | | 100.0% | 700 B/s | 0 B/s |
| 24.110.6.103 | Azureus 2.3.0.6 | L | | 100.0% | 851 B/s | 0 B/s |
| 68.44.73.71 | Azureus 2.3.0.6 | L | | 100.0% | 0 B/s | 0 B/s |
| 84.99.1.184 | Azureus 2.3.0.6 | L | | 30.7% | 19 B/s | 0 B/s |
| 80.26.66.77 | Azureus 2.3.0.4 | L | | 15.9% | 0 B/s | 0 B/s |
| 203.217.70.209 | Azureus 2.3.0.6 | L | | 61.6% | 0 B/s | 0 B/s |
| 65.92.75.225 | Azureus 2.3.0.4 | L | | 100.0% | 0 B/s | 0 B/s |
| 217.121.184.10 | Azureus 2.3.0.6 | L | | 100.0% | 0 B/s | 0 B/s |
| 82.41.29.166 | Azureus 2.3.0.4 | L | | 100.0% | 0 B/s | 0 B/s |
| 84.222.38.34 | Azureus 2.3.0.4 | L | | 100.0% | 0 B/s | 0 B/s |
| 80.144.41.247 | Azureus 2.3.0.6 | L | | 100.0% | 0 B/s | 0 B/s |
| 84.161.33.97 | Azureus 2.3.0.6 | L | | 100.0% | 0 B/s | 0 B/s |
| 80.127.66.20 | BitTornado 0.3.10 | L | | 63.5% | 8 B/s | 0 B/s |
| 63.230.53.111 | BitTornado 0.3.7 | L | | 28.5% | 5 B/s | 0 B/s |
| 72.137.123.237 | ABC 3.0.1 | L | | 74.9% | 1 B/s | 0 B/s |
| 212.195.125.66 | ABC 2.6.9 | L | | 61.5% | 1 B/s | 0 B/s |
| 195.228.232.241 | Mainline 3.4.2 | L | | 61.3% | 0 B/s | 0 B/s |

Azureus 2.3.0.6 | 803,141 Users | {Nov 30, 17:52} IPs: 0 - 0/1/1 | D: 100.3 kB/s | U: [40K] 39.3 kB/s

**Figure 4** Azureus



| Country | Capital | Currency | Population | Dom... |
|---|---|---|---|---|
| Andorra | Andorra la Vella | euro (EUR) | 69,150 (July... | .ad |
| Austria | Vienna | euro (EUR) note:on ... | 8,188,207 (... | .at |
| Belgium | Brussels | euro (EUR) note:on ... | 10,289,088 ... | .be |
| Ireland | Dublin | euro (EUR) note:on ... | 3,924,140 (... | .ie |
| French Guiana | Cayenne | euro (EUR); French fr... | 186,917 (Ju... | .gf |
| Finland | Helsinki | euro (EUR) note:on ... | 5,190,785 (... | .fi |
| France | Paris | euro (EUR) note:on ... | 60,180,529 ... | .fr |

Austria

| | |
|---|---|
| Capital: | Vienna |
| Languages: | German |
| Currency: | euro (EUR) note:on 1 January 1999, the European Monetary Union introduced the euro as a common cur |
| Currency Code: | EUR |
| Population: | 8,188,207 (July 2003 est.) |
| Top Level Domain: | .at |

**Figure 5** Qanyon World Factbook

ities are truly endless. And, just to prove the point, here's a very wide assortment of Eclipse-based applications from all over the world.

GumTree is an Open Source graphical user interface framework for building scientific instrumentation consoles as shown in Figure 2.

EclipseTrader is an Open Source set of plug-ins for the Eclipse RCP dedicated to the building of an online stock trading system, featuring shares price watching, intra-day and historical charts with technical analysis indicators, level II/market-depth views, news watching, and integrated trading. The main view is depicted in Figure 3.

Azureus implements the BitTorrent client protocol through Eclipse RCP plug-ins and comes bundled with many invaluable features for both beginners and advanced BitTorrent users. Azureus is typically one of the most downloaded applications at SourceForge and interface looks native on any platform, thanks to SWT, as shown in Figure 4.

Qanyon World Factbook application was written to explore using Eclipse RCP in a distributed environment. Similar to the CIA World Factbook web site, the Qanyon World Factbook should display country information, albeit in a rich client environment, as shown in Figure 5.

## Going Forward – What's Next for Eclipse?

Eclipse is continuously evolving and will continue to grow both vertically further into the software tools space and horizontally into completely new market segments. Interestingly, the growth into new industry verticals will be for the same reasons that Eclipse was formed in the first place. Although Eclipse was initially formed to build an integration platform for software tool providers, the separate availability of the RCP changes everything. Rather than being a platform exclusively for tool providers, Eclipse has become a general-purpose platform that has simply been leveraged initially in the software tools arena. With this seminal change, Eclipse will begin drawing participants from other verticals who want to cooperate in the same way that the current group of tool providers has. In the near future I expect to see interest in building infrastructure for productivity applications, reporting tools, security, process workflow, and business intelligence among others. Now that Eclipse is completely open and inclusive across the entire software industry, its membership and growth will explode in the coming years.

