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

# JDJ

JULY 2004 VOL:9 ISSUE:7

Announcing JDJ EDITORS' CHOICE AWARDS '04

## ECLIPSE IS ABOUT MORE LANGUAGES AND MORE PLATFORMS THAN JUST JAVA

EXCLUSIVE INTERVIEW: MIKE MILINKOVICH EXECUTIVE DIRECTOR ECLIPSE FOUNDATION

Plus...

- Is Open Source 'Disruptive Technology'?
- Java: State of the Union
- Medical Image Conversion

RETAILERS PLEASE DISPLAY
UNTIL SEPTEMBER 30, 2004

$9.99US $9.99CAN

08>

0 74470 01751 6

Jeremy Geelan

# Is Open Source a 'Disruptive Technology'?

Without it, there would be no software. Without it, there would be no business. What is it – innovation, of course. What about the recent coinage of *disruptive* innovation – is that too an essential ingredient of modern business and technology?

The term *disruptive* has taken on a life all its own, with XML, for example, being called by one of its exponents in 2000 a "disruptive technology."

RSS-disseminated Web logs are the latest thing to be referred to as a disruptive technology. Disruptive technologies, if we ignore the attempt by *Forbes* magazine to sensationalize them by referring to them as "stealth attacks," are to Harvard Business School professor Clayton Christensen, who first came up with the notion, nothing more than "simple, convenient-to-use innovations that initially are used only by the unsophisticated customers at the low end of markets."

It was Christensen too who came up with the concept of "The Innovator's Dilemma," which he introduced to the world in his bestseller of that same name. In his book Christensen stated: "This is one of the innovator's dilemmas: Blindly following the maxim that good managers should keep close to their customers can sometimes be a fatal mistake."

In other words, while industry leaders typically pour resources into their core business and listen to their customers, it is precisely in doing so that they most risk getting blindsided by disruptive innovations – new products, services, or business models that initially target small, seemingly unprofitable customer segments, but eventually evolve to take over the marketplace.

"Disruptive innovation" is a concept that naturally has now spawned the parallel notion of the disruptive innovator. Sun's John Fowler, when promoted recently to EVP for Network Systems, was enthusiastically cited in the Sun Microsystems official announcement as having a "history of driving disruptive innovation" – probably the first time I can remember seeing a major executive being singled out for promotion specifically on account of being a disruptor!

Christensen's original 1997 insight dawned on him when he was studying the evolution of the disk drive industry. After a disk manufacturer explained to him why they did what they did, he realized that trying to please the customer wasn't always the best way ahead, since the customer's ability to actually use new waves of improvement wasn't certain. By constantly trying to sell to your current customers and to keep improving and improving, a company can inadvertently leave a chink through which some "worse" product can sneak to market and maybe get a toehold.

There is a time for listening and a time for merely "getting on with it." IT departments don't necessarily believe any longer that "utility computing" or "on-demand computing" or "grid computing" equals better. Vertical concerns like security, scalability, reliability, or cost are, in the present economic cycle, more assured of corporate adoption than horizontal concerns.

Is open source software a "worse" product, sneaking in through the chink left by the proprietary software vendors? The "chink" part is undeniable; why else would the headlines over the past few weeks have resonated with one big vendor after another espousing open source, from Computer Associates ("CA Open Sources the Heart of Its Ingres Database") and Sun ("Sun Open Sources Solaris…Again!") to even Microsoft ("Windows Installer XML Toolset Released as Open Source").

Paying lip service to open source won't be enough. The proprietary vendors need to understand free and open source software, in a way that Microsoft's Steve Ballmer didn't when, back in 2002, he said: "The Linux client hardly runs any applications except a bunch of shareware stuff that's not very good. There has yet to be any innovation, new features, new capabilities out of the Linux platform.… First they cloned Unix, and there are people working on cloning some of our stuff. But it's just a cloning OS. I don't think anyone should expect anything innovative coming out of that world."

The difficulty with predicting the future, it's always said, is that it hasn't happened yet.

JAVA DEVELOPER'S JOURNAL

Editorial

| | |
|---|---|
| Editor-in-Chief: | Joseph Ottinger |
| Editor-at-Large: | Alan Williamson |
| Executive Editor: | Nancy Valentine |
| Java Enterprise Editor: | Kirk Pepperdine |
| Desktop Java Editor: | Joe Winchester |
| Core and Internals Editors: | Karl Avedal<br>Carlvin Austin |
| Contributing Editor: | Glen Cordrey |
| Contributing Editor: | Jason Bell |
| Founding Editor: | Sean Rhody |

Production

| | |
|---|---|
| Production Consultant: | Jim Morgan |
| Associate Art Director: | Tami Beatty |
| Associate Editors: | Jamie Matusow<br>Gail Schultz<br>Jennifer Van Winckel |
| Assistant Editor: | Torrey Gaver |
| Online Editor: | Lin Goetz |
| Research Editor: | Bahadir Karuv, PhD |

Writers in This Issue

Karl Avedal, Christophe Conraets, Jean-Marie Dautelle, Bill Dudney, Jeremy Geelan, Dave Glende, Pramod Jain, Yayati Kasralikar, Murali Kaundinya, Onno Kluyt, Margaret Leber, Larry McCay, Srinivasan Muralidharan, Joseph Ottinger, Kirk Pepperdine, Heman Robinson, Volker Simonis, Joe Winchester

To submit a proposal for an article, go to
http://grids.sys-con.com/proposal

Subscriptions

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

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: Java Developer's Journal, SYS-CON Publications, Inc., 135 Chestnut Ridge Road, Montvale, NJ 07645.

Worldwide Newsstand Distribution
Curtis Circulation Company, New Milford, NJ
For List Rental Information:
Kevin Collopy: 845 731-2684, kevin.collopy@edithroman.com
Frank Cipolla: 845 731-3832, frank.cipolla@epostdirect.com

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

Don't take chances connecting your application to your data. Rely on DataDirect Technologies™ for premium JDBC drivers with support for advanced functionality like distributed transactions, connection pooling, and BLOB/CLOB. Our Type 4 drivers are fully database independent and are the SPECjAppServer/ECPerf performance and scalability leader.

DataDirect™
TECHNOLOGIES

**Download your free evaluation today.** www.datadirect.com/jdj

www .datadirect.com 800-876-3101

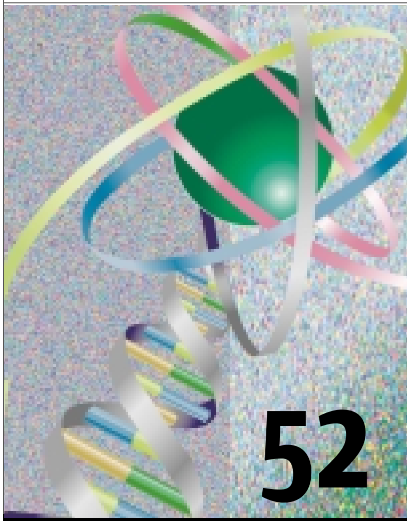
JULY 2004 VOLUME:9 ISSUE:7

# JDJ contents

## Cover Story: *JDJ* Exclusive

JDJ Exclusive
Q & A with Mike Milinkovich,
Executive Director,
Eclipse Foundation

## Features



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.

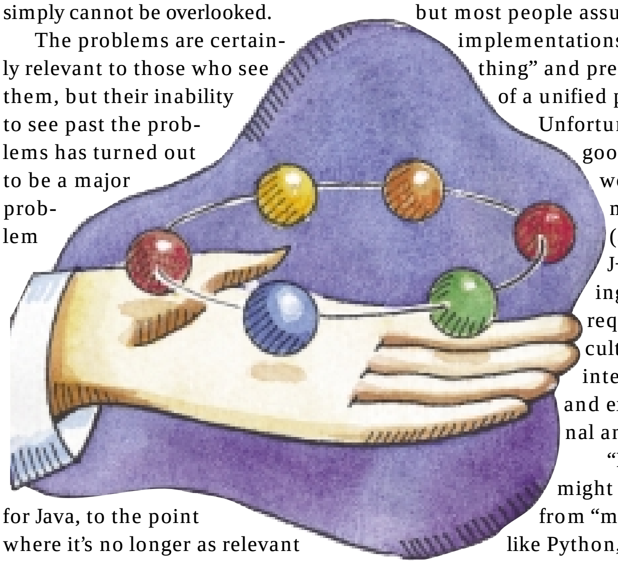
**Joe Ottinger**
Editor-in-Chief

# State of the **Union**

JavaOne is here at last, and while it's not really my place or choice to issue a "State of the Union" for Java, I think we need to do some straight talking.

Sun has, for what it's worth, given us a language and platform. It has restrictions and limitations based on a long-range view of requirements. Originally the requirements were set by Sun Microsystems alone; they then created the JCP to allow the user and vendor community to offer input on what the future requirements would be and how they would be met. What I fear is that some of the recipients of Sun's largesse have decided, in a fit of pique, to reject Java as a gift horse that's simply missing too many teeth. Many people have decided that Java is almost acceptable but "almost" doesn't cut it, because there's *this* blemish or *that* problem that simply cannot be overlooked.

The problems are certainly relevant to those who see them, but their inability to see past the problems has turned out to be a major problem for Java, to the point where it's no longer as relevant as it was.



As one example of Java's problems: we have a splintering of user interfaces in such a way that it renders the JVM less important than the libraries being used. The state of things now is that most users accept this, and some excuse it, saying that the platform you run the JVM on is now more important (again) than the fact that you're supposed to be using Java as the platform. The problem here is so bad that some expect authors to willfully abandon the Java platform in favor of Linux or Windows, which begs the question of why they're using Java in the first place, in my opinion.

That's not to say that Java isn't a powerful language apart from the concept of the Java platform. It's certainly capable, and I suppose I'm happy that people want to use the language while not really caring about the platform. However, I don't think there's a legitimate separation there, especially when you consider Java's history and remember J++.

As another example: we see choirs of cacophony (O, sing!) that are begging Sun to release Java from its proprietary licensing. The licenses offered as replacements don't allow Sun to preserve any real meaning of what Java is, but most people assume that variant implementations will "do the right thing" and preserve the concept of a unified platform. Unfortunately, relying on good will doesn't work when real money is involved (again, remember J++?), and complying with this kind of request is very difficult – unless Sun's intent is to gut Java and expose it to external and internal attack.

"From where?" you might ask. Well, apart from "minor" languages like Python, Ruby, and Perl, we have the possibility of multiple virtual machines running libraries that don't provide the services that the platform requires; we have a 400-pound gorilla competing with Java on its own terms, first with J++ and now with .NET, promising the same cross-platform strengths of Java.

–continued on page 70

**Joseph Ottinger** is a consultant with Fusion Alliance (www.fusionalliance.com) and is a frequent contributor to open source projects in a number of capacities. Joe is also the acting chairman of the *JDJ* Editorial Advisory Board.

*josephottinger@sys-con.com*

SYS-CON MEDIA

President and CEO:
Fuat Kircaali *fuat@sys-con.com*
Vice President, Business Development:
Grisha Davida *grisha@sys-con.com*
Group Publisher:
Jeremy Geelan *jeremy@sys-con.com*

**Advertising**
Senior Vice President, Sales and Marketing:
Carmen Gonzalez *carmen@sys-con.com*
Vice President, Sales and Marketing:
Miles Silverman *miles@sys-con.com*
Advertising Sales Director:
Robyn Forma *robyn@sys-con.com*
Director, Sales and Marketing:
Megan Ring *megan@sys-con.com*
Associate Sales Managers:
Kristin Kuhnle *kristin@sys-con.com*
Beth Jones *beth@sys-con.com*

**Editorial**
Executive Editor:
Nancy Valentine *nancy@sys-con.com*
Associate Editors:
Jamie Matusow *jamie@sys-con.com*
Gail Schultz *gail@sys-con.com*
Jennifer Van Winckel *jennifer@sys-con.com*
Assistant Editor:
Torrey Gaver *torrey@sys-con.com*
Online Editor:
Lin Goetz *lin@sys-con.com*

**Production**
Production Consultant:
Jim Morgan *jim@sys-con.com*
Lead Designer:
Tami Beatty *tami@sys-con.com*
Art Director:
Alex Botero *alex@sys-con.com*
Associate Art Directors:
Abraham Addo *Abraham@sys-con.com*
Louis F. Cuffari *louis@sys-con.com*
Richard Silverberg *richards@sys-con.com*

**Web Services**
Vice President, Information Systems:
Robert Diamond *robert@sys-con.com*
Web Designers:
Stephen Kilmurray *stephen@sys-con.com*
Matthew Pollotta *matthew@sys-con.com*

**Accounting**
Financial Analyst:
Joan LaRose *joan@sys-con.com*
Accounts Payable:
Betty White *betty@sys-con.com*

**SYS-CON Events**
President, SYS-CON Events:
Grisha Davida *grisha@sys-con.com*
Conference Manager:
Lin Goetz *lin@sys-con.com*

**Customer Relations**
Circulation Service Coordinators:
Shelia Dickerson *shelia@sys-con.com*
Edna Earle Russell *edna@sys-con.com*
Linda Lipton *linda@sys-con.com*
JDJ Store Manager:
Brunilda Staropoli *bruni@sys-con.com*

**Borland® JBuilder.®** The #1 Java™ tool in the world for a reason. Pick the size that fits your needs. Automate the routine stuff. Handcraft the unique. Have an active voice at every stage in the process. Move faster, and make every project a hit. Whether your application is headed to the Web, enterprise, or mobile, just pick the feature set you need. And start rockin'!

**Customizable code editor • Refactoring • Local and remote debugging • Integrated unit testing • Two-way visual Struts designer • JSP™ tag library/ framework support • XML and Web Services • Mobile application development • Advanced build and configuration management with Apache® Ant • Visual EJB™ designer • Two-way deployment descriptor editor • Archive builder • Integration with all major J2EE™ application servers**

**go.borland.com/j6**

Borland®
Excellence Endures™

**Learn from the experts at the premier event for technical education – the 2004 Borland Conference – http://connect.borland.com/borcon04s**

**Kirk Pepperdine**
Java Enterprise Editor

# Déja` Vu

Has it really been a year since the last JavaOne? It seems like yesterday that I was watching James Gosling launch T-shirts into the audience using a trebuchet! What a year it has been. Who would have thought that Microsoft and Sun would come to the agreement they did? Those of us watching the JBoss saga were relieved when an agreement halted overt hostilities on that front. With all of this behind us, we could focus on the next challenge in enterprise computing: Microsoft's best shot at infiltrating enterprise computing, the .NET platform.

The future seems full of opportunities. All the vendors are looking at an expanding market for their products. Given the number of companies that have invested in the technology that we've all come to be passionate about, why should we worry about the new platform that has recently come into being? Though people are sniffing at it, no one is using it for anything serious. Does it really matter that the technology that we depend on is owned and sponsored by a single company? After all, we do have IBM, the perennial anchor in the IT industry, on board. They have invested a lot of resources and have focused a lot in this direction. Surely they are doing this because they see this as the future of enterprising computing. More important, businesses feel secure in the knowledge that IBM is a major player in the market even if the competition is a giant. That platform is new and unproven. From a purely technical perspective, the technology looks inferior in many respects and certainly scalability and performance would appear to be an issue. On the other hand, we have a history of successes, a proven track record that is irrefutable in enterprise computing. To further illustrate this point, look at the number of companies that are offering products and services in addition to IBM. Certainly this competition has to be a good sign, a good indication or validation of the strength of the market. But still, there are some doubts, some nagging points that have raised questions.

> "If manpower is the major consumer of resources required to build and deploy a piece of software, it's also the place where small gains in productivity can offer the largest return on investment"

The first and most worrisome source of our doubts has to be the seeming trouble that the guardian of the technology has had in remaining profitable. This has led to a number of people in the industry wondering out loud about the long-term viability of the company. Sure the company is cash rich, but how long can it continue to suffer these losses before some drastic action will need to be taken? In a not-so-unrelated issue, our guardian has managed to corral a fairly substantial brain trust and, now, that trust is starting to disperse. Okay, companies do lose key people over time and healthy organizations are able to continue to function. Why should we worry? ParcPlace will survive these setbacks because after all, Java will never amount to anything.

The problem is, ParcPlace (and yes, I do mean ParcPlace) didn't survive these "minor" setbacks and with it, the superior technology, Smalltalk, has again been relegated to obscurity. Although, there are not so many insignificant differences between Sun and ParcPlace as there are startling similarities. Sadly, the Smalltalk community never recognized the threat that Java posed. IBM quickly hoisted the anchor and moved to Java. Though IBM's revenues still continued to grow well after the industry proclaimed Smalltalk to be a has-been technology, its growth was more like that of rocket ship that has blown up just after take-off. Even as this was happening, the feeling in the community was that both languages could coexist.

The reality is that Smalltalk, Java, and the C#/.NET platforms are all very different from each other. Each requires a special set of skills and it takes a special person to be able to work effectively in more than one environment. Consequently, businesses are forced to make a choice between them. More often than not that choice comes down to cost. How much will it cost to deliver business-critical functionality in each of these technologies? As technologists, we understand that the accounting of these costs is less than perfect, but in the end they do work. Once more, the largest portion of the costs is acquiring the manpower required to develop enterprise systems. If manpower is the major consumer of resources required to build and deploy a piece of software, it's also the place where small gains in productivity can offer the largest return on investment.

One of the differences between Sun and ParcPlace is that Sun clearly understands that to really be successful, it needs to help developers be more productive. The realization of this understanding has resulted in Project Rave, Java Studio Creator – a positive step forward. I look forward to seeing more announcements at JavaOne that outline how Sun plans to secure its future.

**Kirk Pepperdine** is the chief technical officer at Java Performance Tuning.com and has been focused on object technologies and performance tuning for the last 15 years. Kirk is a co-author of *Ant Developer's Handbook* (Sams).

*kirk@javaperformancetuning.com*

# Edge Computing with Java Edge Side Includes

## An end-to-end architectural illustration

by Murali Kaundinya &
Srinivasan Muralidharan

Content Delivery Networks (CDNs) have been used commercially to cache static content across a distributed network. Edge Side Includes is a W3C-acknowledged submission that supports the fragment assembly model. It provides the semantics on how to assemble dynamic content at the edge, when to cache content, when to expire content, etc. Java Edge Side Includes (JESI) provides extensions to Java that make it easy to program JSPs through a custom JSP tag library that will generate ESI markup. In this article we'll describe how to develop applications using JESI and show its impact on the end-to-end architecture.

A typical Web application may be browsed more often for its nontransactional static content than for its business transactions that require a trip to its back-end systems. In such a scenario it's cost-effective to serve some of the content on multiple low-cost edge servers and preserve the back-end compute and network resources for business operations that have transactional requirements.

Standard caching mechanisms, be it on the edge or near the enterprise, are designed to operate with minimal dependency on the back-end enterprise application. The caching entity caches a whole HTTP object and refreshes it based on its cache life properties specified through HTTP headers. This works well for "static" content that can be treated as indivisible objects. There is no mechanism, excluding a few proprietary ones, for applying caching to dynamic content. Edge Side Includes (ESI) was born out of the need for filling this void.

Edge Side Includes is a simple markup language that developers can use to identify content fragments for dynamic assembly at the network edge. ESI defines three main elements:

- A markup language for delimiting content into fragments for deferred assembly
- An extension of the traditional Web/caching architecture that assembles fragmented content
- A way to invalidate the content at the edge

The ability to assemble dynamic pages from individual page fragments means that only noncacheable or expired fragments need to be fetched from the origin Web site, decreasing the need to retrieve complete pages and alleviating the load on the Web site's content-generation infrastructure.

Figure 1 shows a comparison of ESI before and after. Without ESI, client browsers from the Internet issue requests to access content from a particular site hosted in a data center. These requests funnel into the many Web and application servers. Every request that comes from client browsers gets serviced by both the Web and application servers and eventually database servers. With ESI, it's possible to offload the burden from the Web server and application server to "Edge Servers" deployed close to the consumers. The requests from client browsers get serviced by these edge servers, which serve cached content a lot faster. Every now and then, when the cached content expires, it results in a trip to the origin server to update the ESI fragment. With such architecture, the burden on the "Content Generation" infrastructure is drastically reduced.