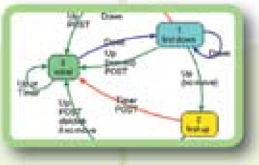
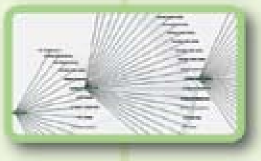
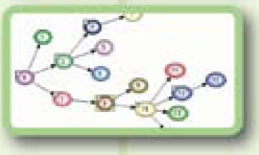
Another vehicle of Eclipse's future growth will likely come from completely outside the software industry. Consortia from such diverse industries as healthcare, automotive, and finance regularly set software platform and interoperability standards. However, without a portable, cross-platform implementation of the standards, each consortium member must independently construct its own, solely based on the industry specifications. This tremendous duplication of work is both expensive and error-prone. Collaborating on building a common set of specification-compliant infrastructure would universally cut costs while insuring interoperability. But what competitors require before they can cooperate is a level playing field that benefits all of them equally. When they begin to research their options, they will find that Eclipse's maturity, extensibility, and royalty-free redistribution model is very attractive as the base for their collaborative development efforts.

Eclipse is constantly expanding, evolving, and surprising all of us. So much so that it would have been impossible to envision where it has gone in its first few years of existence. And, going forward, doing a reasonable job predicting what is next for Eclipse seems just as difficult. There's only one thing for certain; the future is arriving every day and no one really knows what it holds. Software visionary Alan Kay once said, "The best way to predict the future is to invent it." And, whatever the "next big thing" is, one thing is increasingly likely; it will be built on Eclipse.

"Eclipse is constantly expanding, evolving, and surprising all of us. **So much so that it would have been impossible to envision where it has gone in its first few years of existence"**

**Build interactive diagrams easier than you ever imagined**

Create custom interactive diagrams, network editors, workflows, flowcharts and design tools. For web servers or local applications. It's easy-to-use and very flexible. We're the first developer of diagramming components and still the best. Find out for yourself: download our FREE fully functional evaluation kit, with full support at www.nwoods.com.

**FREE Download With Full Support!**

GoDiagram

Interactive diagram components

www.nwoods.com

# A JNI-bridged Java Desktop Application

*Native Performance and Java Control – Bridging Domain Gaps*

Mário de Sá Vera

I'm going to share my experience of enabling a graphics-oriented GIS visualization module with a C++ rendering engine for a Java desktop application using JNI technology.

The solution was implemented in the GIS library TerraLib as part of the TerraLib Develoment Toolkit (Tdk), applying a JNI-bridged drawing canvas as part of the Components API used by the rendering engine.

The solution gave the Java desktop application visualization module a native equivalent performance and saved a lot of duplicative effort in natively implemented rendering functionalities that could be accessed by the Java application layer. It also promoted full integration between the GIS visualization module and the application control peer.

First I'll present the architecture and then discuss how JNI can be a great solution for a well-designed native layers integration. I'll also present, throughout the text, some third-party solutions currently available, giving references and links for more information on this still challenging matter.

## The Architecture

The TerraLib Development Kit – Tdk – core is entirely written in C++. It consists of a framework to make GIS developers' experience with TerraLib easier. As a proof of usage for Tdk, a visualization tool called VIPE (Visualization, Interaction, Printing, and Editing) was successfully built with the Tdk API. This native version of the application is quite stable and designed over an event-oriented model. As we'll discuss later this was a key factor in the proposed solution. I took advantage of that existing design and had the two mixed language control layers communicating through JNI. The other JNI touching point was restricted to the data layer to provide native data state management directly from the corresponding Java data layer. The latter became a very thin layer since it only holds the state management operators (accessor and mutator methods) and the JNI bridge to access the corresponding native data container effectively holding the data (see Figure 1).

**Mário de Sá Vera** is a software architect and works as an IT consultant in Brazil. desavera@netscape.net

## Some Relevant Implementation Details

One very important implementation detail was the need to have the rendering take place on the native visualization layer where the data was formally accessible. But the user experience is actually with a JPanel instance. That made us take the risky strategy of keeping the C++ and Java control layers communicating by sending application state control and visualization UI events and finally applying the Bridge pattern on the Canvas component to enable the native rendering logic control over the Java Graphics2D-based rendering engine.

The Java control layer simply delegates the UI events over to the native layer, which in turn does the response callbacks to the Java Bridge through a jobject proxy. See Listing 1.

We ended up with JVIPE (Java VIPE) having equivalent performance to the C++ version saving development duplication related to rendering the Maps and Cartographic elements – to mention a few. What we do now is implement first in C++ and bridge it to Java. The single data cache was also a major achievement because it wasn't possible with the previous solution using CORBA. We basically succeeded on performance and memory allocation too. The client application inherited JVIPE functionalities through the Tdk Java API and is currently in production.

The drawback is still the unavoidable conversion of the geometry native data types to Java for plotting. I'm currently checking JDK1.4 NIO features for the vector data blocks' native access to the raster data types whose conversion is also very costly.

## JNI Analysis

Looking back at the process that resulted in this successful solution I can risk some conclusions to pass on to those willing to embrace JNI or analyze it against some other technologies.

## Native Code Access from Java Analysis

The first question that we have to answer before deciding to use JNI is when should we try to access native code in a Java development scenario?

The answer isn't as obvious as you may think. Code reuse is a questionable issue. Moving to a modern programming language will usually be a worthy move it because new language efforts usually gather technological improvements together and get rid of past mistakes. Experience says that old libraries will eventually be migrated to modern languages to be continued and improved. So the question becomes, what kind of native code should we directly access against a rewrite approach?

That question is a little simpler to answer. To answer it I'll risk generalizing two answers taken from a design decision:
1. Integration demands combined with manpower or time frame limitations for rewriting the existing code in Java (legacy case)
2. Java considerations for getting an adequate solution as a whole (scratch time case)

We also mustn't forget that not all that's written in a legacy native code can be accessed by Java. Generally speaking I'd say that the best candidates for being kept or developed in native code beforehand would be modules that implement functionalities that Java has trouble with. I mean mostly time-consuming code where Java performance misses (despite JITs, we're still buying new hardware, folks, and with new hardware the native code will run faster!)

Now that we have some idea of when to bring a tower of Babel to our code, let's analyze some technologies.

## The JNI Choice

From my point of view the first analysis to do in deciding to use any technology is its applicability to the situation. It's a good approach considering we'll make serious mistakes using a technology inadequate to our situation. So we could ask ourselves if JNI is really the right technology to access native code from Java.

JNI is a standard Java API. By definition Java demands native resources access. Different technologies could replace JNI however. I've seen some sites analyzing CORBA pros and cons against JNI, and COM has been a choice for Microsoft solution providers. But it's best to analyze the situation you're trying to apply the technology to and then decide if it makes sense.

The first candidate to bridge the two worlds was CORBA since its IDL-based specifications provide language independence and we could take advantage of the client/server technology to create a distributed version of the application. After a couple of weeks of implementing a CORBA middle layer we ended up rewriting the caching model multiple times and wound up with very poor performance and a duplicated cache (the data was loaded first into the native layer and then into the Java visualization layer). All these problems basically stemmed from the separate processes scenario that CORBA – and any other inter-process (IPC) technology – brings in. That recommended JNI as the middle layer.

## JNI Considerations

Accessing external modules written in a language different from the main routine isn't new to most readers. Most languages carry that feature along. We declare some known convention for method prototyping and the two modules can start talking at runtime. With JNI it's no different. Sun's decision to use C as the JNI base for prototyping was quite adequate since most languages can integrate with C code. JNI, though, is an API and requires some level of expertise to be programmed, while linking C code to FORTRAN is a link time task and can be done with only slight understanding of your compiler directives. In JNI the glue must be programmed.

JNI, as an API, offers flexibility but requires some education to use though it's not too steep of a learning curve.

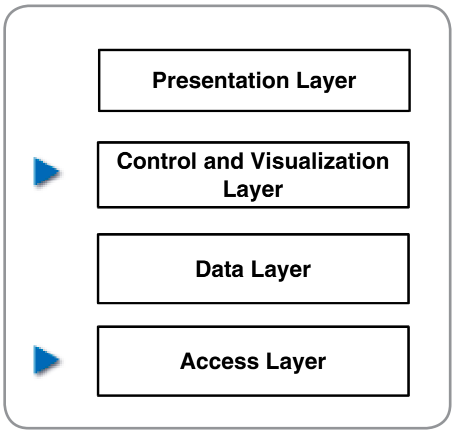


**Figure 1** JNI touching points

There are some products and tools that can be used to help make the JNI experience simpler. One major effort is Noodle Glue – also a Bridge pattern-oriented solution that works as a bridger to the native class to automatically generate a Java wrapper. It's sophisticated and robust, and was being open sourced when last seen. The other is JNIWrapper, the apparent proprietary market leader so far, which follows more of an Adapter pattern model in that it tries to adapt some native types so you can access native resources through direct method calls.

Another painful experience is debugging in JNI. I found ETNUS, a JNI IDE where you can debug Java code and native code in the same environment, but it will cost you some extra bucks. However, the context switching between Visual Studio and Eclipse can be really painful. We explored some hardcore turnarounds like using the "_asm int 3" interruption call on Visual Studio so that we could force an interruption call but that's not elegant.

## Conclusion

At this point I hope I've convinced you folks that JNI is definitely the right choice for native code access from a Java desktop application in the scenarios involving, say, graphics and numerics in general. CORBA causes headaches, for example, not being able to bind to the ORB because of a local host Naming Service misconfiguration.

JNI overcomes limitations in Java solutions, especially performance.

Another insight we've had is that accessing a native application through some inter-process technology doesn't benefit much from knowing other designs. In JNI this is a tighter conversation.

Legacy code should be well designed for access through JNI.

If your legacy code isn't modularized, it will very hard to access cleanly from Java or from other languages. The solution proposed in the JVipe scenario was only possible due to the well-designed event-oriented native layer.

So I take the current number of solutions coming out as motivation as industry jumps on JNI support. Let's hope for more JNI support from IDE vendors.

## References

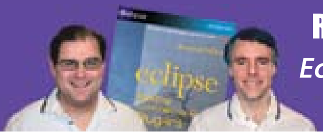
• TerraLib - www.terralib.org
• NoodleGlue - www.noodleglue.org
• JNIWrapper - www.jniwrapper.com
• ETNUS – www.etnus.com