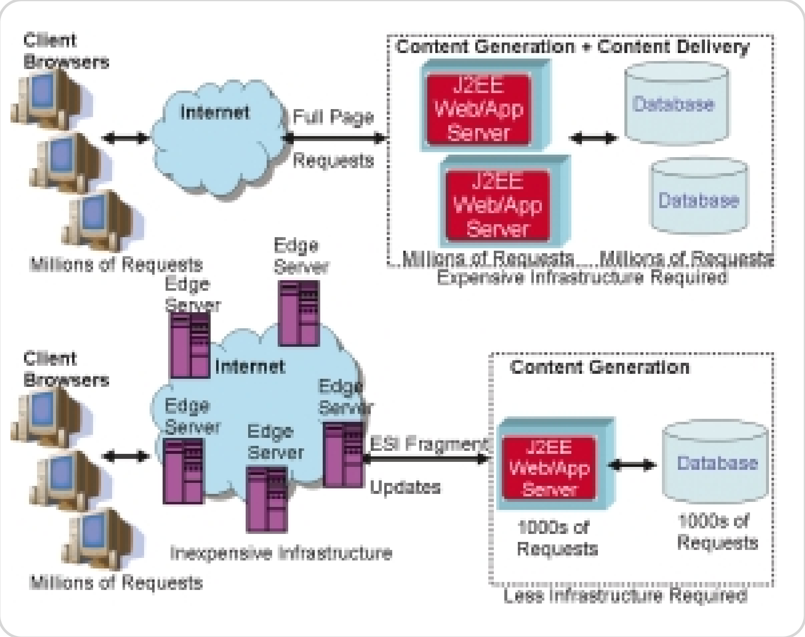


**Figure 1** Before and after edge side includes

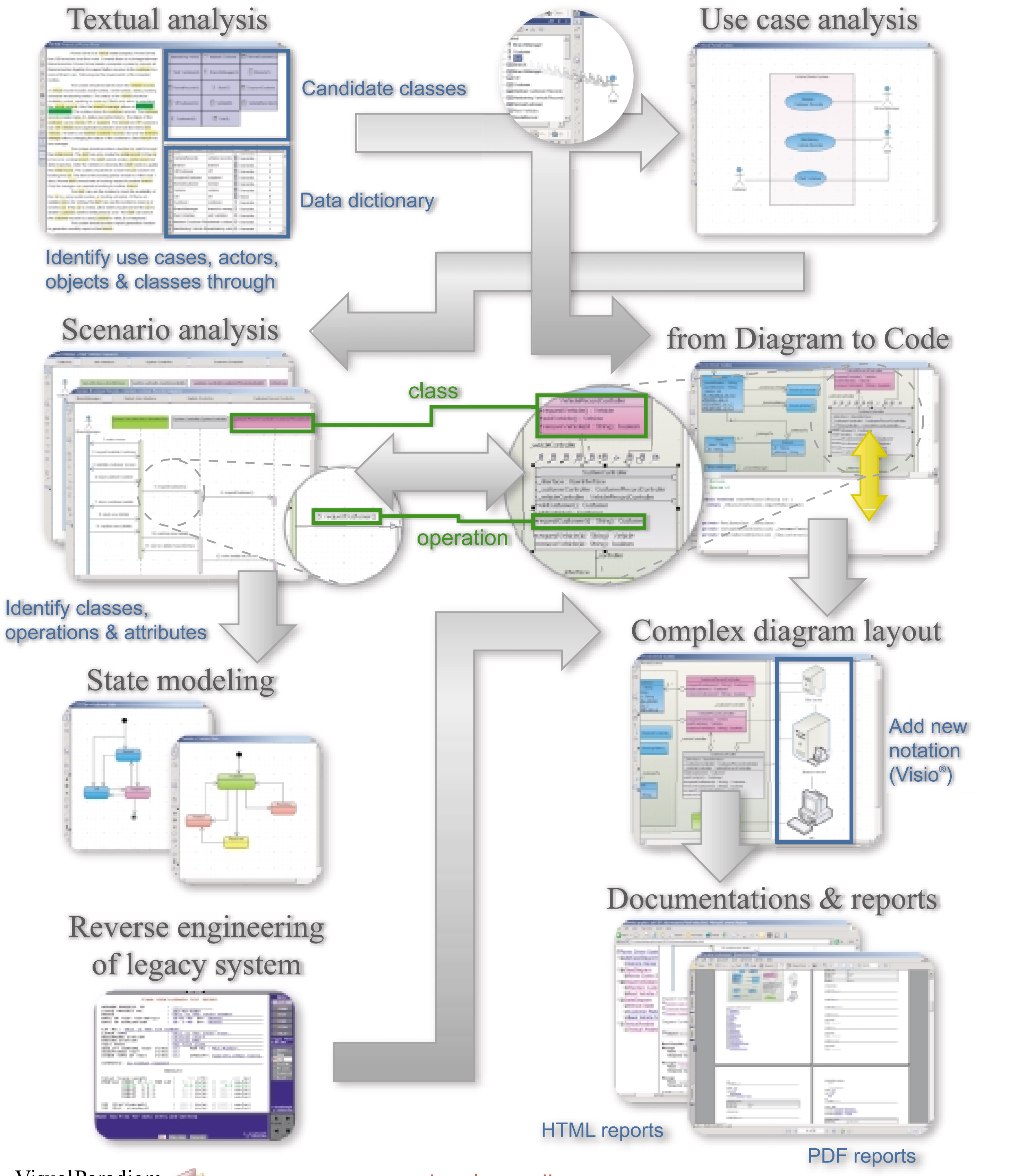
**Murali Kaundinya** is a senior architect with Sun Software Services. He leads enterprise architecture projects for Sun's customers and delivers solutions using Sun Java technologies. He is a member of IEEE and ACM.

*murali.kaundinya@sun.com*

**Srinivasan Muralidharan** is an advisory engineer at IBM in RTP, North Carolina. He works on host integration, emerging technologies.

*muralisr@us.ibm.com*

# Visual Paradigm for UML
# - from Diagram to Code



Textual analysis

Candidate classes

Data dictionary

Identify use cases, actors, objects & classes through

Use case analysis

Scenario analysis

from Diagram to Code

class

operation

Identify classes, operations & attributes

State modeling

Complex diagram layout

Add new notation (Visio®)

Reverse engineering of legacy system

Documentations & reports

HTML reports

PDF reports

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

## Java Edge Side Includes

ESI is an XML-style specification for subdividing a page into component parts with individual cache-control properties. An HTML page from the enterprise (business) server could contain ESI tags to be processed by an Edge component termed ESI processor en route to the client. The ESI processor, as part of the content delivery network, could be embedded in a Web server or application server and use the ESI markup to control caching characteristics of page requests. Java Edge Side Includes (JESI) is the specification of ESI for Java and is a natural fit for JSP-based Web development.

## JESI Architecture

JESI Architecture involves the configuration of a JESI tag library to the Web container, as shown in Figure 2. The JSP programmer develops JSP pages using JESI tags. These JESI tags get processed and result in HTML pages with embedded ESI tags. An ESI processor sitting between the client and the server, typically on the CDN, processes these ESI tags that indicate which of the page fragments need to be served from the ESI Processor's cache and which need to be fetched from the origin server. If the page is not in the ESI processor's cache, it sends a request to the origin server for the page. It stores it in its cache and sends an aggregated response to the user. If the page is in cache, the ESI processor evaluates the cache properties of the components of the page. This determines the list of cache components in the page that need to be refreshed. This may necessitate sending a request to the source to fetch the expired components. Finally, it processes the response, caches the refreshed component, assembles the page from the components, and sends it to the client.

Though JESI resembles "JSP include", the two have entirely different operational characteristics. A JESI include generates an ESI positional marker for ESI processors to construct the page. Construction of a page will involve hits to only those JESI include pages that have expired or are not in cache. Further, each JESI include page is fetched by a separate HTTP request. Hence, a JSP page should not assume that the HTTP request object passed to an include page will be the same as that passed to the containing JSP page. This should be bourne, especially when migrating an old Web page to JESI technology.

## JESI Tags

While Web designers can directly embed pages with ESI tags in JSP, JESI offers a better alternative for several reasons:

- It's natural to express new markup using custom tags.
- JESI is convenient to use as it internally generates verbose ESI markup.
- ESI configuration options and deployment parameters can be naturally specified in application configuration files to be picked up by JESI.
- Installation and maintenance is through the familiar tag library management process.

JESI comes as a JAR file containing the tag library and an associated Tag Library Definition (.tld) file. The application server can be enabled to use JESI merely by configuring the system with these files.

Before describing JESI tags, we must delineate two models of usage of JESI tags. There's a template/fragment model and a control/include model. It's important that JESI be flexible enough to be easily used with old Web pages and at the same time provide richer control for new Web page development. It's difficult to convert old Web pages into component Web pages to be "included" into the main composite Web page. The template/fragment model is used to convert old applications to JESI with relatively little effort. In this model a page is broken up into cacheable fragments, each fragment with its own cache properties, and the page outside of the fragments is marked up with the "template" tag. The "control/include" model is recommended for creating new Web pages. The Web page's cache control properties are provided under the "control" tag and other pages are included with the "include" tag. As with fragments, an included page has its own cache properties.

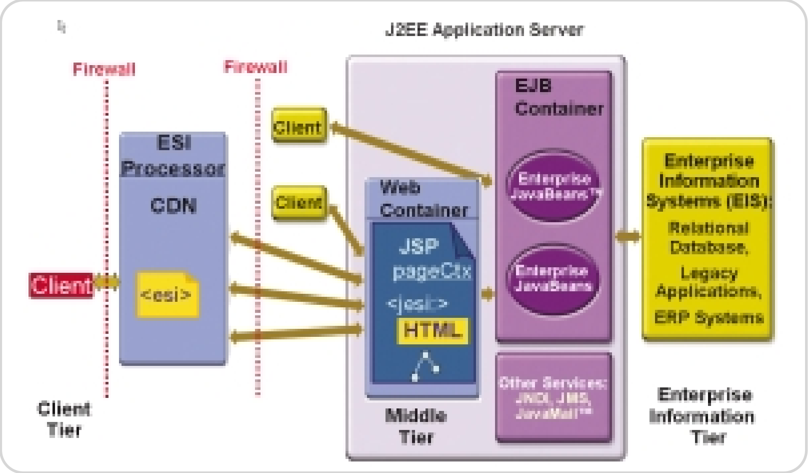


**Figure 2** JESI architecture

Listing 1 illustrates the syntax of some of the JESI tags. The JESI specification in the reference section provides comprehensive coverage of all the tags. <jesi:control> sets the caching characteristics of a page composed of "include" pages. It should be placed to surround the cacheable content of the page. The properties of the tag affect the content within the control tag but outside of the "include" tags. The properties of the tag in a page do not apply to the included pages. <jesi:include> is used to include cacheable resources identified by a URL. The JESI include tag, which is similar to a JSP include tag, is different in that JESI include tags are fetched by the ESI processor with separate HTTP requests.

The consequences of this provide a key to when it is appropriate to replace a JSP include with a JESI include. The include tag can be used in a template/fragment model as well. <jesi:template> tag, like the control tag, affects the HTTP headers only and does not generate HTML tags. The template tag is intended to be used with JESI fragments and controls cache characteristics of content in the page outside of fragments but within the template tag. In other words, the content outside the fragments but within the template tag is one cacheable object.

The main intent of the template tag is for the conversion of existing pages to the JESI model. In this model, the ESI processor's defaults will not apply to the page in the absence of a template tag. <jesi:fragment> tag is a

Status Report
Phone:
Fax:
E-mail:
Initial
Phone
Email
Status
Interview

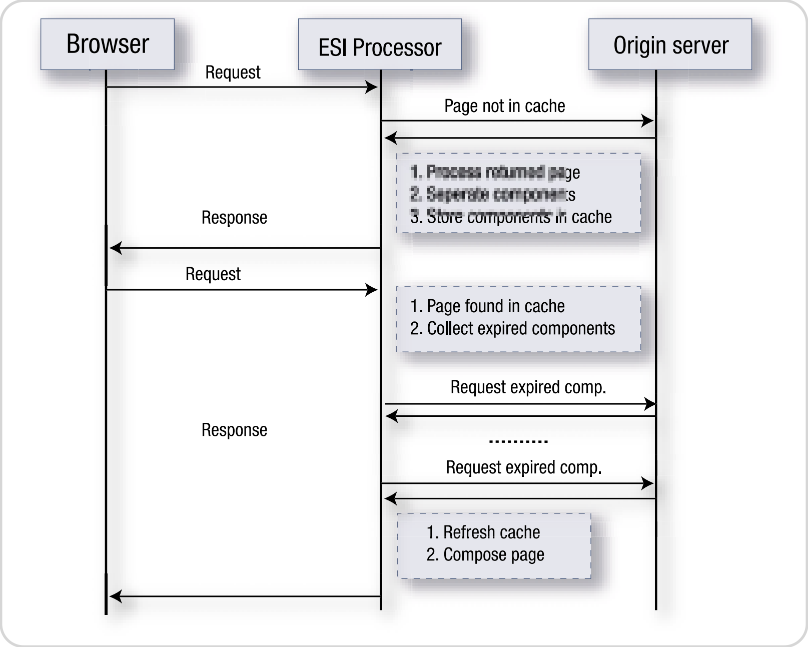


**Figure 3** JESI and a sequence diagram

cacheable object to be used in the template/fragment model to divide pages into individual cacheable components. Fragments have the same cache properties as template or control tags. Just like the template tag, a fragment has its own cache control and cannot inherit properties from the template or from the ESI processors. <jesi:codeblock> tag is used to conditionally execute code outside of fragments in the template/fragment model. The conditions specify if the code is to be executed when a template or a fragment is fetched by the ESI processor. <jesi:invalidate> allows the JSP programmer to invalidate cache objects. If a prefix is specified, the directive applies to all objects with the specified prefix. The username/password attributes provide an authentication mechanism for logging into the cache server to execute the directive. Invalidate uses ESI's invalidation protocol, which is executed over HTTP. The optional "cookie" and "header" information can be used to determine what to invalidate. The <jesi:personalization> tag allows the ESI processor to set cookie values in responses, thus saving a trip to the origin Web server. If the cookie was found in the request header and has a non-null value, that value is used. If the cookie is not found in the header or the value is null, the value specified in the tag is used.

The source JSP processor processes the refresh requests from the ESI processor as follows (see Figure 3).

### *Request for a Template*

The template content outside of the fragments is processed. Content in the codeblock tag whose execute property specifies "template" or "all" is processed. The JSP processor assembles the resulting content, inserts placeholder fragment tags at the specified location in the content, and sends the page to the ESI processor.

### *Request for a Fragment*

First the JSP processor processes the template content outside of the fragment to initiate any side effects of the processing used in the fragment. All codeblock tags preceding the requested fragment whose execute property specifies "fragment" or "all" are processed followed by the fragment itself. The resultant page contains the processed fragment and placeholder tags for fragments not processed. The template content is discarded.

### *Request for an Include Page*

This is a request to the origin Web/application server that generates the included page that follows the usual Web page service path.

## JESI-ESI Example

Consider a "home" page of a company. It consists of a basic template that defines the presentation of most of the Web pages in the Web site. There is both static content, such as the company logo, banners, footers, etc., and dynamic content. There is an external business news section that changes several times a day, an overview section that changes once a week, a product outline section that changes once every 10 days, and a map to the rest of the Web site, which rarely changes. Content like business news that changes often is expensive to obtain and is well suited to be cached at the edge server. It's best to be modeled as a JESI fragment as part of a JESI template.

Also assume that a blood donation campaign is in progress and that the organizers want to display a bar chart showing their goal regarding the number of donors and the number of people signed up so far as part of all corporate pages. This information is stored in a special database table and is updated four times a day. The chart is a good candidate to be an additional JESI fragment. Therefore, you would add a JESI template tag at the top of the page and use JESI fragment tags to enclose the fragments that are to be cached as separate entities.

Assume that the URL to the corporate page is www.acme.com/employee_page.jsp. Figure 4 shows the JSP code snippet and the ESI-embedded response beginning with the header.

The ESI processor is alerted by the Surrogate-Control response header. Note the no-store directive, generated because of the cache="no" setting in the JESI template tag. The ESI processor makes two additional requests, where it fetches and caches the two fragments. After that, the composite page is returned to the employee. When the employee works with the page again, the dynamic content will be newly generated but the chart and the footer will be served from the cache.

Without JESI and ESI, you can at best cache static content on a surrogate (reverse proxy) and force the enterprise server to generate the dynamic content

```
// Source Code at Server
<%@ taglib uri="http://localhost/jesitaglibrary.tld"
prefix="jesi" %>
<jesi:template cache="no" >
<p>BEGIN</p>
... some dynamic page content...
<jesi:fragment>
This_is_the_body_of_Blood_Donation_Campaign
</jesi:fragment>
... some more dynamic content...
<jesi:fragment>
This_is_the_body_of_Business_Footer
</jesi:fragment>
<p>END</p>
</jesi:template>

// Response Header
Surrogate-Control: content="ESI/1.0", max-age=86400+0, no-store

// Response body:
<p>BEGIN</p>
... some dynamic page content...
<esi:include src="/employee_page.jsp?__esi_fragment=1"/>
... some more dynamic content...
<esi:include src="/employee_page.jsp?__esi_fragment=2"/>
<p>END</p>
```

**Figure 4** JESI – ESI transformation

Jtest®

# Automate unit test case generation for JUnit and Java with Parasoft Jtest®

## The Problem with Most Team-Wide Practice Implementations

Most development teams recognize that if they want to improve software quality, their entire team should implement practices such as coding standards enforcement and unit testing. However, many teams don't recognize that implementing these practices requires more than simply purchasing the tools that facilitate the practices and then installing those tools on every developer's workstation.

Although it would be nice if tools alone could drive a team to implement a practice, it is not possible. Tools don't operate in a vacuum. Truly implementing a practice in a team requires more than just tools. It also requires the team culture, workflow, and supporting infrastructure required to embed the practice into the team's development process.

Teams that attempt to implement practices with tools alone typically do not achieve the expected quality improvement benefits. For instance, assume that a team tries to implement the coding standards enforcement practice by only purchasing a coding standards enforcement tool and asking each developer to use that tool. Over time, it's likely that most of the coding standard violations will remain in the code. Why? Without additional team-wide support for the coding standards enforcement practice, developers typically become overwhelmed by the number of problems reported and do not know how to approach them. The tool helps the team members recognize the faults in their code, but if the developers do not have the necessary support, the faults remain and the code quality does not significantly improve.

The Parasoft AEP Methodology details one strategy for embedding best practices into a team's development process. To learn how this methodology works, visit **http://www.parasoft.com.**

*– **Adam Kolawa,** Ph.D.*
*Chairman/CEO of Parasoft*

## Introducing Parasoft® Jtest®

### Parasoft Jtest is the first and only automated unit testing product for Java development.

With just a click, Jtest reads, analyzes and tests code — creating harnesses, stubs and inputs automatically. Jtest also enables you to automate regression testing and coding standard analysis. For loyal JUnit users, Jtest is designed to fully support existing test cases and automate the creation of new JUnit-style test cases.

### Learn how Jtest can enhance JUnit capabilities...

Download a free eval copy of Jtest along with our informative new white paper entitled "Automated Java Unit Testing, Coding Standard Compliance, and Team-Wide Error Prevention."