**Listing 1**

```
class TdkCanvas {

 //...
 public void plotLine(TeLine2D& line) = 0;
 //...

}

class TdkJNICanvas : public TdkCanvas
{

public:

 // constructor expects the bridged TdkSwingCanvas instance
        TdkJNICanvas(JNIEnv *env, jobject *jThis);

 //...
 public void plotLine(TeLine2D& line);
 //...

private:

        JavaVM *jvm_;          //Stores the Java virtual
machine
        jobject objCanvas_;  //Stores the Java´s TdkSwingCanvas
instance

}

void TdkJNICanvas::plotLine(jobject& line)
{
        //Gets the current environment
        JNIEnv *env = getCurrentJAVAEnv(jvm_);
        jclass clsCanvas = env->GetObjectClass(objCanvas_);

        //Get the plotLine method ID
        jmethodID metCanvas = getMethodID(env, clsCanvas,
                                        "plotLine", "(Ltdk/
core/geometry/TeLine2D;)V");
        //Runs the method
        env->CallVoidMethod(objCanvas_, metCanvas, line);
}

//-------------

package tdk.graphics;

interface TdkCanvas {

 void plotLine(TeLine2D line);
}

public class TdkSwingCanvas implements TdkCanvas {


 //...
 private Graphics currGraphics_;
 //...

 public void plotLine(TeLine2D line) {

  //...

  // all geoms are Graphics2D Shape compositions
  currGraphics_.draw(line.getShape());

  //...
```

JAVA DEVELOPER'S JOURNAL **Advertiser Index**

| Advertiser | URL | Phone | Page |
|---|---|---|---|
| Quest | www.quest.com/JavaCode | | 2 |
| Altova | www.altova.com | 978-816-1600 | 4 |
| IBM | ibm.com/takebackcontrol/flexible | | 7 |
| Intersystems | www.intersystems.com/Cache17P | 617-621-0600 | 9 |
| Fiorano | www.fiorano.com/downloadsoa | 800-663-3621 | 13 |
| OPNET | www.opnet.com/pinpoint | 240-497-3000 | 19 |
| Roaring Penguin | www.roaringpenguin.com | 613-213-6599 | 23 |
| RogueWave Software | www.roguewave.com/developer/downloads | | 27 |
| Real World Flex Seminar | www.flexseminar.com | 201-802-3020 | 31 |
| Tibco | www.tibco.com/mk/gi | 800-420-8450 | 35 |
| LinuxWorld Conference & Expo | www.LinuxWorldExpo.com | | 37 |
| iTVcon | www.iTVcon.com | 201-802-3023 | 45 |
| EclipseWorld Conference & Expo | www.eclipseworld.net | 415-785-3419 | 51 |
| Northwoods | www.nwoods.com | 800-434-9820 | 53 |
| Instantiations | www.instantiations.com/rcpdeveloper | 800-808-3737 | 57 |
| AJAXWorld Conference & Expo | www.AjaxWorldExpo.com | 201-802-3020 | 58,59 |
| SoftwareFX | www.softwarefx.com | | 63 |
| Parasoft | www.parasoft.com/JDJmagazine | 888-305-0041x3501 | 64 |



**RCP Developer™** is created by the experts who brought you the popular book *Eclipse: Building Commercial Quality Plugins* — Eric Clayberg & Dan Rubel

## Java on the desktop is back!

# RCP Developer™

eclipse READY™

Instantiations RCP Developer harnesses the power of the Eclipse Rich Client Platform. Quickly create native desktop applications that are rich, cross-platform, and high-performance. The vast expertise of the Eclipse developer community and the experience of Instantiations lets you...

## STAND ON THE SHOULDERS OF GIANTS

### Create blazing fast rich Java clients.

Tap into the ultra high performance Eclipse graphical user interface libraries and ensure a native look and feel.

### Build rich Java client applications faster.

Focus on creating business specific functionality rather than reinventing the low-level logic required for applications to run.

### Deliver a more consistent user experience.

Tame the most demanding user interface requirements and reliably deliver a rich and consistent user experience across all applications.

### Fully exploit the Eclipse Rich Client Platform.

"The Eclipse Rich Client Platform is the leading framework for creating Java client applications. RCP Developer is the first product to bring comprehensive application construction, GUI testing and packaging of rich-client applications to Eclipse RCP."

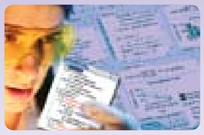
*—Mike Milinkovich, executive director of the Eclipse foundation*

## RCP Developer™

### SWT Designer™

**Design.** Quickly create views, editors, perspectives, and preference pages for your applications. Intuitive visual design and access to the high performance SWT GUI framework of Eclipse RCP.

### WindowTester™



**Test.** Automatically test the GUIs that comprise Eclipse RCP applications. Minimize the need to hand-code GUI test cases. Automated recording, test generation, assertion coding and playback facilities.

### RCP Packager™

**Deploy.** Streamline build and deployment for Eclipse RCP applications. Assemble application elements into a single package. Quickly create and maintain high-quality Eclipse RCP installations.

**Download a risk-free trial copy:**
www.instantiations.com/rcpdeveloper

instantiations
the development tools experts

www.instantiations.com 1-800-808-3737

*The leader in Eclipse RCP development tools*

eclipse FOUNDATION MEMBER

# Rich Internet Applications: AJAX,

## SANTA CLARA SILICON VALLEY

**SYS-CON Events is proud to announce the first-ever AjaxWorld Conference & Expo 2006!**

**The world-beating Conference program will provide developers and IT managers alike with comprehensive information and insight into the biggest paradigm shift in website design, development, and deployment since the invention of the World Wide Web itself a decade ago.**

The terms on everyone's lips this year include "AJAX," "Web 2.0" and "Rich Internet Applications." All of these themes play an integral role at AjaxWorld. So, anyone involved with business-critical web applications that recognize the importance of the user experience needs to attend this uniquely timely conference – especially the web designers and developers building those experiences, and those who manage them.