For Downloads go to www.parasoft.com/jtest

**Email: jtest@parasoft.com or call: 888-305-0041 x3303**

PARASOFT®
*We make software work.*™

*Founded in 1987, Parasoft's clients include IBM, HP, DaimlerChrysler and over 10,000 companies worldwide.*

**FEATURES**

- Generates and supports JUnit test cases
- Checks over 400 rules; corrects violations of over 200 rules
- Manages team-wide sharing of test settings
- Integrates with IBM® WebSphere® Studio Application Developer and Eclipse
- Supports Borland® JBuilder®, Together® Control Center®, and Ant

**BENEFITS**

- Makes error prevention feasible and painless, which brings tremendous quality improvements, cost savings, and productivity increases
- Makes unit testing and coding standard compliance feasible and painless
- Encourages the team to collaborate on error prevention

**PLATFORMS**

Windows 2000/XP
Linux
Solaris

**Contact Info:**

Parasoft Corporation
101 E. Huntington Dr., 2nd Flr.,
Monrovia, CA 91016
**www.parasoft.com**

for each request without consideration of the dynamicity of the various parts. The only alternative would be to let the application be aware of the cache life of the various parts and cache internally, though this is hardly a recommended or practiced option as it makes both development and maintenance complex and error prone.

Using JESI and ESI, however, the page could naturally be broken down into component parts based on the life of each part and let the ESI processor orchestrate composition of the page to be served up.

## Conclusion

Web content can be classified into static content, transactional non-cacheable content, and content that has both. Efforts to reduce round-trip latency have been attempted from the advent of HTTP 1.1. These efforts have been to cache content either at the browser or at the various hopping points of the content's path. Caching should be transparent to the developer and if at all exposed should be at coarse granularity. JESI provides for fine-grain control and this can result in additional complexity if not architected carefully. Caching and JESI introduces an additional, orthogonal characteristic for page composition, namely the cache life of the components. Too small a cache time will defeat the purpose of ESI caching and too large a cache time has the danger of stale data (with more impact as some of the related content may have different cache characteristics). Web pages with dynamic content that necessitate fetching dynamic data from a back end may offset the benefits of caching.

In summary, JESI will be effective when effort is spent on optimally designing Web pages with careful consideration to the cache-life of the components. This will require careful thought as to content regeneration policies.

## References

- *JESI – JSP Tag Library for Edge Side Includes (ESI):* http://jcp.org/en/jsr/detail?id=128
- *IBM's WebSphere APIs for Edge Side Includes:* www.alphaworks.ibm.com/tech/wesi
- *JESI Tags for Edge Side Includes:* http://otn.oracle.com/docs/tech/java/oc4j/jsp1131/jesitags.htm
- *Akamai ESI:* www.akamai.com/devnet/esi.html

*Listing 1*

The <jesi:control> tag

Syntax:

```
<jesi:control
      [expiration="value"]
      [maxRemovalDelay="value"]
      [cache= "yes" | "no" | "no-remote" ]
      [control="control_str"]
      [useRelativeURLs= "true" | "false" ]
      [scheme="scheme_str" ]
      [serverName="server_name_str"]
      [serverPort="server_port_str"]
      [servletPath="servlet_path_str"]
      [pathInfo="path_info_str"]
/>
```

The <jesi:include> tag

Syntax:

```
<jesi:include
    page= "url_str" | "<%= expression %>"
    [flush="true"|"false"]
    [copyparam="true"|"false"]
    [alt="alt_page_str"]
    [ignoreError="true"|"false"]
        />
```

Or

```
<jesi:include
    page="url_str" | "<%=expression %>"
    [flush="true"|"false"]
    [copyparam="true"|"false"]
    [alt="alt_page_str"]
    [ignoreError="true"|"false"]
        >
    ( <jesi:param
         name="paramterName"
         value="{ parameterValue | <%= expression %> }"
         /> )+
</jesi:include>
```

The <jesi:template> tag

Syntax:

```
<jesi:template
      [expiration="value"]
      [maxRemovalDelay="value"]
      [cache={"yes", "no", "no-remote"}]
      [control="control_str"]
      [useRelativeURLs="true" | "false" ]
      [scheme="scheme_str"]
      [serverName="server_name_str"]
      [serverPort="server_port_str"]
      [servletPath="servlet_path_str"]
      [pathInfo="path_info_str"]
      [executeEntirePage="true" | "false"]
         >
```

The <jesi:fragment> tag

Syntax:

```
<jesi:fragment
      [expiration="value"]
      [maxRemovalDelay="value"]
      [cache="yes" | "no" | "no-remote"]
      [control="control_str"]
         >
```

The <jesi:codeblock> tag

Syntax:

```
<jesi:codeblock execute="template" | "fragment" | "always" >
```

The <jesi:invalidate> tag

Syntax:

```
<jesi:invalidate
        [url="url" username="value"password="value"]
       <jesi:object uri="..."/>
  </jesi:invalidate>
```

Or

```
<jesi:invalidate
        [url="url" username="value" password="value"]
       <jesi:object uri="..." [prefix="yes"]
[maxRemovalDelay="..."]>
           [<jesi:cookie name="..." value="..."/>]
           [<jesi:header name="..." value="..."/>]
       </object>
  </jesi:invalidate>
```

The <jesi:personalization> tag

Syntax:

```
<jesi:personalize name="a_name_str" value="a_value_str" />
```

# if(extremeProgramming.equals(scientificMethod))

## Programming as theory building

by Larry McCay

### article.getIntroduction().toString();

Comparing software development to manufacturing has been ingrained in the industry for years. This has led to such concepts as software factories and the commoditization that is currently sending much of the industry offshore.

The Scientific Method is a better analogy for the development of complex and evolving software systems in that it leads to a methodology that evolves software systems in such a way to make them more general and true over time.

This article describes how the Scientific Method supports the Extreme Programming methodology practices prescribed by Kent Beck in *eXtreme Programming Explained*.

The intent of this article is to illustrate the ways in which Extreme Programming is supported by the proven practices and guidelines of the Scientific Method, not to be a comprehensive guide or introduction to either methodology.

### article.getMotivation().toString();

Having recently read Neal Stephenson's novel, *Quicksilver*, I've enjoyed a renewed interest in all things scientific. While reading an Isaac Newton biography, it dawned on me that the Scientific Method used by the scientific community in its search for knowledge, and taught to us as early as grammar school, is indeed analogous to the goals and very tenets of Extreme Programming (see Figure 1).

After a bit of googling, it becomes obvious that I am not the first to draw these conclusions. Rick Mugridge wrote a terrific paper that describes the commonalities between the Scientific Method and Test Driven Development (TDD), which happens to be a practice of XP.

In addition, I have never been comfortable with the connotations associated with the unfortunate name of Extreme Programming. It makes a hard sell even more so in some circumstances. Especially since the dot-com bust, management is leery of anything considered remotely extreme – let alone named "extreme."

### extremeProgramming.getHistory().toString();

In the beginning there was Smalltalk and the Smalltalk culture or way. This culture included such things as refactoring, development in pairs, rapid change, constant customer feedback and integration, iterative development, and constant testing as key elements. This culture and technology date back to the mid-1980s at a time when Kent Beck and Ward Cunningham worked for Tektronix.

In addressing what he saw as the basic problem with software development – risk – Kent Beck developed Extreme Programming. Extreme only in the way that it applies common-sense principles and practices to the development of software. The fact that programming is part of the name is due to the code-centric nature of the methodology. Communication, documentation, and training are facilitated through the code – especially through the test cases used to drive the development process.

Being a lightweight software development methodology geared for small-to-medium-sized teams, XP's strength is in dealing with vague and/or rapidly changing requirements.

XP builds on best practices such as unit testing, pair programming, and refactoring.

The basic principles of XP are:

1. Communication
2. Simplicity
3. Feedback
4. Courage

Extreme Programming consists of the following steps:

1. Choose a story.
2. Write tests.
3. Run tests.
4. Refine, program, and refactor – repeat #3 as needed.
5. Go to #1 until all stories are complete.

### scientificMethod.getHistory().toString();

The Scientific Method was first introduced by Francis Bacon (1561–1626); however, it was not used as a strict discipline until Isaac Newton later in the 17th century.

The goal of the Scientific Method is to provide a set of steps to ensure the development of provable theories that may lead to new and/or greater understandings of the workings of nature and its systems. These theories are gradually stepped up in generality until the highest level, at which point there may be opportunity to unify the theories.

In his *First Book of Aphorisms*, Bacon asserted that:

> *There are and can be only two ways of searching into and discovering truth. The one flies from the senses and particulars to the most general axioms, and from these principles, the truth of which it takes for settled and immovable, proceeds to judgment and to the discovery of middle axioms. And this way is now in fashion. The other derives axioms from the senses and particulars, rising by a gradual and unbroken ascent, so that it arrives at the most general axioms last of all. This is the true way, but as yet untried.*

Both ways set out from the senses and particulars, and rest in the highest generalities, but the difference between them is infinite. For the one just glances at experiment and particulars in passing, and the other dwells duly and orderly among them. The one, again, begins at once by establishing certain abstract and useless generalities; the other rises by gradual steps to that which is prior and better known in the order of nature.

To accomplish this, a methodology would need to remove any bias of the scientist from the experiment that leads to the proving or disproving of a given hypothesis or theory.

The Scientific Method consists of the following steps:

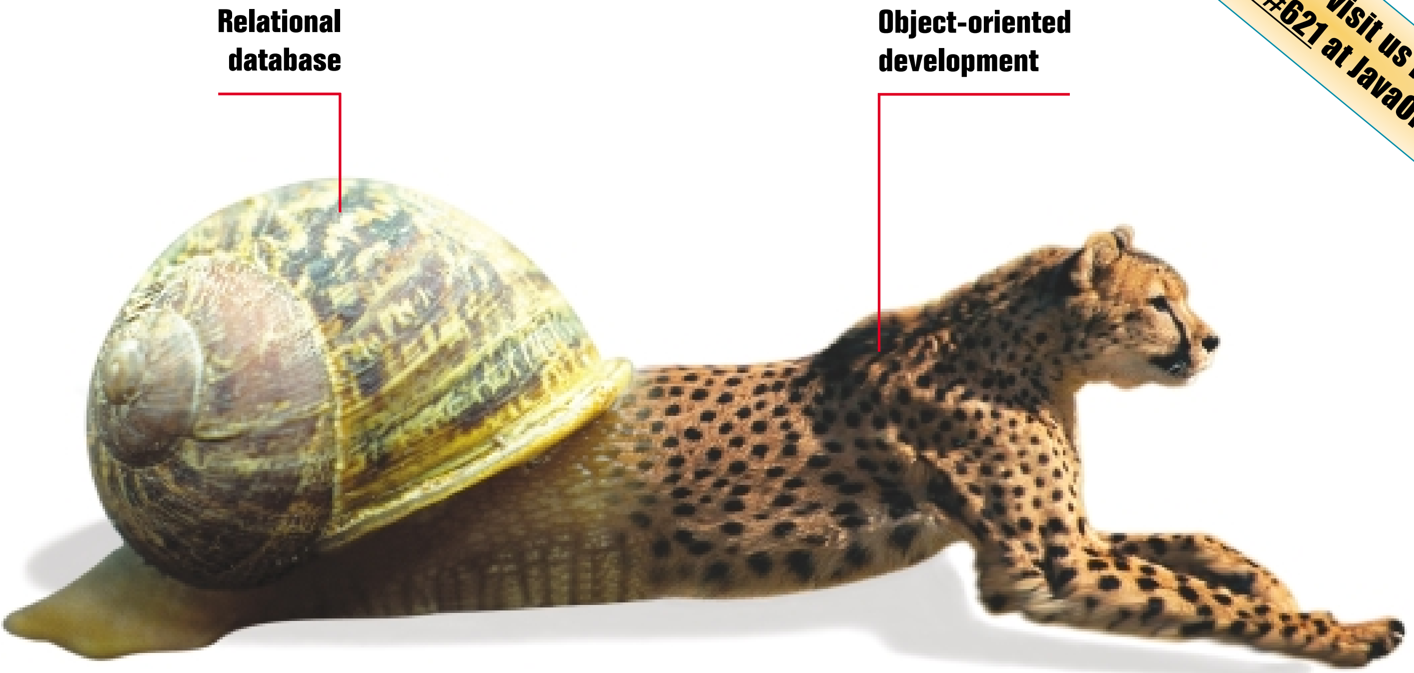
1. Make observations.
2. Create hypotheses.
3. Make predictions.
4. Conduct experiments.
5. Modify hypotheses if predictions are not met and go to step 3.
6. Declare hypothesis as theory.

Newton, inspired by Descartes, believed in mathematics as the truest mechanism of proof. Through experimentation, Newton tested and proved all of his greatest scientific discoveries. Some of these experiments were used

**Larry McCay** is a senior software engineer with 15 years of software development experience. He works for Probaris Technologies in Philadelphia developing an enterprise class secure electronic-forms routing engine and server.

*lmccay@probaris.com*

Come visit us in Booth #621 at JavaOne



# DON'T LET YOUR DBMS SLOW YOU DOWN

If your back-end database isn't a good match for your front-end development, you need a new database.

Caché is the *post-relational* database that combines high-performance SQL for faster queries and an advanced object database for rapidly storing and accessing objects. With Caché, no mapping is required between object and relational views of data. That means huge savings in both development and processing time.

Applications built on Caché are massively scalable and lightning-fast. Plus, they require minimal or no database administration.

More than just a database system, Caché incorporates a powerful Web application development environment that dramatically reduces the time to build and modify applications.

The reliability of Caché is proven every day in "life-or-death" applications at thousands of the world's largest hospitals. Caché is so reliable, it's the leading database in healthcare – and it powers enterprise applications in financial services, government and many other sectors.

We are InterSystems, a specialist in data management technology for over twenty-six years. We provide 24x7 support to four million users in 88 countries. Caché is available for Windows, OpenVMS, Linux and major UNIX platforms – and it is deployed on systems ranging from two to over 10,000 simultaneous users.

InterSystems CACHÉ™
Make Applications Faster

## Try a better database. For Free.

Download a free, fully-functional, non-expiring version of Caché or request it on CD at www.InterSystems.com/match1 to prove the theories developed by earlier great thinkers such as Galileo.

## programming.equals(theoryBuilding);

The Scientific Method is concerned with the generalization, unification, and development of Theories to express the workings of the universe and its components. Much of what comprises the Scientific Method is related to supporting this goal.

Peter Naur published a paper in 1985 entitled "Programming as Theory Building." His contention is that software development tools or methodologies cannot solve the inherent problems of software systems development alone. Naur prescribes the notion of programming as building a theory for the solution of the problem being solved by the project. This provides a greater understanding of the source code and system architecture and leads to a longer life for the system.

According to Peter Naur, programming itself may be viewed as an act of Theory Building. In viewing software development as Theory Building, a development team is focused on building, understanding, and communicating the theory of the solution to a given problem. One of the artifacts of the development and maintenance of such a theory is a computer program that is easily extended and modified by a group of developers that understand the original development team's theory of the solution.

Extreme Programming builds on Naur's concept through the use of the Metaphor in communicating and the continuous development of the architectural aspects of the program.

The common goal of a communicable theory across software development and the Scientific Method provides a jumping off point for discussing how the Scientific Method supports the extreme programming method of delivering software projects.

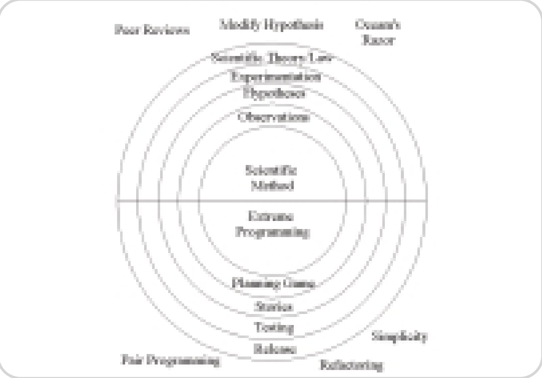


**Figure 1** Scientific Method vs Extreme Programming

Just as any new introduction or change in scientific theory must retest and prove previous theory validations, in Extreme Programming enhancements must not break any of the existing unit or functional tests. In this way, the tests provide a mechanism to not only prove the integrity of the solution but aid in communicating the original and ongoing theory of the solution. Previously valid tests or experiments that no longer hold true in light of new developments may signify a need for change or dismissal of the original theory or solution.

## pairProgramming.equals(peerReview);

The practice of Pair Programming within XP is a form of real-time code review. Its intentions are to facilitate the communication of important design decisions, catch violations of the chosen metaphor, or point out simpler designs and possible areas to refactor.

The Scientific Method must ensure the quality and integrity of the resulting data for public use – this is done through Peer Review. Peer reviews consist of a critical review by technical experts who don't have a vested interest in a particular investigation. The purpose of a peer review is to confirm that the research has been conducted in a scientifically sound manner.

## story.equals(observation);

The development of user stories within XP is crucial in the collection of requirements and the understanding of the problem domain. The onsite customer and the clear communication and feedback between him or her and the development team is arguably the most important tool for the development team. Using this tool the developers are able to make the most accurate observations of the problem at hand, much the same way that scientific research requires accurate tools to collect observations that drive the development of the hypothesis to be proved.

Unit and functional tests are developed to determine the state of the delivery at any point in time. These tests are continuously run so as to catch any change in the integrity of the system. Once all of the tests are run successfully, the system is complete as described by the user's stories. This complete set of stories and tests document the theory of the system.

Using the scientific method, predictions are made based upon the observations that support the given hypothesis. These predictions are verified through unbiased experimentation. Based upon the results of the experimentation, predictions and/or tests are changed until the hypothesis is proven or disproven. If proven, the hypothesis may go on to be declared a Scientific Theory.

## simplicity.equals(occamsRazor);

One of the core principles of XP is that of simplicity. "Which is the simplest thing that could possibly work?" The simpler the implementation and design, the easier it will be to test and change. This leads to designs and implementations that are more flexible, understood, and longer lived.

Occam's Razor is a principle proposed by the fifteenth century philosopher, William of Ockham, that leads to the practice of choosing the simpler of two theories that explain the same phenomena. This does not necessarily mean that the simpler of the two is more likely correct, rather that the simpler of the two is more easily tested.

## article.getConclusion().toString();

To arrive at the best solution or theory for a given problem, scientists and software developers alike must adhere to certain practices that support the continued testing, evolution, and discarding of previous works.

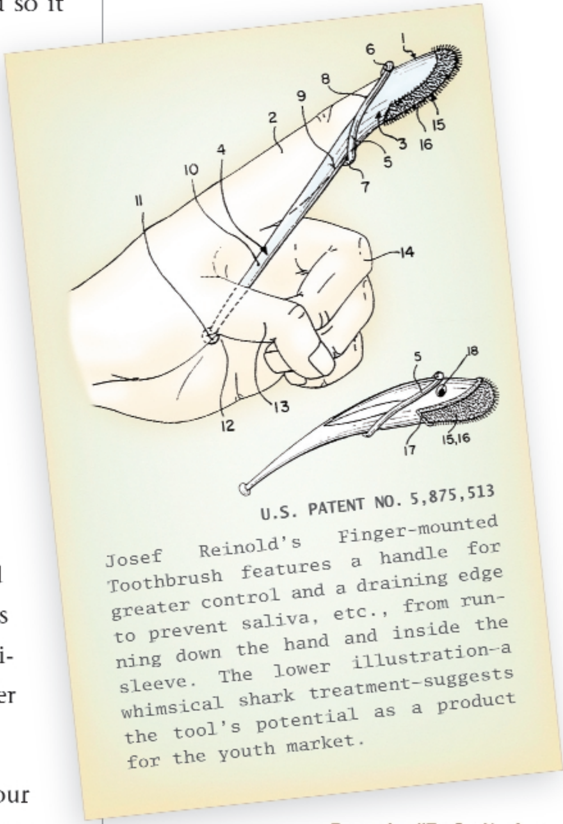
The Scientific Method and Extreme Programming methodologies are practices that support these concepts.

Extreme Programming supports the notion of programming as theory building, which emphasizes the importance of the knowledge, communication, and understanding of the original development team.

But really, it's just a Theory.

## article.getReferences().toString();

- Beck, K. (2000). *eXtreme Programming Explained*. Addison Wesley.
- White, M. (1997). *Isaac Newton: The Last Sorcerer*. Perseus Books.
- Mugridge, R. "Test-Driven Development and the Scientific Method." Agile Development Conference, June, 2003.
- Fowler, M. (1999). *Refactoring*. Addison Wesley.
- Naur, P. (1985). "Programming as Theory Building." Microprocessing and Microprogramming, 15: 253–261.
- Stephenson, N. (2003). *Quicksilver*. Harper Collins.
- *Portland Pattern Repository WikiWikiWeb. XpRoots*: http://c2.com/cgi/wiki?XpRoots

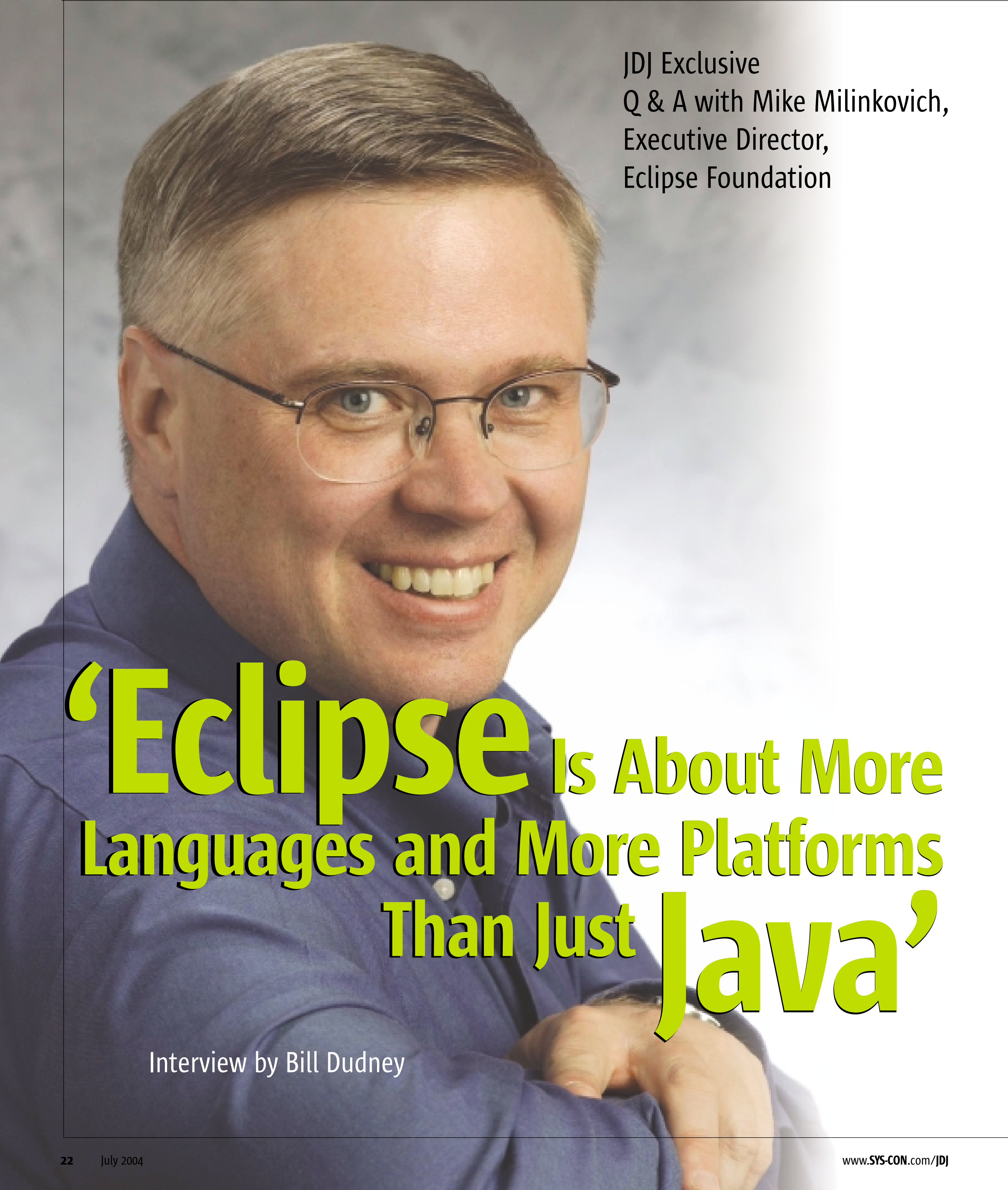


U.S. PATENT NO. 5,875,513

Josef Reinold's Finger-mounted Toothbrush features a handle for greater control and a draining edge to prevent saliva, etc., from running down the hand and inside the sleeve. The lower illustration–a whimsical shark treatment–suggests the tool's potential as a product for the youth market.

From the "Tools You're Not Likely Ever To Use" archive.

JDJ Exclusive
Q & A with Mike Milinkovich,
Executive Director,
Eclipse Foundation



# 'Eclipse Is About More Languages and More Platforms Than Just Java'

Interview by Bill Dudney

*On June 1, 2004, the Eclipse Board of Directors named Mike Milinkovich the executive director of the Eclipse Foundation. The hunt had been on since Eclipse became an independent entity earlier this year. Here* JDJ*'s Eclipse editor, Bill Dudney, talks exclusively with Milinkovich about the direction he wants to take Eclipse and what his vision is for the community.*

Mike Milinkovich comes to the Eclipse Foundation from Oracle, where he served as the vice president of OracleAS Technical Services. Prior to Oracle he was at WebGain as the vice president of World Wide Services, Training and Support Services. Milinkovich joined WebGain as a result of their acquisition of TopLink from ObjectPeople, where he was vice president of business development. Prior to ObjectPeople he was with Object Technology International (OTI) – the company that IBM acquired and that eventually built Eclipse, so this move is sort of a homecoming for Mike.

The interview generally covers four areas. First is SWT and its impact on the Java market. Second is Eclipse as an open source community. Next, the WebTools project, and finally I asked Mike about JSR 198, the JSR to define a common plugin API so plugins can work across IDEs.

**JDJ:** Does SWT bifurcate the Java market?
**Mike Milinkovich:** I know there is a perception in the marketplace that Eclipse is primarily a Java IDE. But in many ways Eclipse is and always has been about more than just Java – more languages and more platforms.

Part of the rationale behind supporting SWT and now the Rich Client Platform (RCP) was not to bifurcate the Java market, because our focus was on building tools that were of interest to people who perhaps were not interested in Java at all. We're interested in tools for other languages like what we're doing with the CDT for C and C++, and for other platforms and for other things like modeling and tools for the embedded marketplace. These areas are quite different from what you typically think of when you think of Java tools. So from that perspective, I think SWT is here to stay.

However, I don't want this characterized as a Sun Java versus Eclipse scenario because I frankly just don't see it that way. We are an open source project; we are trying to build a tools integration platform that can be used universally for many languages and many platforms. We are happy with what we are getting from SWT.

**JDJ:** SWT runs fast on Windows, but on other platforms (Mac OS X, Linux, etc.) the performance is not as good. What plans do you have to get the performance on other platforms more in line with the Windows platform?
**MM:** I'm too new in the job to bring a huge amount of perspective on how we got to where we are and what the processes are that are achieving the results we are getting. I think as a general comment, though, that we are looking for more involvement from the Linux people, be it Novell or Red Hat, and from other places like Apple to help us improve the tool set on those platforms. I think there's probably a lot of performance tuning that we could do but the best way to go about that is to get deeper involvement from the people who own those projects.

**JDJ:**That leads me back to the bifurcation of the marketplace: Apple and, I believe, Red Hat as well have invested heavily in making Swing run well on their platforms – now you are asking them to reinvest in SWT.
**MM:** It's actually news to me that Apple and/or Linux did anything on their platforms to optimize for Swing. I think they may have helped the Swing people improve performance on those platforms. I'm not aware of anything that, for example, Apple would have done to specifically improve performance of their platform to make Swing run faster. It might be something that I'm just not aware of.

**JDJ:** Doesn't the presence of the RCP bifurcate the market because of the requirement to build the user interface with SWT? For example, tools such as the Visual Editor (an Eclipse subproject) are having significant work poured into them to make the SWT aware. Doesn't the RCP/SWT combination bifurcate the market, even if that bifurcation is unintended?
**MM:** No, it doesn't. I'd like to point out that one of the key features of the 3.0 platform is SWT/Swing interoperability, and it was the investment on our side that made that happen.

**JDJ:** What about the rest of the platforms that do not support this SWT/AWT bridge?
**MM:** Sometimes it takes more than one release to solve all problems.

**JDJ:** Any other comment on SWT?
**MM:** I just want to reiterate what I said a little earlier. I believe that SWT is here to stay. I don't believe that statement should be characterized as trying in any way to split the Java community. We are an open source project doing what we need to do for our project to be successful.

**JDJ:** Moving on then, can you shed some light on what the status of the WebTools project is? At EclipseCon there was a lot of excitement about this project, but since then it seems to have died down.
**MM:** I'm too new in the job to give any history since EclipseCon. The fact that we are finally getting this project off the ground is very good news. That is certainly an area I intend to put some energy into, to make sure it gets off to a good start.

**JDJ:** Any kick-off date in mind? When might we see some of the output?
**MM:** No I don't know yet about the output part. I believe the kickoff has actually been posted on the Web site and the project is following the Eclipse development process. We are in a 30-day period of proposal review that ends some time in the next couple of weeks.

[*Author's Note:* The proposal was posted on April 27 and the 30-day period was extended by one week to finish assimilating feedback from the community. Look for the project to become official very soon. The WebTools Project (WTP) awaits a board vote.]

JDJ: We are 2.5 years into Eclipse. How do you feel the model is working for IBM and the other commercial entities that are involved?

**MM:** I think IBM is actually very happy with the payback they have received on the project. Of course I'm not an IBM spokesperson so I can't comment for them, but I think that they view Eclipse and the momentum it has in the marketplace as a very positive thing. I think that's a proof point of the notion that when it comes to open source and commercial participation in open source, the more you put in the more you get out of it.

JDJ: Do you see this as a trend in the industry? For example, BEA recently announced Beehive.

**MM:** I can't speak for other companies but I do believe that there will be more companies doing this. I believe they look at the success of Eclipse as an open source project driving market acceptance and see Eclipse as an exemplar. I would point out also that we are very excited about what BEA is doing with Beehive and we fully intend to be working with them down the road to make sure that the effort they are putting into their project will work well with Eclipse.

JDJ: Do you see Eclipse's dominance of the market as good or bad for innovation?

**MM:** I think it's wonderful for innovation. Eclipse is open source and therefore open platform. I can't see how, by growing the community of people who are contributing to an open source project, Eclipse could possibly be stifling innovation. I think there is a lot of interest in Eclipse; there is a huge amount of effort going into not just adding to the open source platform as it exists today but growing entirely new open source projects.

Just as interesting is building a vibrant commercial ecosystem of people building products on top of Eclipse or building plugins that work with Eclipse. People are making businesses building software in that way.

I think Eclipse is a platform for innovation and in no way stifles innovation. And I think that is part of the original vision for Eclipse. From the beginning, Eclipse was about making a common open source platform that solves problems people have been solving repeatedly in proprietary commercial implementations. By creating an open source and open implementation of the previously commercial, proprietary implementations, it allows people to go and innovate and solve interesting problems instead of building yet another text editor or yet another code browser.

JDJ: In which ways do you see the Eclipse market broadening? What's the most important area for it to broaden into?

**MM:** In my own mind I don't have any one particular area. I would go back to what I said earlier, that Eclipse is and always has been a universal platform for tool integration that goes beyond just solving Java developers' problems.

Where would I like to see Eclipse grow? First of all, we will continue to be very active as a Java IDE and you will see that in Eclipse 3.0. This next version will solve the problems that Java developers want solved. We will continue to grow in that area.

When I look forward into other areas where I want to see the technology grow, embedded systems is certainly one area I'm interested in seeing Eclipse grow. If you look at the board of stewards, three members – QNX Software Systems, MontaVista Software, and Ericsson – are on our board because they are interested in seeing Eclipse grow in the embedded space. The fact that Wind River joined as a member is another example of the embedded space companies joining to see Eclipse grow in that market. So I believe that growing Eclipse as a standard tool set for developers in the embedded market place is an area that we are very interested in.

Other technology areas will include the area of modeling and model-driven architecture. These are areas where Eclipse has got some traction today and I'd like to see it grow quickly in the future.

This list is not in any particular priority. As I mentioned before, the WebTools project is an area where I really see that Eclipse can and should grow to make sure it has a complete tool set for the J2EE developers who are out there. By the way, I'd add that in and of itself this is a pretty impressive list of things that we are already into that we will now be trying to grow.

JDJ: At EclipseCon there was concern that in adding the WebTools project you'd be cutting into some of the margins and feature sets of some of the commercial members of the Eclipse community. Do you have any comment on that?

**MM:**: I've spoken to a number of the add-in providers that are out there. I believe the consensus is (I've not spoken to anyone who disagrees with this) that Eclipse will continue to grow; it needs to continue to grow. We can't say that because a particular add-in provider is in a particular market segment or solving a particular problem that we can never have an open source project moving in that direction.

What there is a strong desire for, and I believe this a very fair request and one that I certainly intend to honor, is that there be visibility and communication about the direction Eclipse is going in. If you read the by-laws and development process there is a huge emphasis on having a very transparent strategy-setting and planning process. There is visibility from the add-in provider community on where Eclipse is going so they can make forewarned and rational business decisions about what they want to do and what they want to focus on.

> "Eclipse is and always has been a universal platform for tool integration that goes beyond just solving Java developers' problems"

JDJ: Do you have any particular areas you want to see Eclipse grow in that we have not already discussed?

**MM:**: It will focus on the areas we have already discussed. The one point I want to make is that right now Eclipse actually has excellent brand awareness and a lot of recognition in the marketplace. But my understanding from the people I've spoken to is that if you go out into the IT shops, the enterprise development shops, there are people out there who don't necessarily know that much about Eclipse. One of the areas we will focus on is developing marketing programs to target those developers. I want to see these developers become more aware of Eclipse and its benefits.

**JDJ:** How do you see the relationship between Eclipse and Sun in the future? Do you have a particular direction you'd like to see this go? Are there currently any ongoing talks?
**MM:** There's nothing in particular in the works right now. I think to a large degree those talks were understandably on a hiatus while the executive director was being recruited and the Eclipse Foundation re-created itself as an independent entity. There are no talks that I'm aware of.

That said, from my perspective the door is always going be open. If there is something Sun wants to talk to us about, I will always respond to their phone calls and have good-faith conversations about what we could be doing together. I personally think the entire conversation about Sun versus Eclipse is kind of missing the point. Sun is doing what they are doing in support of their shareholders and in their role as the steward of the Java community. Eclipse is doing what it is doing in its role as the steward of the Eclipse Open Source community. And to the degree possible, where we have congruent goals, we should be working together and I'd be thrilled to do so.

**JDJ:** Do you see a place for you to go after Sun to get some cooperation between the Eclipse Foundation and Sun?
**MM:** I certainly plan on making a few phone calls and introducing myself. Ultimately it takes two to tango and Sun is going to have to decide whether it's in its business interests to reopen the conversations with Eclipse. That will be their decision to make.

**JDJ:** IBM is on the JSR 198 expert group but there is currently no one from Eclipse. Are there plans for a member of the Eclipse community to join the JSR 198 expert group?
**MM:** Let's bear in mind that, up until the formation of Eclipse as a not-for-profit corporation, under the JCP process there was no way for Eclipse to have a representative on that JSR expert group. I don't know about trying to get someone on that group – it will be something that I will have to look into after I've been on the job for more than a week!

**JDJ:** Thank you Mike for talking with the world's leading i-technology magazine. We'll be closely watching the continuing story of Eclipse as a universal platform for tool integration.
**MM:** We'll always be here to answer whatever concerns and questions the technology community may have.

**Bill Dudney, *JDJ*'s Eclipse editor, is a senior consultant with Object Systems Group. He has been doing Java development since late 1996 after he downloaded his first copy of the JDK. Prior to OSG, Bill worked for InLine Software on the UML bridge that tied UML Models in Rational Rose and later XMI to the InLine suite of tools. Prior to getting hooked on Java he built software on NeXTStep (precursor to Apple's OS X). He has roughly 15 years of distributed software development experience starting at NASA building software to manage the mass properties of the space shuttle. You can read his blog at http://jroller.com/page/BillDudney.**

Karl Avedal
Core and Internals Editor

# J2SE 1.5: Growing the Language – Finally

A major event is about to happen – the final release of version 1.5 of the core Java platform.

The changes in 1.5 are some of the most important to the Java language. This is a big step for Java and it's not an easy one. People with an existing investment in the platform tend to be very conservative about the language and core platform, usually for good reasons: nobody likes to have their investment rely on something that has been deprecated, made obsolete, and possibly becoming unsupported. Dramatic changes to the Java language upset a lot of people, not just anyone but some of the long-time Java users who have played a part in making Java successful. Language growth is both painful and necessary.

In Guy Steele's famous speech and article "Growing a Language" from 1998, he discusses the growth of programming languages. He explains how Lisp's ability to grow in a seamless way made it large and has kept it alive for an amazingly long time, while behemoths like PL/1, which was designed to solve everything out of the box, never succeeded.

Java is looking more and more like the language for solving everything. It was never a language that supported growth in the way Steele meant. It doesn't have constructs, like macros, that enable developers to extend the language syntax. This means that the growth has had to come through official channels, which is why the 1.5 release is so important to Java. In the early days of Java, every new and major upgrade was a revolution. The differences between the alpha API and 1.0 were huge, and the changes in 1.1 and 1.2 were substantial. With 1.2, Java became big, successful, and "real"; people started depending on it for large projects. The constant revolutions had to be replaced by slower evolution.

However, if Java does not evolve fast enough, something without a history to care about will soon turn Java into a legacy technology. This is a pretty normal cycle in the computing world. One technology gets wide acceptance, but eventually grows large and can't evolve as fast, then something new comes along and replaces it. The Java language has felt some of this. Competing languages, such as C#, Ruby, and Groovy, are not tied to previous decisions and existing architectures. They use this to try to leapfrog Java.

For a long time it seemed as if Sun had decided to accept a slower pace of development so it wouldn't upset existing Java stakeholders. Soon people who were not tied to Java for legacy reasons started looking at these competing languages. The only way to stop this is to have the courage to make the necessary changes, even if they hurt. With J2SE 1.5 I think Sun is finally showing that they have this courage. This is also true for the EJB 3.0 specification, which relies on features from 1.5 to drastically change how EJBs are developed.

> "Dramatic changes to the Java language upset a lot of people"

It did take a long time for Sun to make these dramatic language changes and this has helped Java's competitors a lot. Considering that, for example, generics have been in the pipeline since at least 1998 and a JSR since 1999, it's amazing that they haven't become a reality until now. I'm not saying that generics should have been added to 1.4 – they weren't ready then – but it doesn't seem as if Sun has made this a high priority until now. You could speculate that the economic crisis at Sun has played a role in this. When everything was going great for Java and Sun, they might have been afraid to make big changes since that could risk what they had. Now that Sun is in a much tougher position, they might be willing to risk a little in order to gain much more.

Does 1.5 provide what Steele wanted? Steele's point was that the language must facilitate language growth; it must supply patterns for extending the language. Steele suggested a few examples, most notably generics and operator overloading. Java 1.5 does contain generics but not operator overloading. However, it does provide one feature that defines a pattern for language growth: annotations. I would love a complete macro facility for Java, but I know that many wouldn't and annotations do give me some of the power that macros would. This is a great thing that moves Java to a position that will allow it to get older and still evolve. Seeing how annotations will be used to revolutionize EJB development, this is truly about growing the language.

Of course, I don't like all the changes in 1.5. For example, static imports are taking Java one step closer to write-only code. If you want to save keystrokes when using common statics, you should tell the IDE that when you type abs(), the IDE should expand it to Math.abs(). The language should not sacrifice readability for the sake of writability; the language needs to work well for both writing and reading, and enable the developer to be efficient by providing help to the IDEs. Static imports should be an IDE feature, not a language feature.

J2SE 1.5 will obviously be one of the main talking points at JavaOne and there are plenty of sessions available. If you want to learn about the new features, I recommend attending TS-1952, "Fast Track to the Java 2 Platform, Standard Edition (J2SE) v1.5," which will be presented by my co-editor, Calvin Austin, as well as Mark Reinhold.