**CALL FOR PAPERS NOW OPEN!**

We are interested in receiving original speaking proposals for this event from i-Technology professionals. Speakers will be chosen from the co-existing worlds of both commercial software and open source. Delegates will be interested learn about a wide range of RIA topics that can help them achieve business value.

# Flash, Web 2.0 and Beyond...

REGISTER TODAY AND $AVE!

**October 3-4, 2006**

**Santa Clara Convention Center**
Hyatt Regency Silicon Valley
Santa Clara, CA

**To Register**
Call 201-802-3020 or
Visit www.AjaxWorldExpo.com

**May 7-8, 2007**
First International AjaxWorld Europe
**Amsterdam, Netherlands**

***"Over the two information-packed days, delegates will receive four days' worth of education!"***

**Early Bird***
**(Register Before August 31, 2006)** ........................................ **$1,495****
See website or call for group discounts

**Special Discounts***
**(Register a Second Person)** ........................................ **$1,395****
See website or call for group discounts

**(5 Delegates from same Company)** ........................................ **$1,295/ea.****
See website or call for group discounts

**On-Demand Online Access**
**(Any Event)** ........................................ **$695**

*Golden Pass access includes Breakfast, Lunch and Coffee Breaks, Conference T-Shirt, Collectible Lap-Top Bag and Complete On-Demand Archives of sessions in 7 DVDs!

**OFFER SUBJECT TO CHANGE WITHOUT NOTICE, PLEASE SEE WEBSITE FOR UP-TO-DATE PRICING

**"It Was The Best AJAX Education Opportunity Anywhere in the World!"** —John Hamilton

## Topics Include...

**Themes:**
- > Improving Web-based Customer
- > Interaction
- > AJAX for the Enterprise
- > RIA Best Practices
- > Web 2.0 – Why Does It Matter?
- > Emerging Standards
- > Open Source RIA Libraries
- > Leveraging Streaming Video

**Technologies:**
- > AJAX
- > The Flash Platform
- > The Flex 2 Framework & Flex Builder 2
- > Microsoft's approaches: ASP.NET, Atlas, XAML with Avalon
- > JackBe, openLaszlo
- > JavaServer Faces and AJAX
- > Nexaweb
- > TIBCO General Interface

**Verticals:**
- > Education
- > Transport
- > Retail
- > Entertainment
- > Financial Sector
- > Homeland Security

**GROUP DISCOUNTS AVAILABLE:**
— 5 Delegates from Same Company —
for only $995 (each)
— Register a Second Person —
for only $1195

**LIVE SIMULCAST!**
**AROUND THE WORLD ON SYS-CON.TV**

## Receive FREE WebCast Archives of Entire Conference!

The best news for this year's conference delegates is that your "Golden Pass" registration now gives you full access to all conference sessions. We will mail you the complete content from all the conference sessions in seven convenient DVDs after the live event takes place.

► **This on-demand archives set is sold separately for $995**

SYS-CON EVENTS
**For more great events visit www.EVENTS.SYS-CON.com**
VISIT WWW.AJAXWORLDEXPO.COM FOR THE MOST COMPLETE UP-TO-DATE INFORMATION

# Mindreef SOAPscope Server

*The rare distributed development environment*

Brian Barbash

By nature Web Services is a distributed technology. With distribution comes great flexibility for architectural topologies. Components can be strategically placed in different physical locations to optimize performance, maintenance and business processes. In large organizations one physical location may handle sales services, while another delivers contract management. As organizations build Service Oriented Architectures that stitch together these physically dispersed services, distributed development becomes an interesting challenge to overcome. Many collaborative technologies exist today to facilitate better communications and information sharing among workers, but it's rare to find a distributed development environment.

Enter Mindreef SOAPscope Server. SOAPscope Server is a development platform that provides a centralized work environment designed specifically for SOAs enabled by Web Services. Development teams collaborate in specialized virtual workspaces that manage Web Services definitions, messages, recorded actions, simulations, and notes.

## The Development Environment: Creating Workspaces

Mindreef SOAPscope Server is based on the concept of workspaces. As mentioned, workspaces are central repositories that contain the assets of a given Web Services-enabled project. There are three kinds of workspaces:

1. Private – All assets in private workspaces are accessible only to the logged-in user
2. Team – Assets in team workspaces are accessible to any logged-in user
3. Community – Community workspace assets are available to any user in a read-only state, and an editable state to those with accounts on the server

As an example, assume that an organization has separate physical locations for sales, contract management, and master data services (customer, product, etc.). The support teams and developers of these individual services are also located in different physical locations. As part of an effort to improve ties with its trading partners, this company is building an application so buyers can submit price checks and purchase orders using Web Services. During development a workspace will be created in the team area.

When establishing a workspace in Mindreef SOAPscope Server, developers add WSDL definitions referred to as service contracts. As shown in Figure 1, service contracts can be added to a workspace via either a URL or a WSDL file located on the file system. Service contracts can also be added from the developer's other private workspaces, and from all team and community workspaces. For this example, the ContractService and SalesService WSDL files will be added.

With the service contracts loaded, SOAPscope Server presents them to developers in multiple views:

- Overview – Displays the details of a specific service contract as a tree structure. Each operation is an expandable node on the tree in which the operational details are stored including actions and input and output message constructs.
- Documentation – Lists all of the components of the Web Service by namespace.
- Files – Displays the XML files that make up the Web Service definition in a formatted view.
- Coverage – A general listing of usage statistics for a given service. Metrics captured here include total calls, faults, call duration, request size, and response size.