**Karl Avedal**, a core and internals editor for *JDJ*, is leading a research project for developing an environment for creating and using domain-specific languages. He has been doing Java development since the first official Java beta in 1995 and, among other things, he was a cofounder of Ironflare, the creators of the Orion Application Server. In addition, he's served on several expert groups within the J2EE field and has coauthored two J2EE books.

*karl.avedal@sys-con.com*

# Turbo-Charging Java for Real-Time Applications

## Accelerating code execution

by Jean-Marie Dautelle

*The Java platform is usually perceived as inadequate for real-time applications because of its lack of determinism, that is, its unpredictable execution time.*

*For example, garbage collection (GC), which removes no-longer-needed Java objects and reduces memory overhead, may automatically and transparently freeze the system from time to time. Such behavior is obviously unacceptable in the real-time world. (A commonly recognized goal of real-time computing is to meet an application's time constraints.)*

To address this issue, new Java Virtual Machines (JVM) are being developed (e.g., JVM with concurrent GC). In addition, a new Real-Time Specification for Java (RTSJ, JSR-001) has been finalized.

Unfortunately, these solutions achieve predictability to the detriment of performance. For example, concurrent GC is less efficient than "stop the world GC" (which requires total CPU usage), and the memory model advocated by the RTSJ requires runtime checks that impact performance.

This article examines a new solution, one that provides determinism for real-time threads and also has the positive side effect of significantly "accelerating" code execution.

**Jean-Marie Dautelle** is a principal software engineer at Raytheon in Marlborough, MA, for the air traffic control department. He recently filed two patents for its work on the Raytheon Gazelle high-performance graphic toolkit (lead architect). He holds a master degree in electrical engineering from Orsay University, France, and a post graduate degree in information processing and simulation. He is also the author of the open source JADE library (http://jade.dautelle.com).

*jean-marie@dautelle.com*

### The High Cost of Object Creation

Creating new objects in Java has a significant memory/CPU impact. The impact is somewhat proportional to the object size, but creating even small objects is quite expensive. The memory has to be allocated and initialized and, eventually, when the object is no longer needed, garbage collection is used to free up the memory.

Avoiding memory allocation can significantly increase the performance of your application. (The J.A.D.E. library provides an XML parser significantly faster [2x–3x] than any conventional XML parser only because it does not perform dynamic allocation.)

To minimize object creation and its associated overhead, Java programmers can:

- ***Use primitive types:*** For example, using primitive type "double" is 10 times faster and requires one-third less memory than creating instances of class java.lang.Double.
- ***Use the "return value" parameter technique:*** The basic idea is to avoid object creation by passing a local static object to a function. The function returns this extra parameter after modifying its state to correspond to the desired value. Numerous examples of this technique can be found in the Java standard library (for example, Component.getLocation(Point rv)).

Both of these approaches are error-prone, however. Java primitive types cannot be strongly typed, and the "return value" parameter has to be mutable (modifiable at runtime), which is inherently unsafe (see "Item 13: Favor Immutability," in *Effective Java Programming Language Guide* by Joshua Bloch for a detailed explanation). Also, the "return value" approach unnecessarily increases the number of Java methods because conventional functions "without" the additional parameter are still provided (for example, Component.getLocation()). To Sun's credit, it should be mentioned that the expense is mitigated quite a bit in their latest versions of Java compared to, say, 1.3.x, especially for short-lived, small objects – thanks to the HotSpot Generational Collector. It may be worthwhile to point this out, even if the improvements don't scale to the same speed as object reuse.

### A Real-Time Solution for All Virtual Machines

Garbage collection occurs when memory is being allocated. Therefore, if "new" ready-to-use objects exist (and need not be allocated/initialized because they have been recycled), the memory/CPU is not stressed. As a result, code execution:

1. Is faster
2. Is not interrupted by the garbage collector (thereby providing more predictable scheduling)
3. Has no assignment constraint, as all objects originate from the heap (see RTSJ assignment rules where heap objects cannot refer to scoped objects [JSR-001, pg. 8])

All Java Virtual Machines work in a "heap context" where objects are allocated on demand ("new") and recycled through garbage collection. To support object "recycling" in a transparent manner, we could either use some reference-counting mechanism or work with thread stacks. Due to possible circularities in the general case, the first approach is difficult to implement. The second approach is easier and faster, but the application has to ensure that stack objects are not referenced anymore after the stack is "popped." Fortunately, this risk can be greatly mitigated in practice (using the export method, as we'll see later), which makes this approach far more attractive as a general purpose solution.

# 99.9% of the world won't find these screen shots terribly exciting. But if you're in the other 0.1%, yeehaw.

Use Crystal Reports 10 with your J2EE applications

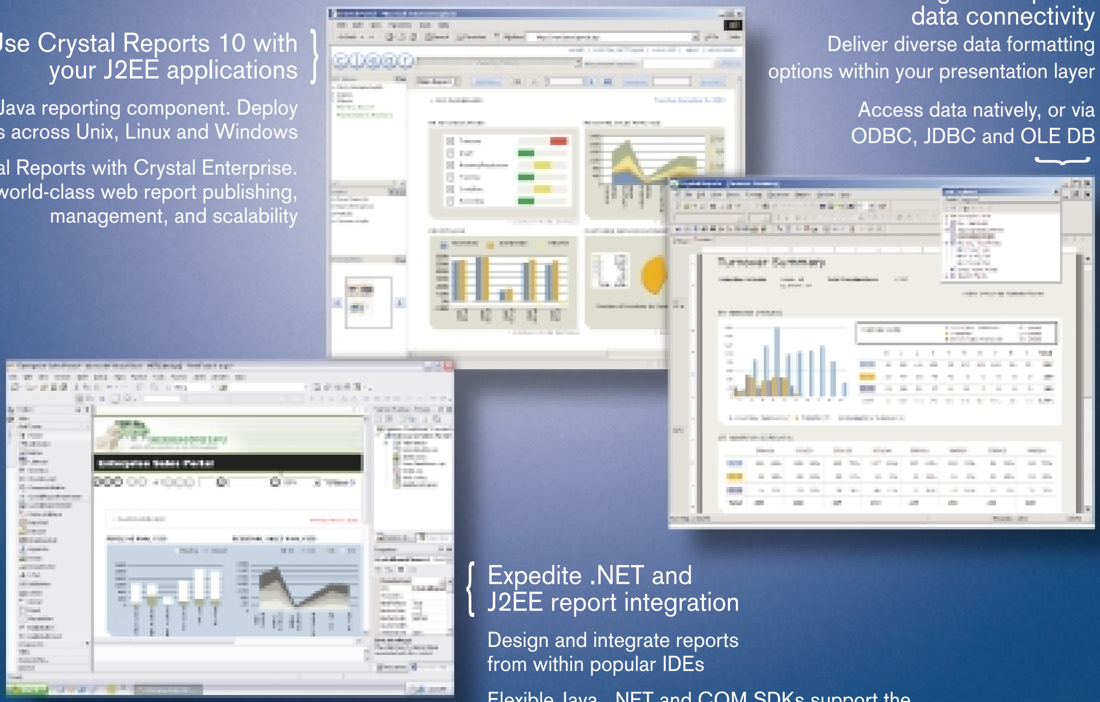
New 100% Java reporting component. Deploy reports across Unix, Linux and Windows

Extend Crystal Reports with Crystal Enterprise. Get world-class web report publishing, management, and scalability

Visual Designer simplifies data connectivity

Deliver diverse data formatting options within your presentation layer

Access data natively, or via ODBC, JDBC and OLE DB



Expedite .NET and J2EE report integration

Design and integrate reports from within popular IDEs

Flexible Java, .NET and COM SDKs support the tight integration of report interactivity including: group tree navigation, exporting, printing, and drill down

## New Crystal Reports 10.

The best in business intelligence now offers the best in business reporting. New Crystal Reports® 10 is a faster and simpler way for developers to integrate dynamic data into applications and implement high-quality viewing, printing, and exporting. Learn more about Crystal Reports 10 and Crystal Enterprise™ 10, and access technical and evaluation resources at www.businessobjects.com/v10/047 or contact us directly at 1-800-877-2340.

BUSINESS OBJECTS®

## Context Programming to the Rescue

Often the same piece of code might have to behave differently based on some thread-locale information. It's not always practical to pass this information as extra parameters to the methods' calls. For example, arithmetic operations might depend on a common modulo number or concurrent threads might log information in separate files. For such situations, the open source J.A.D.E. library defines specific zones called Context, where threads may execute independently from each other (see Java Addition to Default Environment, http://jade.dautelle.com). The scope of a context is defined by a try-finally block statement, which starts with a static enter call and ends with a static exit call, the class name identifying the type of context; for example:

```
LocalContext.enter(); // Context used for local setting.
try
    DEBUG.setValue(true);
    ...
} finally {
    LocalContext.exit();
}
```

Context can be nested; it inherits the setting/behaviors of its outer contexts (unless these setting/behaviors are mutually exclusive). This characteristic also applies to concurrent threads executed while in the context's scope (see Listing 1).

Context programming is somewhat complementary to aspect-oriented programming. Whereas context programming is dynamic by nature (thread based), AOP is typically code based (AspectJ tool/compiler). Both can be used in conjunction to insert custom context code automatically.

## The Pool Context

This context implements the "stack" approach mentioned earlier. It ensures that most of the CPU is used to perform the actual task and not maintenance tasks such as memory allocation and garbage collection. In other words the CPU is used at its maximum efficiency.

Pool contexts allow objects to be recycled so that after the pool/stack of recycled objects gets large enough, no memory allocation need ever be performed.

As far as the application is concerned, pool objects need not be mutable; in fact, it's better (safer) if they are immutable. Remember that within a pool context, creating immutable objects is as efficient as reusing mutable objects.

All objects that have been allocated while in a pool context are recycled at the same time when the thread exits the pool context. Recycling is extremely fast and independent from the number of objects allocated (a lot faster than GC). (Recycling is almost instantaneous; it basically consists of resetting the pool/stack's pointers.)

Listing 2 illustrates how pool contexts can be used to accelerate calculations on multiple inputs.

As you can see in Listing 2, it may be necessary to export important results from the current pool context to the outer context to keep these results from being overwritten after the pool objects are recycled. In most cases, the only object that needs to be exported is the result of the operation; all intermediate/temporary objects can be ignored (they are automatically recycled).

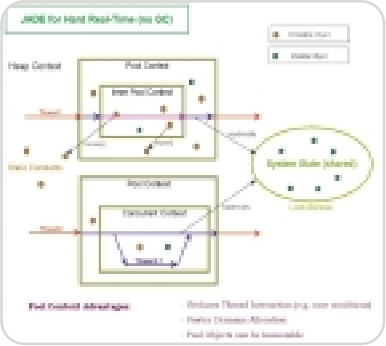


**Figure 1** Threads in pool context

## No Garbage Collection Ever

For some, a real-time application being interrupted by the garbage collector and consequently missing a deadline is simply not acceptable (considered a critical error in hard real time). Fortunately, by using pool contexts it's relatively easy to avoid running the garbage collector.

There will be no garbage collection ever as long as all your threads run in a pool context, only static constants are exported to the heap, and your system state can be updated without allocating new objects (e.g., StringBuffer instead of string or FastMap instead of HashMap) (see Figure 1). (FastMap class, unlike HashMap, does not allocate a new entry each time a new object is added to the collection.)

For concurrent access/modification of the system state, the use of a reentrant lock is recommended, such as com.dautelle.util.ReentrantLock or the new (JDK1.5) java.util.concurrent.locks.ReentrantLock. Provided that factory methods are used instead of the new keyword for object creations, most of the application code is oblivious of the garbage collection issue. (The new keyword always allocates on the heap. The J.A.D.E. library cannot/does not change the virtual machine behavior with regards to class instantiation.) Particular care should be taken with some JDK library methods that may allocate temporary objects onto the heap at each call (setup/initialization heap allocations are okay), and therefore should be avoided or replaced by cleaner classes (e.g., TypeFormat [J.A.D.E. class: com.dautelle.util.TypeFormat] for parsing/formatting of primitive types). Listing 3 provides an example of a real-time handler processing UDP messages

## A Nice Side-Effect: Increase of Execution Speed

The cost of allocating an object on the heap is somewhat proportional to that object's size. The cost of reusing an object, however, is independent of its size. In other words, the larger the object, the more performance gain you can expect from using a pool context. For example, adding 1024-bits immutable integers is up to five times faster (LargeInteger versus BigInteger, J.A.D.E. benchmark results). The high performance associated with pool contexts is due not only to object reuse but also to a more efficient use of the CPU internal cache (cache hits are a lot more frequent when objects are being reused).

Recycling objects is more powerful than just recycling memory (a.k.a. GC). It's particularly true for objects requiring some CPU-intensive setup at initialization (e.g., preallocated linked lists or tables). Unlike hardware recycled objects, software recycled objects are as good as new.

## Limitations

The strength of Java resides mostly in its comprehensive library. Unfortunately, the Java API may allocate temporary objects on the heap, which may annihilate the performance gained from using pool contexts (if you save 100 allocations, that's good…but if the API does 1,000 allocations in the process of running your code, saving 100 allocations isn't as big a gain as might be imagined). One solution is for the JVM to support pool contexts, making the new keyword context-sensitive. This change would be backward compatible, as the default context is the heap context. Then the whole Java API would be more deterministic and execute faster.

## Concurrent Context: Harnessing Hyper-Threading and Multiprocessors Potential

With the JDK1.5 Tiger release, a significant effort has been accomplished with regard to concurrent programming. Still, the

## Formula One e.Spreadsheet Engine

# EXCEL ENABLE YOUR JAVA APPLICATIONS

**API-driven, 100% Pure Java toolset for embedding financial functionality into projects and applications**

### *Automate the building of Excel reports.*

Access raw data in databases, XML files, and other data sources. Perform calculations and formatting. Generate Excel reports with formulas, merged cells, charts, outlining, and other rich formatting upon delivery to users.

### *Use Excel files to govern calculations and business rules in Java server environments.*

Java developers no longer have to translate Excel formulas to Java code. Spreadsheet experts (business analysts, actuaries, financial gurus, etc.) continue to own the core business logic and calculations. Calculation errors and application downtime is greatly reduced.

### *Embed Excel-compatible data grids in applets and Java desktop applications.*

Users can manipulate data as if they are using Excel, including the use of Excel formulas and functions. After users make changes in your embedded grid, they can save the file to their desktop as an Excel file or commit the changes back to your server where they can be saved to a database or passed to another application.

***PRODUCTIVE DEVELOPERS • SUCCESSFUL PROJECTS • EMPOWERED USERS***

ReportingEngines™
JAVA REPORTING TOOLS FROM ACTUATE

**WEB:** http://www.ReportingEngines.com
**EMAIL:** sales@ReportingEngines.com
**TEL:** 913-851-2200 • 888-884-8665
**FAX:** 913-851-1390

FREE TRIALS

**FORMULA ONE E.SPREADSHEET ENGINE** (100% PURE JAVA TOOL FOR BUILDING FINANCIAL APPLICATIONS)
Excel reporting. Excel-like data grids. Server-side calculation and business rules engine.

**FORMULA ONE E.REPORT ENGINE** (EMBED PDF, HTML, DHTML, AND XML REPORTS IN JSP AND SERVLETS)
Build reports against Java objects, JDBC, and XML. No report server to set up or maintain.

**FORMULA ONE E.REPORT ENGINE FOR WEBLOGIC WORKSHOP** (ACCESS LIQUID DATA AS A DATA SOURCE)
All the power of the regular e.Report Engine as a fully integrated BEA Workshop Extension.

**BEA PORTAL REPORTING SOLUTIONS** (POWERED BY THE FORMULA ONE E.REPORT ENGINE)
Visually build reports against BEA WebLogic Portal Server log files and BEA Liquid Data.

JDK1.5 concurrency packages (java.util.concurrent, java.util.concurrent.atomic, and java.util.concurrent.locks) rely on the dynamic creation of new threads in order to take advantage of concurrent algorithms, which is usually a no-no in the real-time world. Furthermore, it's inefficient for low-level libraries (too much overhead) and synchronization can be tricky.

To address this particular issue, a concurrent context has been created. It allows real-time applications to take advantage of parallel algorithms on multiprocessor cards or even single processors with hyper-threading technology without creating new threads. (HyperThreading doubles the number of executing threads per processor.) This objective is achieved by maintaining a limited number of threads on stand-by. These threads can then be utilized on demand to perform concurrent executions. If all concurrent threads are busy, the current thread executes the concurrent operation itself. Concurrent context is easy to use, provides automatic load-balancing between processors with almost no overhead, and does not require any synchronization code as the parent thread is not allowed (blocks on the exit() call) to exit its concurrent context until all concurrent executions are complete. As soon as a concurrent thread completes its execution, it becomes available again for more, resulting in concurrent threads/processors being busy most of the time. Last but not least, concurrent contexts guarantee the same behavior whether or not the execution is performed by the current thread or a concurrent thread, granted that the concurrent execution's order has no impact on the behavior. In particular, any exception raised by a concurrent thread is propagated to the parent thread and concurrent threads execute in the same context as their parent.

```
ConcurrentContext.enter();
try {
     ConcurrentContext.execute(runnable1);
     ConcurrentContext.execute(runnable2);
     ...
} finally {
     ConcurrentContext.exit(); // Waits for all concurrent threads
}                                   // to complete.
```

## Direct Memory Access: Struct and Union

It's not rare for real-time/embedded projects to use Java and C/C++ together. By mixing them, projects get the best of both worlds: the high-performance of C/C++ with the rapid development cycle typically associated with Java.

Until recently data exchange was problematic as the storage layout of Java objects is not determined by the compiler. The layout of objects in memory is deferred to runtime and determined by the interpreter (or just-in-time compiler). This approach allows for dynamic loading and binding, but also makes interfacing with C/C++ code difficult.

This particular issue has been addressed in the form of two public domain classes: Struct and Union. These two classes mimic the C struct and union types. They follow the same alignment rules, support the same features (e.g., bit fields, packing), and make it extremely easy to convert C header files to Java classes (one-to-one mapping).

Using these classes, embedded systems can map Java objects to a physical address to control hardware devices or communicate through shared memory with external apps.

## Conclusion

Garbage collection is not the only issue preventing Java from being used for a real-time system. Other issues include thread scheduling, accurate timer, synchronization overhead, lock queuing order, class initialization, and maximum interrupt response latency. Until now it has definitively been a "stopper." Because of it, most real-time systems today are developed in C/C++ despite the existence of Java compilers.

The good news is that whereas before you had to use C/C++ and some real-time OS, now you can use GCJ/J.A.D.E. and the same real-time OS (with JNI/Struct for the interface).

Pool contexts are a substitute for the complicated memory model of the RTSJ. The concept of scoped memory and immortal memory and how to transfer data between these areas leads to a cumbersome programming style. And the runtime checks for this model are a real performance killer. However, to see the full advantage of this approach for real time, you need a real-time kernel. Since the RTSJ (implemented as Reference Implementation or jRate) is the only available Java real time, it would be interesting to see some results on top of it.

## References

- *J.A.D.E. Real-Time FAQ:* http:/jade.dautelle.com/api/com/dautelle/realtime/package-summary.html#FAQ
- *RTJ API:* http://rtj.org/doc/index.html
- *Ajile RTJ chips:* www.ajile.com/downloads/aJ100Datasheet_1.3.pdf
- *JStamp:* http://jrealtime.systronix.com/
- *Restriction of Java for Embedded Real-Time Systems:* www.jopdesign.com/doc/rtjava.pdf
- *The Real-Time for Java Expert Group:* www.rtj.org
- Brosgol, B., et al. (2000). *The Real-Time Specification for Java*. Addison-Wesley.
- *RTSC (JSR-001):* www.rtj.org/rtsj-V1.0.pdf

*Listing 1*

```
PoolContext.enter(); // Allows for „stack‰ allocations.
try {
     ...
     ConcurrentContext.enter();
     try {
          // Concurrent executions have access to
          // the „stack‰ objects of the outer pool context.
          ConcurrentContext.execute(logic, ...);
          ConcurrentContext.execute(logic, ...);
     } finally {
          // Waits for concurrent executions to complete.
          ConcurrentContext.exit();
     }
     ...
} finally {
     // Recycles all stack objects including
     // objects allocated by concurrent threads.
     PoolContext.exit();
}
```

*Listing 2*

```
Matrix[] inputs = ...;
Matrix[] results = new Matrix[inputs.length];
for (int i=0; i < inputs.length; i++) {
     PoolContext.enter() {
     try {
          Matrix tmp = inputs[i].inverse();
          ... // Some function of inputs/results
          results[i] = (Matrix) tmp.export();
     } finally {
          // Recycles all temporary objects allocated.
          PoolContext.exit();
     }
}
```

*Listing 3*

```
// Defines UDP message internal fields.
class Message extends Struct { ... }

class MulticastConnection extends Thread {
     MulticastSocket socket;
     public void run() {
          byte[] bytes = new byte[1024];
          DatagramPacket packet
               = new DatagramPacket(bytes, bytes.length);
          Message msg = new Message(bytes);
          while (!socket.isClosed()) {
               PoolContext.enter();
               try {
                    socket.receive(packet);
                    ... // Process msg (real-time).
                    packet.setLength(bytes.length);
               } catch (IOException ioError) {
                    ... // Process error.
               } finally {
                    PoolContext.exit();
               }
          }
     }
}
```

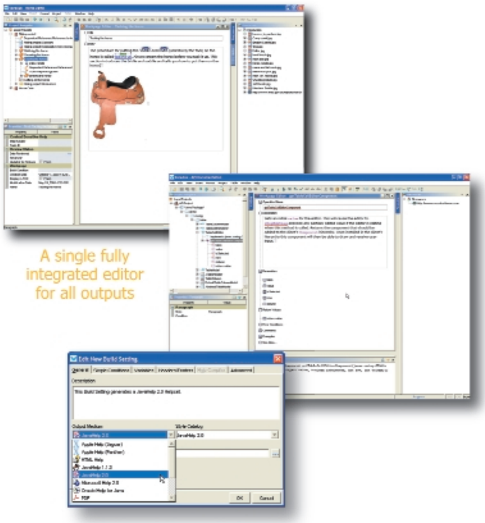
> Can't someone develop a single sourcing authoring tool that saves time, allows for collaboration and reduces headaches for developers and technical writers?

# Finally someone has!

## Introducing Veredus™ 2.0

### Single Sourcing Made Easy

VEREDUS



A single fully integrated editor for all outputs

Output is as easy as choosing the output medium, style and where to store the output to create Java Help

### API Documentation

> Produce documentation using terms and vocabulary common to writers and developers

> Create documentation automatically by scanning Java source code and Microsoft Windows COM Objects.

> Automatically generate signatures of API properties and methods, return values, parameters, and when available, error codes

> Re-scan developer code to merge changes into your Veredus project quickly and easily, keeping documentation accurate and up-to-date

> Generate documentation in a variety of formats from your Veredus project such as JavaHelp and PDF

### Single Sourcing

> A single product that creates content once for multiple deliverables in content and format

> Veredus saves you time, reduces headaches and creates W3C compliant output for customized printing, Web portals and a variety of Help formats.

### Help Authoring

> Quickly create professional help systems and user manuals from a single source.

> Change your output target without painful conversion processes.

> Produce output in any format: Java Help, Oracle Help, HTML Help, Apple Help, Microsoft Help, WinHelp and others.

> Translate the same information into different languages via XLIFF and a localization partner

**Download a FREE trial of Rascal Software Veredus 2.0**

**Visit www.rascalsoftware.com/jdj or call 206-624-7300**

Rascal SOFTWARE

# Scrolling on **Demand**

## A scrollable toolbar component

by Volker Simonis

Modern GUI programs enable you to easily access status information and functionalities through various menus, toolbars, and information panels. However, as a program becomes more complex or when users have the possibility of configuring and extending these components, they tend to get overfilled. This leads to scrambled or even truncated components.

This article introduces a new container component called ScrollableBar that can be used as a wrapper for any Swing component. As long as there is enough space to lay out the contained component, ScrollableBar is completely transparent. As soon as the available space gets too small, ScrollableBar will fade in two small arrow buttons on the left and the right side (or on the top and the bottom if in vertical mode) that can be used to scroll the underlying component, thus avoiding the above mentioned problems.

ScrollableBar is a lightweight container derived from JComponent that uses the standard Swing classes JViewport and JButton to achieve its functionality. It fills a gap in the set of the standard Swing components and offers the possibility of creating more robust and intuitive user interfaces.

### Introduction

Every professional application includes a fancy graphical user interface and with Swing, the standard widget set of Java, it's quite easy to create such applications. However, the design and implementation of a robust and user-friendly GUI is not a trivial task. One common problem is that the programmer has no knowledge about the client's desktop size. This may vary today from the standard notebook and flat panel resolution of 1024x768 to 1900x1200 for high-end displays. Even worse, Java applications can run on many other devices such as mobile phones, which have an even more restricted resolution.

Another challenge arises from the extensibility of applications. While the possibility of extending an application with various plugins may be a nice feature for the user, the fact that these plugins will populate the menus and toolbars in an unpredictable way presents new problems for the programmer.

One way to solve these problems is to limit the GUI components to a certain minimal size; however, this may impose unnecessary restrictions on the user. (Think, for example, of somebody who by default works with an application that needs a resolution of at least 1024x768 but who occasionally gives demo talks with a beamer that only supports an 800x600 resolution.) Furthermore, if an application with a graphical user interface pretends to be resizable by displaying a resizable frame, the user expects he will be able to resize it based on his needs, not the programmer's.

The second possibility is to do nothing and wait and see what happens. This is how most GUI applications are written today. Just compare Figure 1 with Figure 2 and see how parts of the status bars and toolbars are cut off if the window is shrunk beyond its optimal size. In the best case, the user could just reenlarge the application if this happens. In the worst case, if he or she is working on a device with a restricted resolution, it may be impossible to access the desired functionality. In any case such an application looks highly unprofessional.

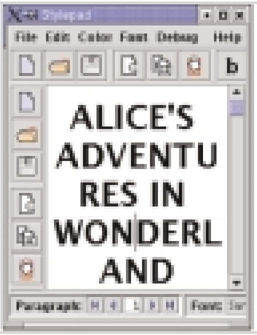
### Scrollable Menus and Toolbars

The solution for all of the above mentioned problems would be scrollable menus and toolbars. However, Swing, as many other widget sets, does not offer this type of component. Using the standard JScrollPane component as a container for menus and toolbars is not an option here as JScrollPane is too heavyweight; its scrollbars are simply too big. However, there is another Swing component that can serve as a template: since version 1.4, the JTabbedPane class offers the possibility of scrolling its panes instead of wrapping them on several lines if they don't fit on a single line. As shown in Figure 3, arrow buttons for moving the tabs have been added at the upper right part.

We now want to achieve the same behavior with menus, toolbars, and other status bars and information panels. To get a visual impression of how the modified components will look, compare Figures 1 and 4. They both show a screenshot of the Stylepad demo application shipping with every JDK, which has been extended by a vertical toolbar and a useful status bar (see Figure 2). While the menu, status bar, and toolbars are truncated and partially inaccessible in Figure 1, they can be scrolled and are fully functional in Figure 4 by using the arrow buttons that have been faded in.

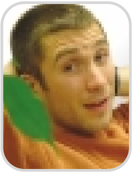
### The Implementation

I'll now describe how to implement a class called ScrollableBar, which can serve as a container for a java.awt.Container object or any other object derived from it. Most of the time ScrollableBar objects are completely transparent. If the place required by the wrapped component for layout becomes too small, the ScrollableBar object will fade in two arrow buttons at



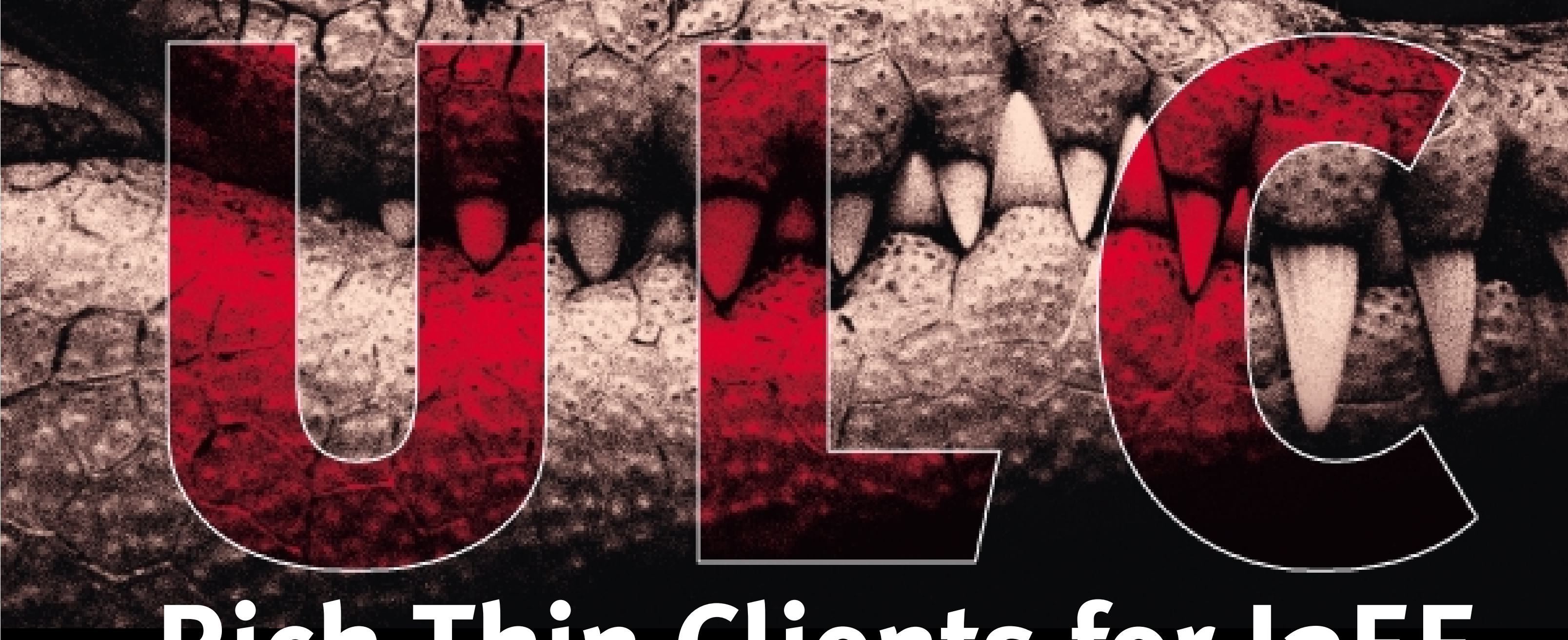
ALICE'S
ADVENTU
RES IN
WONDERL
AND

**Figure 1** The Stylepad application from Figure 2 with truncated tool and status bars



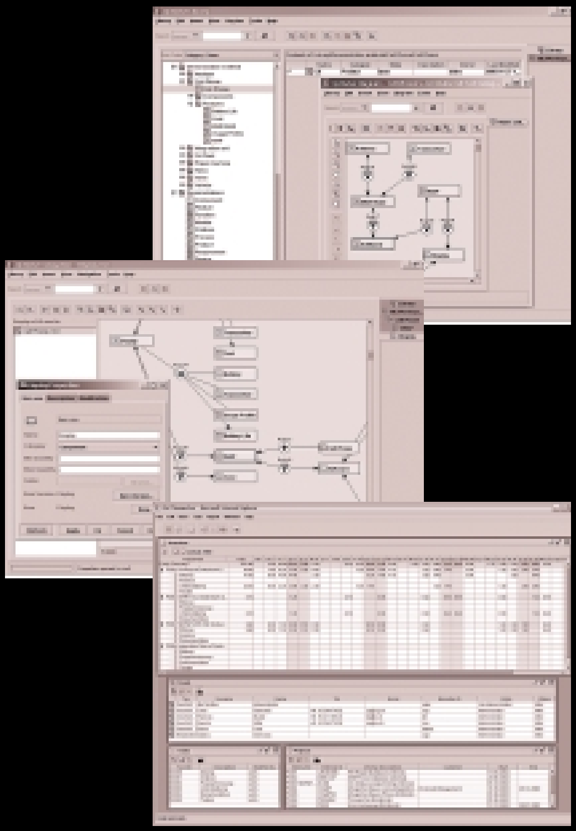
**Volker H. Simonis** is an independent software developer and consultant. He is an expert in the field of software design and documentation, i18n, generic programming, C++, and Java with about eight years of working experience in these areas. Volker received a masters degree in computer science from the University of Tübingen (Germany) and is currently finishing his PhD thesis.

*simonis@progdoc.com*



# Rich Thin Clients for J2EE

## UltraLightClient offers a server-side API to Swing, providing rich GUIs for J2EE applications.



- **Server-side programming model:** develop scalable web applications for thousands of users as simply as stand-alone Swing applications.
- **Superior security:** no application code is executed on the client, nothing is stored in a browser cache.
- **Application deployment on server:** a lean Java presentation engine on the client serves any number of applications.
- **Pure Java library:** use your favorite IDE and get add-on tools for visual editing, client/server simulation, and load/performance testing.

-canoo-com

Canoo Engineering AG http://www.canoo.com/ulc/

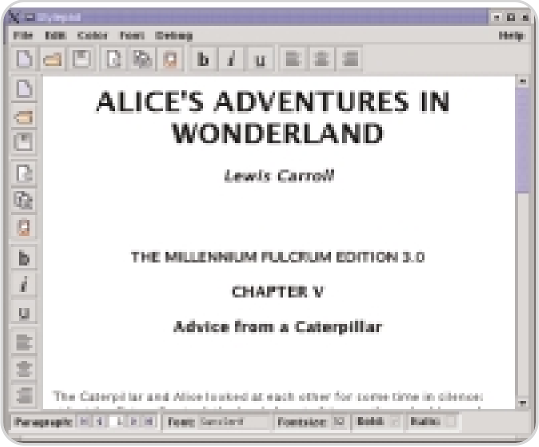
**Download your free trial today!**

Ready for IBM WebSphere. Studio software the left and right sides of the component (or on the top and the bottom if in vertical mode), which can be used to scroll the wrapped component. As soon as there is enough place for the layout of the enclosed component again, these arrow buttons will immediately disappear.

## The Swing Architecture

For a better understanding of the ScrollableBar implementation, it's helpful to revisit the architecture of Swing. The Swing library is a modern widget set based on the Model-View-Controller (MVC) pattern. While the classical MVC pattern consists of three independent parts – model, view, and controller – Swing uses a simplified version of this pattern in which the view and the controller parts are combined in a so-called Delegate (see Figures 5 and 6).

As an example, Figure 6 shows how this Model-Delegate pattern applies to the JButton class. In Swing, all visible components are descendants of the JComponent class. They usually capsule a component-specific model with a delegate object, which is a descendant of the ComponentUI. These delegates are called user interface (UI) classes in Swing. They are look-and-feel specific, i.e., they're used to implement the different look and feel–dependent properties of a component, but they can also be used for other kinds of customization, for example, localization.



ALICE'S ADVENTURES IN WONDERLAND

Lewis Carroll

THE MILLENNIUM FULCRUM EDITION 3.0

CHAPTER V

Advice from a Caterpillar

The Caterpillar and Alice looked at each other for some time in silence:

**Figure 2** The Stylepad application at preferred size

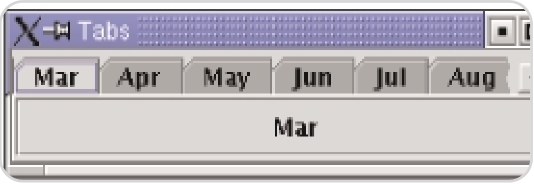
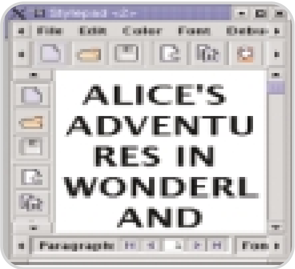


**Figure 3** JTabbedPane with the tab layout policy set to SCROLL_TAB_LAYOUT



ALICE'S ADVENTU RES IN WONDERL AND

**Figure 4** Stylepad with scrollable menu, tool, and status bars

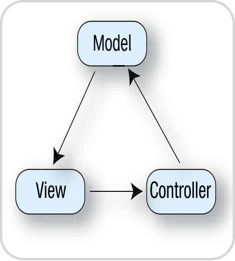


**Figure 5** The classical MVC pattern

One of the main responsibilities of the UI delegate is to paint the component it's tied to. In contrast to the AWT library, in Swing it's not the paint() method of every component that does the work of painting itself. Instead, the component's paint() method just calls the paint() method of its delegate with a reference to itself.

### *The ScrollableBar Class*

Figure 7 shows the class diagram of the ScrollableBar class. As already mentioned, it's derived from JComponent. It also implements the SwingConstants interface in order to easily access the constants HORIZONTAL and VERTICAL, which are defined there.

ScrollableBar has four properties. The two Boolean properties "horizontal" and "small" store the orientation of the component and the size of the arrows on the scroll buttons. The integer property "inc" stores the amount of pixels by which the enclosed component will be scrolled if one of the arrow buttons is being pressed. Smaller values lead to a smoother but slower scrolling. Finally, the wrapped component is stored in the "comp" property. While "horizontal" is a read-only property that can be set only in the constructor, the other three properties are read/write bound properties as described in the JavaBeans specification.

The following code shows the two-argument constructor of the ScrollableBar class:

```
public ScrollableBar(Component comp, int
orientation) {
 this.comp = comp;
 if (orientation == HORIZONTAL) {
   horizontal = true;
 }
 else {
   horizontal = false;
 }
 small = true;
// Arrow size on scroll button.
 inc = 4;
 // Scroll width in pixels.
 updateUI();
}
```

Notice the call to updateUI() in the last line of the constructor. As can be seen in the following code, updateUI() calls the static method getUI() from the class UIManager to query the right UI delegate and associates it with the current ScrollableBar object.

```
public String getUIClassID() {
  return "ScrollableBarUI";
}

public void updateUI() {
  setUI(UIManager.getUI(this));
  invalidate();
}
```

UIManager.getUI() calls getUIClassID() to get the key that's used to query the actual UI delegate from a look and feel–dependent internal table. Usually the association of the standard Swing components with the appropriate UI classes is done by the different look and feels while they are initialized. However, as we are writing a new component, we have to establish this link manually, as shown below:

```
static {
  UIManager.put("ScrollableBarUI",
      "com.languageExplorer.widgets. -
                    ScrollableBarUI");
}
```

Notice that linking a component to its UI delegate in this way results in the same UI class being used independent of the actual look and feel.