From these views, services can be invoked, analyzed, or updated, and multiple services can be compared to identify differences in their definitions. The developer can also analyze services for best practices. The choices of algorithms to run are Mindreef Basic Diagnostics, WS-I Basic Profile 1.0, and a combination of the WS-I Basic Profile 1.0 and SOAP Binding Profile 1.0. Users have the option of creating their own algorithms from a library of tests.

## Testing & Verifying Services

Every time a developer invokes a Web Service from a workspace, the request and response messages are captured and the event is stored as an action. This serves as a powerful mechanism for testing and debugging. When issues with a service are identified, the messages that produce the issue can be stored and re-sent to verify that the appropriate collective actions have been taken.

Individual actions can also be strung together to create scripts. This provides for testing dependent services. Web Service parameters can be configured to extract their values from variables allowing for the results of one service to serve as the input to another. For the example in this article, the ContractService.GetContractPrice has been configured to put its results in variables. The values include contract number and price. Subsequently, the SalesService.SubmitPO operation has been configured to extract the contract number and price from the configured variables completing the chain of operations.

## Collaboration

All of the features of Mindreef SOAPscope Server mentioned so far are valuable and serve to assist with developing and testing Web Services. However, SOAPscope Server's differen-

**Brian R. Barbash** is the product review editor for Web Services Journal. He is a senior consultant and technical architect for Envision Consulting, a unit of IMS Health, providing management consulting and systems integration that focuses on contracting, pricing, and account management in the pharmaceutical industry.

tiating functionality is in its ability for teams to collaborate on Web Service development. Features of the system that facilitate this include:

- Workspace Notes – Notes in a workspace provide a way to document activities, changes, issues, and other useful information to members. For example, if an issue is identified with the ContractService, the action that re-creates the error and the specific inputs that are associated with it may be documented as a note. Members of the maintenance team for the ContractService now have a centralized documentation repository to identify and resolve the issue, a location into which the resolution may be entered, and an action script to re-create and diagnose the issue at hand.
- RSS News Feeds – These feeds provide information about the workspace and the notes entered. RSS feeds always include the first and last note entered in the workspace. So teams that consume services in a workspace may be notified by RSS when changes, updates, or issues are resolved in the workspace.
- SOAPscope Server Integration – SOAPscope Server lets workspaces be exported to a proprietary format called a Mindreef Reproducible Package. These packages can be transferred to any SOAPscope Server server with all assets intact. Packages can also be stored in alternate systems, such as bug tracking tools, for archiving and reference.

## Simulation

Simulation in SOAPscope Server refers to the practice of creating dummy messages that serve as placeholders for Web Services. This is particularly useful during the development of composite applications and prototyping where not all services are available. During a simulation, SOAPscope Server acts as a service endpoint, responding with the appropriate message template based on the contents of a request, or throwing a SOAP fault when no matching response is found.

Using the ContractService a new operation has been defined called GetEligibility. This operation will determine which contracts a given customer can buy on, if any. The service itself has yet to be developed, so a simulation will be created for this specific operation.

As shown in Figure 2, the simulation for GetEligibility will be based on the value of the attribute “name” in the incoming XML payload. The response, shown at the bottom of the screen, is a hard-coded XML string that represents a generic eligibility value. Multiple simulations can be created, each configured to react to a specific payload, to accommodate different business cases such as customers being eligible for more than one contract. Invoking the simulation is as simple as sending a request from the service client to the endpoint defined for the simulation.

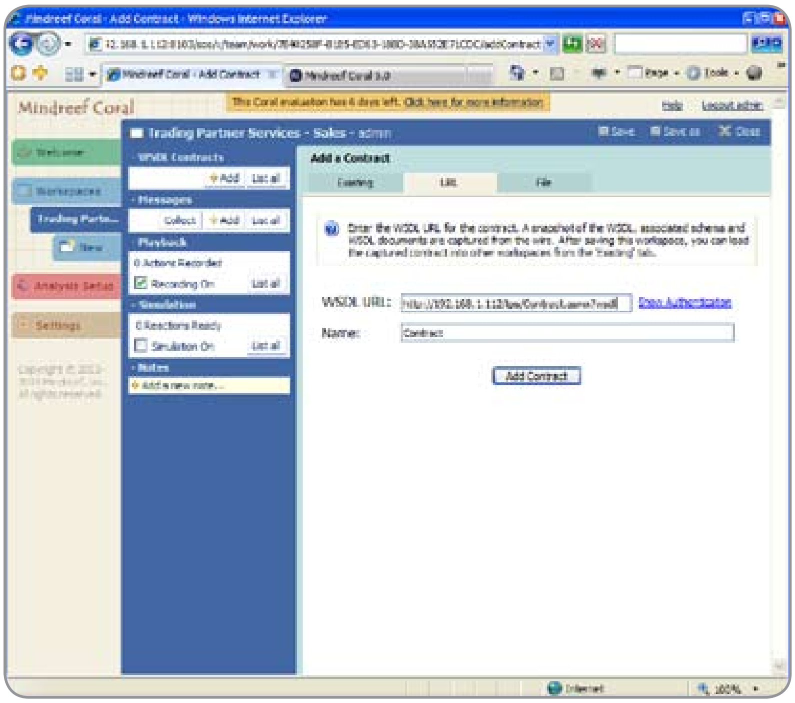


**Figure 1** Adding a service contract

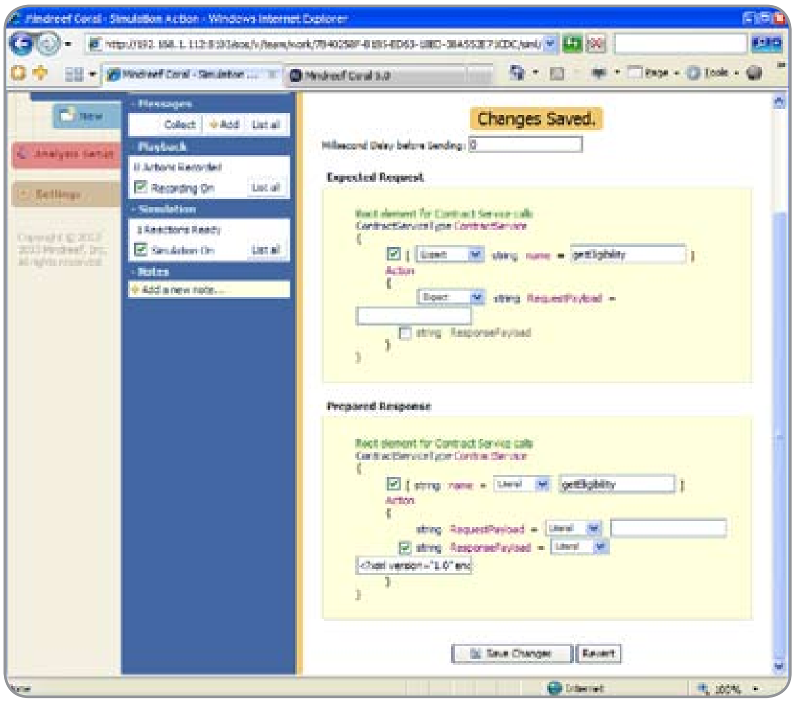


**Figure 2** Simulating a Response

## Summary

Developing Web Services components in a Services Oriented Architecture presents unique challenges. Physical distribution of resources adds to this complexity. Mindreef’s SOAPscope Server platform introduces an interesting solution to this challenge by providing a collaborative work environment that goes beyond the traditional communication functions. The system can play a valuable role in organizations building out new services and supporting existing applications. SOAPscope Server is definitely worth a look.

# The 4th JCP Program Annual Awards Runners-Up

Onno Kluyt

Last month I introduced to you the winners of the 4th JCP Program Annual Awards. But the story is only half told. To get the full picture and understand how tight the competition was, I'm inviting you to meet the runners-up for the JCP Program awards – those who came very close to winning the top honors this year. They are among the top performers to watch in the months and year ahead.

JBoss came close to winning in the JCP Member of the Year category. The company's active participation in the community was noticed by peers and juries. JBoss is involved in a wide range of JSR efforts, including EJB 3.0, Java EE 5, and Web services and provides input and feedback from both developer and user angles.

So did NTT DoCoMo, known in the community as an early adopter of Java technology for handsets. Winner of the "Duke's Choice Award" at JavaOne 2005 for its Java technology-based i-mode FeliCa service, NTT DoCoMo has contributed its rich expertise in running mobile Java services to many of the Java ME Expert Groups in which it participates.

Sun Microsystems, the original creator of the Java technology and specification lead for a wide range of JSRs, was also a runner up in this category. Working with other community members, Sun recently finalized major simplifications to the Java enterprise programming model in JSR 244, Java EE 5, and is in the process of revising the core Java specifications in JSR 270, Java SE 6 (Mustang).

A strong contender for the winner's place in the Most Innovative Java ME JSR category was JSR 248, the first of two specifications developed by the Mobile Service Architecture (MSA) initiative, a group of 14 major mobile industry players represented on the JSR Expert Group. The spec leads are from Nokia Corporation and Vodafone Group Services. The focus of the JSR is to specify an unfragmented, backward-compatible application development platform that supports a set of key APIs to a wide variety of features provided by the latest mass market–oriented handsets and mobile networks today and in the near future.

Led by Motorola and BenQ Corporation, JSR 253, Mobile Telephony API (MTA), stood out too and vied for top honors. The specification enables applications to incorporate telephony features and controls directly into their operation, avoiding the need to pass focus to another application. One example of this might be in a multi-player game that allows a voice call to be placed to other players in the game for strategy discussion (or taunting the enemies). Another use is the ability to incorporate a feature such as calling out to a help desk without exiting the application.

Another contender for the top innovation in the Java ME category was JSR 281, IMS Services API. This API enables application programmers to easily create applications offering multimedia communication services in close integration with IP Multimedia Subsystem (IMS) according to applicable standards. In this way, IMS domain, with all the advantages of merged Internet and telco technologies, will be revealed to the broad Java ME developer community and will encourage faster adoption of the IMS services provided by the wireless networks.

Currently the Spec Lead of JSR 257, Contactless Communication API, and JSR 293, Location API 2.0, Jaana Majakangas of Nokia Corporation came again on the jury's radar screen as a great candidate in the Outstanding Java ME Spec Lead category. Jaana has been a member of the JCP from August 2003 and has consistently participated in JSR development including for JSRs 218, 219, 257, 271, and 293.