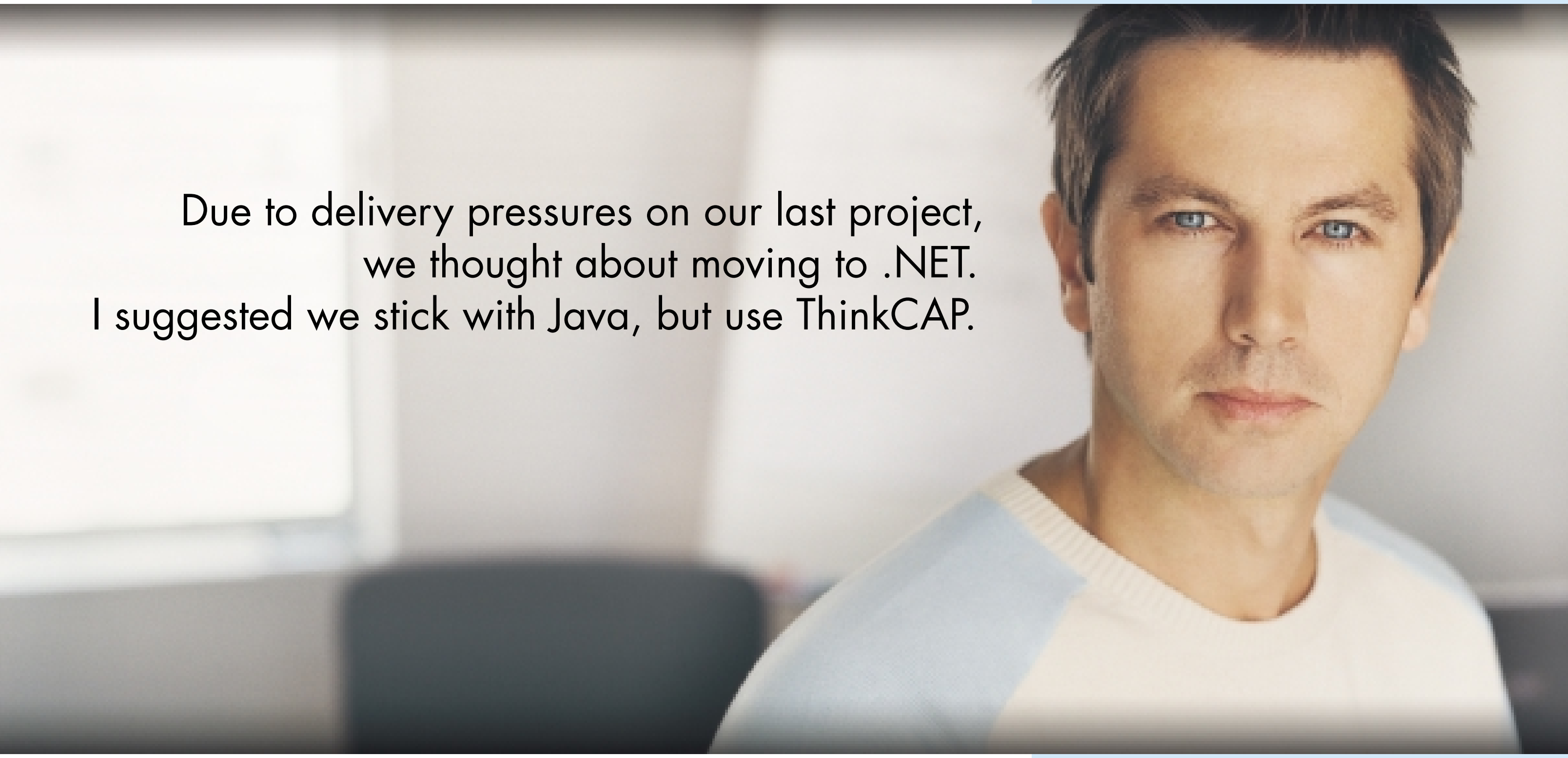
Besides the getter and setter methods for the corresponding properties, there's no more functionality in the ScrollableBar class. All the painting and user interaction is handled by the UI delegate ScrollableBarUI.

### *The ScrollableBarUI Class*

One of the most important methods of the UI classes is installUI(), which is called every time a component is associated with its UI delegate (see Listing 1). This gives the UI delegate a chance to properly initialize itself and the component it's responsible for.

In our case, the UI delegate queries and stores the component's properties along with a reference to the component as private instance variables. Further on, it creates two arrow buttons and an object of type JViewport, which is used to wrap the scrollable component.

Based on the orientation of the


Due to delivery pressures on our last project,
we thought about moving to .NET.
I suggested we stick with Java, but use ThinkCAP.

associated ScrollableBar object, the newly created elements are then being added to it by using a vertical or horizontal box layout. Notice that the scroll buttons are initially set to be invisible. Finally, the UI object registers itself as a property change listener on the associated component, as a change listener on the viewport, and as a mouse listener on the arrow buttons.

The UI delegate is informed about every size change of the ScrollableBar object and the wrapped component by receiving a ChangeEvent from the viewport object. Depending on the new sizes, it can change the visibility state of the arrow buttons and re-lay out the component. Property changes in the ScrollableBar object are signaled to the UI delegate by a PropertyChangeEvent. Based on these events, it can update the internally cached values of these properties.

Finally, the events resulting from the user interactions on the scroll buttons are handled by the different mouse listener methods. The UI delegate keeps a private boolean instance variable pressed that's set to true if a button was pressed, and reset to false as soon as the button is released or the mouse pointer leaves the button. As can be seen in Listing 2, pressing one of the buttons also starts a new thread that scrolls the underlying component by "inc" pixels in the corresponding direction and then sleeps for a short time. These two actions are subsequently repeated in the thread as long as the value of the instance variable "pressed" is true, while the amount of sleeping time is reduced in every iteration step. This results in a continuously accelerating scrolling speed, as long as the user keeps on pressing the arrow button.

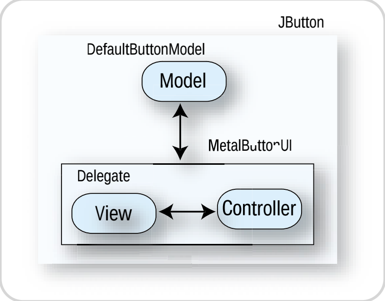


**Figure 6** The Model-Delegate pattern used in Swing

It should be noted that we don't need any special paint method for the ScrollableBarUI class, because painting occurs naturally from the standard Swing button and viewport components that we used.

After we've discussed the main parts of the implementation, it should be evident why the advantages of dividing the functionality of the ScrollableBar class into two classes outweigh the coding overhead. First, we cleanly separated the properties of the component from the way that it displays and interacts with the user. Second, it's easy now to define a new UI delegate that renders the component in a different way or to derive a new UI delegate from the existing one that slightly adopts appearance or user interaction properties to a specific look and feel.

## Using the ScrollableBar Class

Using the ScrollableBar class is easy and straightforward. In fact we can wrap every arbitrary Swing component inside a ScrollableBar object by passing it as an argument to the constructor when creating the object. For the example application shown in Figure 4 it was necessary to only change a single line:

```
JToolBar toolbar = new JToolBar();
...
panel.add("North", toolbar);
```

from the original Stylepad application to:

```
JToolBar toolbar = new JToolBar();
...
panel.add("North",
  new ScrollableBar(toolbar));
```

in order to make the horizontal toolbar scrollable if the space becomes too small to render it as a whole.

In general, the ScrollableBar class is recommended for wide and not very high components in horizontal mode and narrow and high components in vertical mode. If used for other components, the scroll buttons would get too big and take up too much space to be really useful.

### *Menu Bars in JFrame Objects*

As shown in the last section, it's very easy to use the ScrollableBar class in your own applications. Even upgrading existing applications isn't very hard. The only problem that may arise is when a ScrollableBar should be used as a wrapper for a menu bar that will be added directly to a JFrame object. (Notice that in our example application, the menu bar has been added to a JPanel object before the whole panel has been added to the JFrame object.)

The problem arises because JFrame provides a specialized setJMenuBar() method for adding menu bars and this method expects an argument of Type JMenuBar. At first glance, we could just use one of the generic add() methods defined in JFrame's ancestor classes instead. However, if we take a closer look, we'll see that the problem is a bit more complex.

First, in the case of JFrame, children are not being added to the component directly, but to the so-called "root pane", which is a special child component of every JFrame. However, we also can't add the menu bar directly to the root pane, because the root pane also has a special method called setJMenuBar() that expects a JMenuBar object as an argument. Using this method to add menu bars is essential, because only then will the RootLayout layout manager used by the JRootPane class honor the presence of the menu bar. RootLayout, which is a protected inner class of JRootPane, uses the protected JRootPane property "menuBar" that has been set by JRootPane.setJMenuBar() for layout calculations.

To cut a long story short, we have to create a new SMJFrame class (which stands for Scrollable Menu JFrame) that

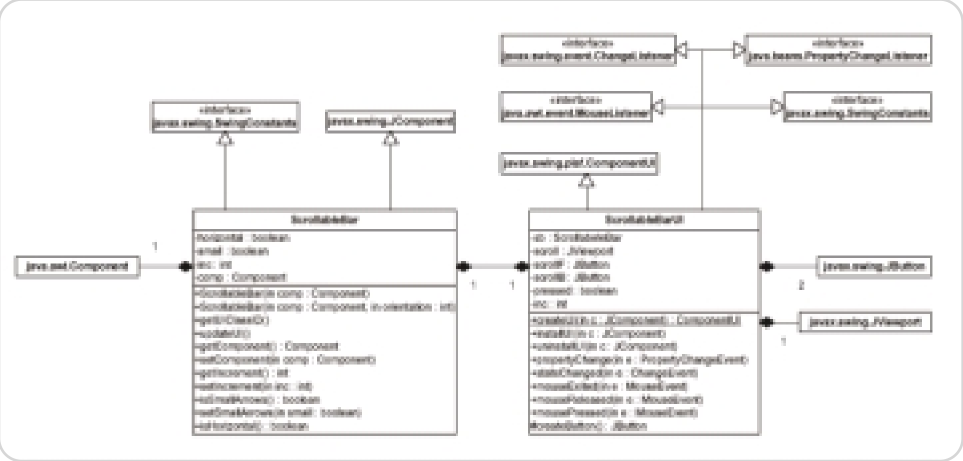

«interface»
javax.swing.SwingConstants
javax.swing.JComponent
«interface»
javax.swing.event.ChangeListener
«interface»
java.beans.PropertyChangeListener
«interface»
java.awt.event.MouseListener
«interface»
javax.swing.SwingConstants
javax.swing.plaf.ComponentUI
java.awt.Component
1
ScrollableBar
-horizontal : boolean
-small : boolean
-inc : int
-comp : Component
+ScrollableBar(in comp : Component)
+ScrollableBar(in comp : Component, in orientation : int)
+getUIClassID()
+updateUI()
+getComponent() : Component
+setComponent(in comp : Component)
+getIncrement() : int
+setIncrement(in inc : int)
+isSmallArrows() : boolean
+setSmallArrows(in small : boolean)
+isHorizontal() : boolean
1
1
ScrollableBarUI
-sb : ScrollableBar
-scroll : JViewport
-scrollF : JButton
-scrollB : JButton
-pressed : boolean
-inc : int
+createUI(in c : JComponent) : ComponentUI
+installUI(in c : JComponent)
+uninstallUI(in c : JComponent)
+propertyChange(in e : PropertyChangeEvent)
+stateChanged(in e : ChangeEvent)
+mouseExited(in e : MouseEvent)
+mouseReleased(in e : MouseEvent)
+mousePressed(in e : MouseEvent)
#createButton() : JButton
javax.swing.JButton
2
javax.swing.JViewport
1

**Figure 7** The UML class diagram of ScrollableBar and ScrollableBarUI

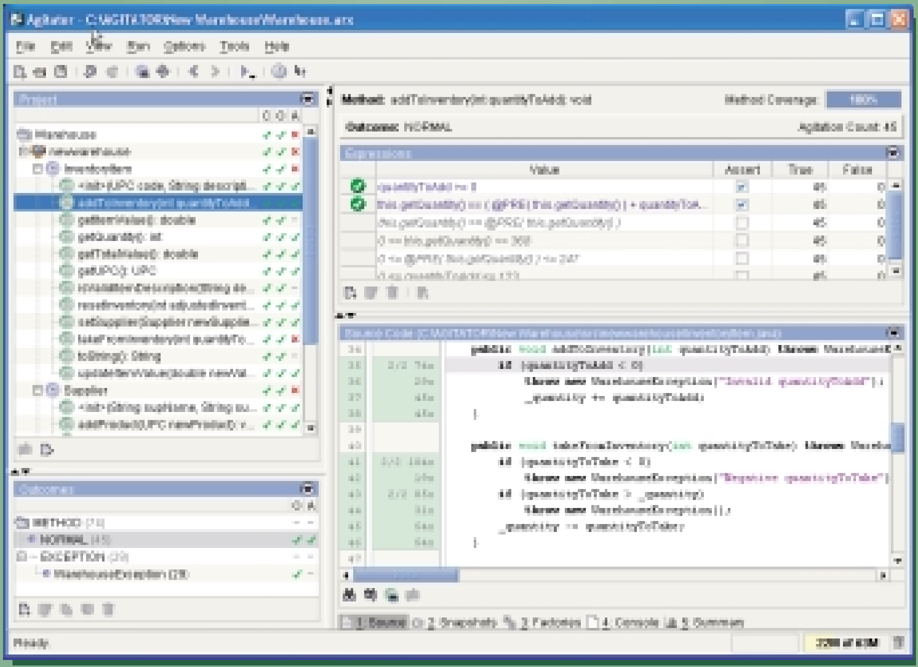
See us at JavaOne at booth #832, and attend Alberto Savoia's presentation "Beyond JUnit."

# JUnit started the Developer Testing revolution.

# Are you ready for what comes next?

See when tests meet your criteria for (C)ode, (O)utcome, and (A)ssertion coverage for each class and method.

Create tests for all possible outcomes, including expected and unexpected exceptions.



```
public void addToInventory(int quantityToAdd) throws WarehouseE
    if (quantityToAdd < 0)
        throw new WarehouseException("Invalid quantityToAdd");
        _quantity += quantityToAdd;
}

public void takeFromInventory(int quantityToTake) throws Wareho
    if (quantityToTake < 0)
        throw new WarehouseException("Negative quantityToTake")
    if (quantityToTake > _quantity)
        throw new WarehouseException();
    _quantity -= quantityToTake;
}
```

Get extreme code coverage with automated test data generation and fully configurable test data factories.

Automatically generate observations for insight into your code's behavior and convert them to durable unit tests with a single click.

Use the integrated code coverage to make sure all your code is fully tested.

## Agitator™ – The future of Developer Testing for Java™

Agitator intelligently exercises Java code to show you observations about its behavior. You can convert valid observations to unit tests with a single click. This unique process, called Software Agitation™, helps you create thorough sets of durable unit tests. You can edit and expand your code with confidence and identify bugs as you write or modify your programs. In conjunction with the Agitar Management Dashboard, Agitator delivers unprecedented productivity, visibility, and control over the developer testing process, leading to profound software quality improvements and cost reductions.

Learn more about Software Agitation, and visit www.agitar.com today!

Agitar™
SOFTWARE
Know Your Code
Trust Your Code

overrides the createRootPane() method to return a new, customized root pane class. For this purpose we just derive an anonymous class from JRootPane that overrides the two methods setJMenuBar() and createRootLayout().

setJMenuBar(), the first of these two methods, wraps the menu bar into our ScrollableBar class before storing it as a protected instance variable and adding it to the layered pane that's a part of the root pane.

The second method, createRootLayout(), returns an anonymous class that inherits from the JRootPane protected inner class RootLayout. It overrides the layout methods in that class in such a way that they use the ScrollableBar instance variable for layout calculations instead of using the bare menu bar, as it was done by the original version of the methods.

These modifications finally give the desired result. A call to setJMenuBar() on a SMJFrame object will be forwarded to the customized root pane. There the menu bar will be wrapped into a ScrollableBar object before it will actually be added to the frame. Because the customized root pane uses a customized layout manager, it will handle the scrollable menu bar in the same way in which a JFrame object handles an ordinary menu bar. With respect to all other concerns, SMJFrame behaves exactly like its ancestor JFrame.

### *Limitations*

The only limitation for the use of the ScrollableBar class so far is that it cannot handle floating tool bars. This is because JToolBar objects have to be laid out in a container whose layout manager is of type BorderLayout if they want to be floatable. In addition, no other children can be added to any of the other four "sides." This is obviously not the case if the toolbar is wrapped inside a ScrollableBar object.

Fixing this problem would require extensive changes in BasicToolBarUI, the UI delegate of JToolBar. Unfortunately, since not all the methods that need to be customized are declared public or protected, a complete rewrite of the delegate would be necessary.

## Conclusion

This article presented quite a small and simple yet powerful container class, which fills a gap in the set of standard Swing components. Using it involves no overhead at development time or at runtime but yields a lot of benefits. The most important ones are better usability and user friendliness, and more robust and intuitive GUI applications.

The source code presented in this article is available at www.progdoc.org/ScrollableBar.

## References

- Eckstein, R., Loy, M., and Wood, D. (1998). *Java Swing.* O'Reilly.
- Gamma, E., Helm R., Johnson, R., and Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software.* Addison-Wesley.
- Hamilton, G. (Ed.) JavaBeans Sun Microsystems, Version 1.01-A, August 1997: http://java.sun.com/beans
- Simonis, V. "International Swinging: Making Swing Components Locale-Sensitive." C/C++ *Users Journal, Java Supplement*, August 2002: www.cuj.com/java/jsup2008/
- Simonis, V. Prog DOC - A Program Documentation System: www.progdoc.org
- Zukowski, J. "Magic with Merlin: Scrolling tabbed panes": www-106.ibm.com/developerworks/java/library/j-mer0905/
- Zukowski, J. and Stanchfield, S. "Fundamentals of JFC/Swing, Part II", MageLang Institute: http://developer.java.sun.comdeveloper/onlineTraining/GUI/Swing2

*Listing 1: installUI() from ScrollableBarUI.java*

```
public void installUI(JComponent c) {
  sb = (ScrollableBar)c;

  inc = sb.getIncrement();
  boolean small = sb.isSmallArrows();

  // Create the Buttons
  int sbSize =
    ((Integer)(UIManager.get( "ScrollBar.width" ))).intValue();
  scrollB = createButton(sb.isHorizontal()?WEST:NORTH,
                                                      sbSize,
small);
  scrollB.setVisible(false);
  scrollB.addMouseListener(this);

  scrollF = createButton(sb.isHorizontal()?EAST:SOUTH,
                                                      sbSize,
small);
  scrollF.setVisible(false);
  scrollF.addMouseListener(this);

  int axis = sb.isHorizontal()?BoxLayout.X_AXIS:BoxLayout.Y_AXIS;
  sb.setLayout(new BoxLayout(sb, axis));

  scroll = new JViewport() {
    ... see source code ...
  };

  Component box = sb.getComponent();
  scroll.setView(box);

  sb.add(scrollB);
  sb.add(scroll);
  sb.add(scrollF);

  // Install the change listeners
  scroll.addChangeListener(this);
  sb.addPropertyChangeListener(this);
}
```

*Listing 2: mousePressed() from ScrollableBarUI.java*

```
public void mousePressed(MouseEvent e) {
  pressed = true;
  final Object o = e.getSource();
  Thread scroller = new Thread(new Runnable() {
    public void run() {
      int accl = 500;
      while (pressed) {
        Point p = scroll.getViewPosition();
        ... Compute new view position ...
        scroll.setViewPosition(p);
        try {
          Thread.sleep(accl);
          if (accl <= 10) accl = 10;
          else accl /= 2;
        } catch (InterruptedException ie) {}
      }
    }
  });
  scroller.start();
}
```

Joe Winchester
Desktop Java Editor

# Where Are the Flying Cars?

Several years back I was watching *Independence Day*, a fairly decent movie about aliens invading earth. It was an enjoyable film with some pretty neat special effects, except my suspension of disbelief broke down when Jeff Goldblum decided he would infect an alien spaceship's computer defense system with a software virus. Doing so would deactivate the force field and allow Will Smith to jump into the pilot's seat and sneak back undetected to alien HQ, before loading the virus onto the mother ship and saving the planet. All good so far, except at the key point when Jeff Goldblum reached for his Apple Macintosh, cranked up a command prompt, and proceeded to write some DOS syntax. As what looked suspiciously like the result of a "dir /w" command scrolled rapidly along the command prompt, the alien spaceship threw up its hands in surrender. For me the whole movie was ruined at this point. Curiously I'm not sure why it shouldn't have been so before given the plotline's basis around little green men in dog fights with jet fighters, but the killer blow for me was the sight of a DOS prompt on a Mac.

While it's fun to knock films for failing to do software correctly, there is a purpose to my current diatribe. By continually mystifying and shrouding computers as the realm of the geek and the nerd and creating ridiculous user interfaces, they perpetuate the idea that computers should be hard to use. The syllogism is that if programmers create hard-to-use applications, they are somehow creating a sophisticated piece of software. Nothing could be further from the truth – the GUI is all about simplicity through sophistication.