Volker Bauche's work for JSR 228, Information Module Profile - Next Generation (IMP-NG), and co-spec leadership of JSR 281, IMS Services API, with other colleagues from BenQ and Ericsson brought him the nomination for Outstanding Java ME Spec Lead. Bauche works in the R&D department of BenQ Mobile as lead of a team specializing in middleware projects.

On the JSR watch list o f developers around the world, JSR 244, Java EE 5, was a natural candidate for the top place in the Most Innovative Java SE/EE JSR category. The JCP EC votes put it in the runner-up race for all the right reasons. Java EE 5 is the most significant release of the Java EE platform since the first version, J2EE 1.2. It enhances the programming model, making it much easier to write enterprise applications. Java EE 5 makes extensive use of Java language annotations to simplify the declarative programming style of Java EE.

Another leading specification, JSR 220, EJB 3.0, was among the top nominations in the Most Innovative Java SE/EE JSR. Its characteristics did not escape the JCP EC jury as highly innovative in a number of ways. It is the first within the group of Java EE specifications that takes advantage of the new features of annotation metadata and parameterized types in Java SE 5. In particular, it defines a strategy for annotations usage subsequently adopted by JSR 244 for the Java EE Platform.

Three great spec leads competed for the top award in the Outstanding Java SE/EE Spec Lead group.

Jose R. Cronembold is a senior development manager at Oracle Corporation. He designed and implemented the Oracle JDeveloper's IDE framework. He has extensive experience in developing IDEs. In addition to Oracle JDeveloper, he has worked on two other IDEs: UIMX: a C++ IDE for developing Motif-based applications, and Visual Age for Basic: a Visual Basic compatible IDE. Currently he is an observer of JSR 227, A Standard Data Binding and Data Access Facility for J2EE.

Ed Burns has worked on a variety of Java Platform, Standard Edition (Java SE) and Java Platform, Enterprise Edition (Java EE) projects in roles ranging from individual contributor to team leader to architect. He co-authored a book, JavaServer Faces: The Complete Reference, which will be available in August 2006.

Ed got involved with the JCP program when he became co-Spec Lead of JSR 127, JavaServer Faces, in October 2002, at the beginning of the JSF development life cycle, and he continued in that role with JSR 252, JavaServer Faces 1.2.

Stefan Hepper works for the IBM development lab in Böblingen, Germany, and he is the lead architect for the WebSphere Portal, Workplace Client and Server programming model, and public APIs. He co-led the Java Portlet Specification V 1.0 (JSR 168) and is now leading the V 2.0 (JSR 286). Stefan also started the Pluto project at Apache that provides the reference implementation of JSR 168.

Join me in congratulating the runners-up and expect more exciting projects to come from them in the months ahead. If you missed my column last month you can check it out at http://java.sys-con.com/read/232104.htm, and meet the winners of the 4th JCP Program Annual Awards.

**Onno Kluyt** is the chairperson of the JCP Program Management Office, Sun Microsystems.

# Eclipse Data Visualization

*(No Silly Glasses Required)*

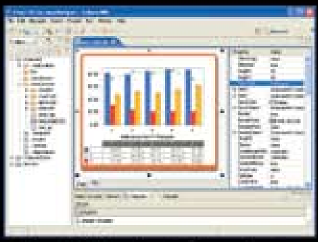


*Chart FX for Java Eclipse plug-in.*

## The Leading Charting Solution Now Provides Powerful Data Visualization for Eclipse

The **Chart FX for Java 6.2 Eclipse plug-in** brings enterprise-level data visualization features to the Eclipse IDE. The Designer is integrated into the IDE allowing quick customization of the charts and the required code generation. In addition to a myriad of traditional chart types, the **Chart FX Maps** extension is included to create dynamic, data-driven image maps, such as geographic maps, seating charts or network diagrams, among others. Chart FX for Java 6.2 is available as a Server-side Bean that runs on most popular Java Application Servers. The 100% Java component produces charts in PNG, JPEG, SVG and FLASH formats. The **Chart FX Resource Center** integrates into the Eclipse Help and includes a Programmer's Guide, the Javadoc API and hundreds of samples. This makes Chart FX for Java the most feature-rich, easy-to-use charting tool available for Java development. *Learn more about the seamless integration and powerful features at www.softwarefx.com.*

Chart FX

www.softwarefx.com

New! version 6.2

Now Includes Maps!

**If your delivery cycle is taking priority over REM cycles,
Jtest's Automated Error Prevention brings your schedule back to reality.**

**Sleeping in your cubicle is no way to stay on schedule.** You need to test while you work, not after everything is done. With Jtest, you'll be home in time for dinner. And a little rest.

Jtest automates the valuable yet tedious task of code reviews and unit testing, allowing development teams to adopt a "test as you go" strategy. This promotes the testing of code as it's developed, so that **quality is built into the code from its inception.** And bugs are eliminated before they infect the rest of the code base.

The result is cleaner, safer, more reliable and consistent code. You'll be reworking less code, which means faster and more predictable release cycles. And more sleep.

- Jtest checks whether code complies with 500+ Java development rules and automatically corrects many reported violations.
- Automatically unit-tests your Java code, evaluating code behavior and exposing unexpected exceptions, boundary condition errors and memory leaks.
- Generates extendable, high-coverage test cases in JUnit format, which can be quickly turned into libraries of reusable test assets.
- Improves productivity by helping team members leverage one another's test settings and related files.

PARASOFT®
*We make software work.*™

**Download your FREE TRIAL at www.parasoft.com/JDJmagazine Or call (888) 305-0041, x3501**