When I was at university in the 1980s I used an IBM 3084 and a Honeywell computer, both of which were powerful boxes but had no GUI in sight. In my last year I had my first Macintosh and thought, "This is the future," and it pretty much inspired me to enter computing as a profession. I was shocked, however, by the fact that so much business software seemed oblivious to the GUI revolution that was to come, and my early colleagues mocked drag and drop with loaded acronyms like WIMP (windows, icons, mouse, and pull-down menus).

> "The GUI is all about simplicity through sophistication"

Hollywood movies seem to have a very odd concept of a user interface, and it's all the more puzzling to me since computers are household items that presumably everyone working on a film has had access to. In earlier movies there wasn't such a ubiquity of PCs in people's homes and offices, so film directors made logon screens with huge fonts while error messages were shown with giant flashing red graphics and klaxon sound effects. Teenagers invariably hacked into any computer from their tree house simply by writing a random password generator and watching it break each digit of the password one by one as they supped on their kiddie soda. This formed the plotline for numerous films, including the 1983 hit *War Games*, which was made all the more unbelievable by the fact that once the whiz kids had managed to unknowingly start World War III, they averted it by reprogramming the defense department's computer to play tic-tac-toe with itself. You wonder what the IT department who programmed the original software was doing at the time the generals wanted to stop the impending war – perhaps they had all been laid off and the code maintenance outsourced overseas? Even latter day films showing people using e-mail software have 24 point font software with each letter of the incoming mail typing itself character by character.

Most of the current GUIs that we use were conceived back in the 1970s at Xerox Parc. This is described excellently in the book *Dealers in Lightning*, which covers the almost incredulous opposition the early developers encountered by many who were opposed to the GUI as being anything more than research that would never catch on. The other amazing thing for me is that nothing much seems to have come along since the Xerox project. The basic concepts of scrollbars, combo boxes, moveable windows, menus, and everything we use were invented over 30 years ago, and while it's good, it's by no means perfect. Where is the next innovation coming from? It's not from movies that latch onto the latest buzzword and try to cover it in mystique (not too dissimilar from a tech company's marketing department). It could come out of research as the Xerox project did, or perhaps it will come from an entirely new angle. Wherever it comes from, the key to its success will be simplification. The WIMP made the GUI easier to use, not harder, and this should be the driving force behind any piece of interface design. There is no shame in making something easy to use, and hopefully one day soon I'll see a Hollywood movie where the software being used is simple and obvious, and breaks the cliché of an expert operator having to be a boffin or child prodigy. Talking of boffins, next time you hit a browser check out www.moller.com/skycar – it's a flying car. Enjoy.

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

*joewinchester@sys-con.com*

# Javavavoom!

## Introducing a high-performance database that's 100% Java.

**Berkeley DB Java Edition**

**Download at www.sleepycat.com/bdbje**

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

Berkeley DB JE has a unique architecture that's built for speed. The software executes in the JVM of your application, with no runtime data translation or mapping required. Plus Berkeley DB JE has been specifically designed to handle highly concurrent transactions, comfortably managing gigabytes of data. And because it's built in your language of choice, your organization enjoys shorter development cycles and accelerated time-to-market.

**Experience the outstanding performance of Berkeley DB JE for yourself.**
Download Berkeley DB JE today at **www.sleepycat.com/bdbje**. Register now, and you'll also receive a 15% discount on a commercial license purchased before November 30, 2004.

SLEEPYCAT
SOFTWARE
Makers of Berkeley DB

# GUI Design Patterns

## Choosing data from long lists

by Heman Robinson

Long lists of data present a problem for GUI development. This occurs when choosing records from large database tables or recipients from a long list of e-mail addresses, or any time a subset of data must be chosen from a long list or table.

Two issues arise when choosing from long lists of data. First, what is the best GUI for long lists so that the users understand how to navigate and manipulate their data? Second, if the list holds more than a few hundred items, performance become an issue.

In this article, simple examples show how to optimize usability for this common GUI design pattern. In a later article, more complex examples will show how to optimize performance for long lists.

### GUI Design Patterns

In the late 1990s, a GUI design pattern emerged for choosing multiple objects from long lists. The book *GUI Design Essentials* by Susan Weinschenk, et al, called this the Selection Summary pattern. In "A Dual Listbox Selection Manager" by Steve Aube, it's also known as the Dual Listbox Selection interface. In "The Java Look and Feel Guidelines, Advanced Topics" by Sun Microsystems, it's called the Add-and-Remove idiom. Figure 1 shows an example of this GUI design pattern.

The Add-and-Remove design pattern has many variations. One common enhancement is to provide "Move Up" and "Move Down" buttons to reorder the chosen list. Sometimes the chosen list is displayed as a table (as in Figure 2) to show additional information.

This GUI design pattern has several advantages. By presenting the original and chosen lists in a symmetric layout, learning is simplified: once users learn one part of the GUI, the corresponding part works consistently. Also, because objects are moved from one location to another, directional icons and disabling can be used to guide the user.

The most important advantage of a standard GUI design like this is that because it is widely known, users will most likely have seen it before and therefore understand it quickly. A standard GUI design pattern is the simplest way to achieve usability.

### Lay Out the Components

Listing 1 shows how this GUI design pattern can be implemented using the Swing components JList, JScrollPane, ImageIcon, and GridBagLayout. The GridBagLayout enables the alignment of unequal-sized components and provides good behavior on resizing. For a basic implementation, we need a grid with three columns and five rows. Lists, labels, and buttons are placed using GridBagConstraints so that when the component is resized, the lists grow or shrink while other components retain their original size (see Figure 3).

To support the Add-and-Remove pattern, the list contents must be mutable; i.e., the lists must dynamically change in response to the button presses. The DefaultListModel provides mutable storage that performs satisfactorily for lists of moderate size.

In Listing 1, the ActionListener interface connects the buttons to the lists. The actionPerformed method uses the button text to determine which of the four buttons was pressed.

> The DefaultListModel is not the default ListModel. By default, JLists use an internal ListModel that is immutable: you can't change its contents. This internal ListModel is efficient when applications only need to view a list because it doesn't carry the overhead of changing list contents.
>
> To change list contents, as in the Add-and-Remove pattern, requires a mutable ListModel. Swing provides one in javax.swing.DefaultListModel, and it gives good performance for most applications. For long lists DefaultListModel is not particularly fast, and a later article will show how to improve on its performance.

The code in Listing 1 produces the GUI shown in Figure 4.

### Follow the Guidelines

"The Java Look and Feel Guidelines" specify precise spacings in multiples of six pixels between components. To follow these guidelines, EmptyBorders are added around the buttons and labels.

```
_originalLabel.setBorder(
   BorderFactory.createEmptyBorder(
   0, 0, 6, 0 ));
_chosenLabel.setBorder(
   BorderFactory.createEmptyBorder(
   0, 0, 6, 0 ));
_add.setBorder(
   BorderFactory.createCompoundBorder(
   BorderFactory.createEmptyBorder(
   0, 11, 5, 11 ),
   _add.getBorder()));
_addAll.setBorder(
   BorderFactory.createCompoundBorder(
   BorderFactory.createEmptyBorder(
   0, 11, 11, 11 ),
   _addAll.getBorder()));
_remove.setBorder(
   BorderFactory.createCompoundBorder(
   BorderFactory.createEmptyBorder(
   0, 11, 5, 11 ),
   _remove.getBorder()));
_removeAll.setBorder(
   BorderFactory.createCompoundBorder(
   BorderFactory.createEmptyBorder(
   0, 11, 0, 11 ),
   _removeAll.getBorder()));
```

The Guidelines also recommend mnemonics to support keyboard equivalents of mouse actions, as shown in the code below.

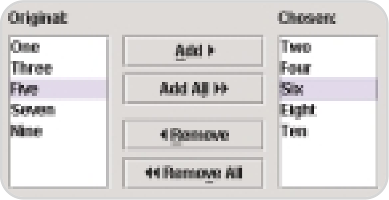


**Figure 1** Add-and-Remove pattern

**Heman Robinson** is a senior developer with SAS Institute in Cary, NC. He holds a BS in mathematics from the University of North Carolina and an MS in computer science from the University of Southern California. He has specialized in GUI design and development for 15 years and has been a Java developer since 1996.

*hemanrobinson@yahoo.com*

Compuware

# OptimalJ®

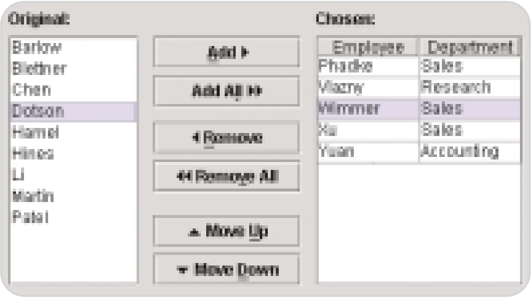
## THE POWER TO Develop, Transform, Reuse

Put your J2EE™ application development into overdrive—with unmatched quality and unprecedented flexibility—using Compuware OptimalJ. Increase developer productivity up to 90%. Seamlessly instill quality into development. Forge existing infrastructures and new technologies into an integrated whole. Realize business agility with applications that are truly reusable, time and time again.

**THE POWER IS RIGHT HERE.**

THE LEADER IN IT VALUE.

COMPUWARE®
www.compuware.com



| Employee | Department |
|---|---|
| Phadke | Sales |
| Vlazny | Research |
| Wimmer | Sales |
| Xu | Sales |
| Yuan | Accounting |

**Figure 2** Add-and-Remove pattern with "Move" buttons and table

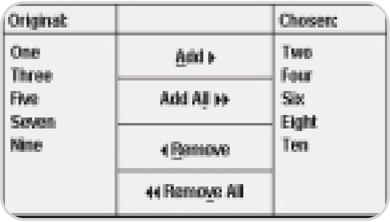


**Figure 3** GridBagLayout

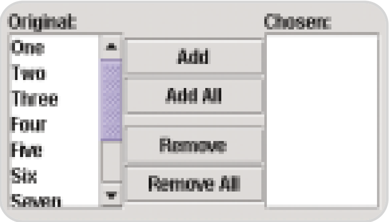


**Figure 4** ListChooser

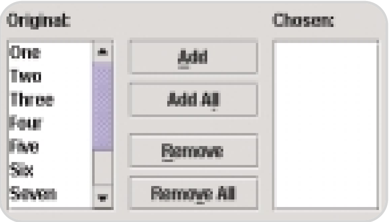


**Figure 5** ListChooser with spacing and mnemonics

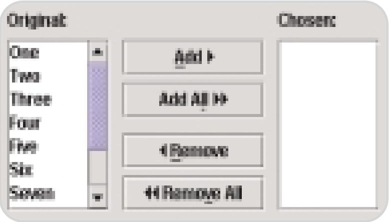


**Figure 6** List selector with icons

```
_add.setMnemonic( 'A' );
_addAll.setMnemonic( 'l' );
_remove.setMnemonic( 'R' );
_removeAll.setMnemonic( 'v' );
```

Note the difference between Figures 4 and 5. By following the guidelines, the GUI appearance is improved.

This example uses the guidelines for the Java ("Metal") Look and Feel. If you target a different platform the guidelines will be different. For example, Windows Guidelines specify different spacings between the components.

Whatever the target platform, the reason for following guidelines is to make life easier for the user. When your application looks and works the way the users expect, they have less to learn.

## Help the Users Learn

"The Java Look and Feel Guidelines, Advanced Topics" by Sun recommends the use of icons in the buttons. If icons are stored with the ListChooser class, they can be assigned with the following code.

```
_add.setIcon( new ImageIcon(
   ListSelector.class.getResource(
   "images/add.gif" )));
_add.setHorizontalTextPosition(
   SwingConstants.LEFT );
_addAll.setIcon( new ImageIcon(
   ListSelector.class.getResource(
   "images/addAll.gif" )));
_addAll.setHorizontalTextPosition(
   SwingConstants.LEFT );
_remove.setIcon( new ImageIcon(
   ListSelector.class.getResource(
   "images/remove.gif" )));
_remove.setHorizontalTextPosition(
   SwingConstants.RIGHT );
_removeAll.setIcon( new ImageIcon(
   ListSelector.class.getResource(
   "images/removeAll.gif")));
_removeAll.setHorizontalTextPosition(
   SwingConstants.RIGHT );
```

These simple icons transform the buttons into a diagram of information flow through the GUI. The icons make the users' possible actions clear before they read the buttons' text, enabling new users to understand the buttons' functions at a glance.

The icons in Figure 6 are chosen because they're familiar to most users; similar icons appear on most tape and disk players. "The Java Look and Feel Guidelines, Advanced Topics" illustrate other icons specific to the Java look and feel. Whatever icons are chosen, the program should provide equal margins on the left and right to ensure consistent spacing between the icons and the button text. A slight margin below the icon, as shown in Figure 7, can improve vertical alignment.

## Prevent User Errors

One of the best ways to improve usability is to prevent the user from making a mistake. The authors of *GUI Design Essentials* recommend disabling unavailable actions. This technique guides users through the GUI, preventing them from taking actions that make no sense.

This effective GUI design pattern is trivially easy to code. The ActionListener interface enables the "Add All" and "Remove All" buttons when the lists contain objects, and disables them when the lists are empty.

```
public void actionPerformed(
   ActionEvent e )
{     .
      .
      .
   _addAll.setEnabled(
      originalModel.getSize() > 0 );
   _removeAll.setEnabled(
      chosenModel.getSize() > 0 );
}
```

The ListSelectionListener interface enables the "Add" and "Remove" buttons when objects are selected, and disables them when no objects are selected.

```
public void valueChanged(
   ListSelectionEvent e )
{  _add.setEnabled( _originalList.
      getSelectedValues().length > 0 );
   _remove.setEnabled( _chosenList.
      getSelectedValues().length > 0 );
}
```

As shown in Listing 1, the ListChooser registers itself as an ActionListener to receive events when buttons are pressed. To receive events when list contents are selected, the ListChooser registers itself as a ListSelectionListener. Also, at the end of the constructor, two dummy events are initiated to set the default enabled states.

```
public ListChooser(
   Object[] original, Object[] chosen )
{     .
      .
      .
   _originalList.
      addListSelectionListener( this );
   _chosenList.
      addListSelectionListener( this );
   actionPerformed(
      new ActionEvent( this, 0, "" ));
   valueChanged(
      new ListSelectionEvent(
      this, 0, 0, false ));
}
```

The effect of this code is shown in Figure 8. Rather than trying to deal with the user pressing the wrong button, the program uses disabling to pre-

We have
a problem.

vent the mistake from occurring. Disabling unavailable actions is one of the simplest and most effective GUI design patterns.

## GUI Variations

All of the GUI design patterns described above apply to tables as well as lists. Listing 2 shows an implementation using JTable and TableModel instead of JList and ListModel. "The Java Look and Feel Guidelines" recommend only one table column in the original list and multiple columns in the chosen list. As shown in Listing 2, this can be done by disabling the header and grid lines in the original table, and by assigning the preferred viewport size based on TableColumn widths.

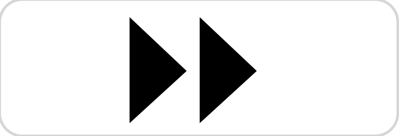


**Figure 7** "Add All" icon

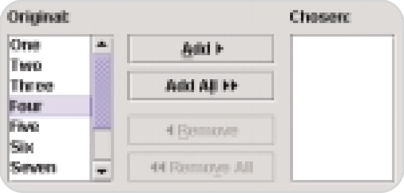


**Figure 8** ListChooser with unavailable actions disabled

The GridBagLayout code is omitted from Listing 2 as it is similar to the simpler variation of Listing 1. If "Move Up" and "Move Down" buttons are provided, the GridBagLayout contains seven rows instead of five. Following "The Java Look and Feel Guidelines," the spacing between these optional buttons is six pixels greater than between the "Add" and "Remove" buttons.

The ActionListener implementation is slightly different in Listing 2 because of the use of TableModels instead of ListModels. The code in Listing 2 produces the GUI shown in Figure 2.

Details of the JTable and TableModel can be found in *The JFC Swing Tutorial* (by Kathy Walrath and Mary Campione), along with descriptions of the JList and ListModel. The JTable is more complex than the JList, but when displaying long lists they share many of the same performance issues. A future article will discuss these issues and how to optimize performance for this common GUI design pattern.

## Conclusion

The Add-and-Remove pattern enables users to choose multiple objects from long lists. This standard GUI design pattern improves usability by easing the user's learning curve. Standard spacing, directional icons, and button disabling all reduce the users' efforts, enabling them to accomplish their tasks easily in a professional user interface.

## Resources

- Aube, S. (2000). "A Dual Listbox Selection Manager": www.codeguru.com/Cpp/controls/listbox/article.php/c4755
- Walrath, K., and Campione, M. (2004). *The JFC Swing Tutorial: A Guide to Constructing GUIs*. Addison-Wesley Professional: http://java.sun.com/docs/books/tutorial/uiswing/components/
- "Official Guidelines for User Interface Developers and Designers." Microsoft Inc. (2004): http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnwue/html/welcome.asp
- Sun Microsystems Inc. (2001). *Java Look and Feel Design Guidelines, Second Edition*. Addison-Wesley *Professional:* http://java.sun.com/products/jlf/ed2/book/HIG.Misc.html
- Sun MicroSystems Inc. (2002). *Java Look and Feel Design Guidelines: Advanced Topics*. Addison-Wesley *Professional:* http://java.sun.com/products/jlf/at/book/Idioms6.html
- Weinschenk, S., Jamar, P., and Yeo, S. (1997). *GUI Design Essentials*. John Wiley & Sons.

*Listing 1*

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;

/**
 * The ListChooser implements the "Add-and-Remove"
 * GUI design pattern.
 */
public class ListChooser extends JPanel
   implements ActionListener
{
   /** @serial  Original list. */
   private JList _originalList = null;

   /** @serial  Chosen list. */
   private JList _chosenList = null;

   /** @serial  Original label. */
   private JLabel _originalLabel = null;

   /** @serial  Chosen label. */
   private JLabel _chosenLabel = null;

   /** @serial  "Add" button. */
   private JButton _add = null;

   /** @serial  "Remove" button. */
   private JButton _remove = null;

   /** @serial  "Add All" button. */
   private JButton _addAll = null;

   /** @serial  "Remove All" button. */
   private JButton _removeAll = null;

   /**
    * Constructor.
    * <p>
    * @param  original  original objects
    * @param  chosen    chosen objects
    */
   public ListChooser( Object[] original, Object[] chosen )
   {
      // Create Lists and ListModels.
      super();
      setFocusable( true );
      DefaultListModel originalModel =
         new DefaultListModel();
      DefaultListModel chosenModel =
         new DefaultListModel();
      for( int i = 0; ( i < original.length ); i++ )
         originalModel.addElement( original[ i ]);
      for( int i = 0; ( i < chosen.length ); i++ )
         chosenModel.addElement( chosen[ i ]);
      _originalList = new JList( originalModel );
      _chosenList = new JList( chosenModel );

      // Place lists.
      setLayout( new GridBagLayout());
      GridBagConstraints c = new GridBagConstraints();
      c.fill = GridBagConstraints.BOTH;
      c.weightx = 1.0;
      c.weighty = 1.0;
      c.gridwidth = 1;
      c.gridheight = 4;
      c.gridy = 1;
      c.gridx = 0;
      JScrollPane pane = new JScrollPane( _originalList );
      _originalList.setVisibleRowCount( 7 );
      pane.setPreferredSize( new Dimension( 80, _originalList.
         getPreferredScrollableViewportSize().height ));
      add( pane, c );
      c.gridx = 2;
      pane = new JScrollPane( _chosenList );
      _chosenList.setVisibleRowCount( 7 );
      pane.setPreferredSize( new Dimension( 80, _chosenList.
```

InetSoft
open standards innovation

# Style Report 6

## Solid Building Blocks for Your BI Powerhouse

OLAP

Analytics

Ad-Hoc Reports

Interactive Reports

Production Reports

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

**Challenges:**

Too costly to start off and expand

Too many UI to train

Too complex to manage and support

**Solutions:**

One step at a time, at your own pace

One zero-client user interface

One web application

## One Integrated, Open Architecture Solution

JAVA DEVELOPERS JOURNAL JDJ READERS' CHOICE AWARD

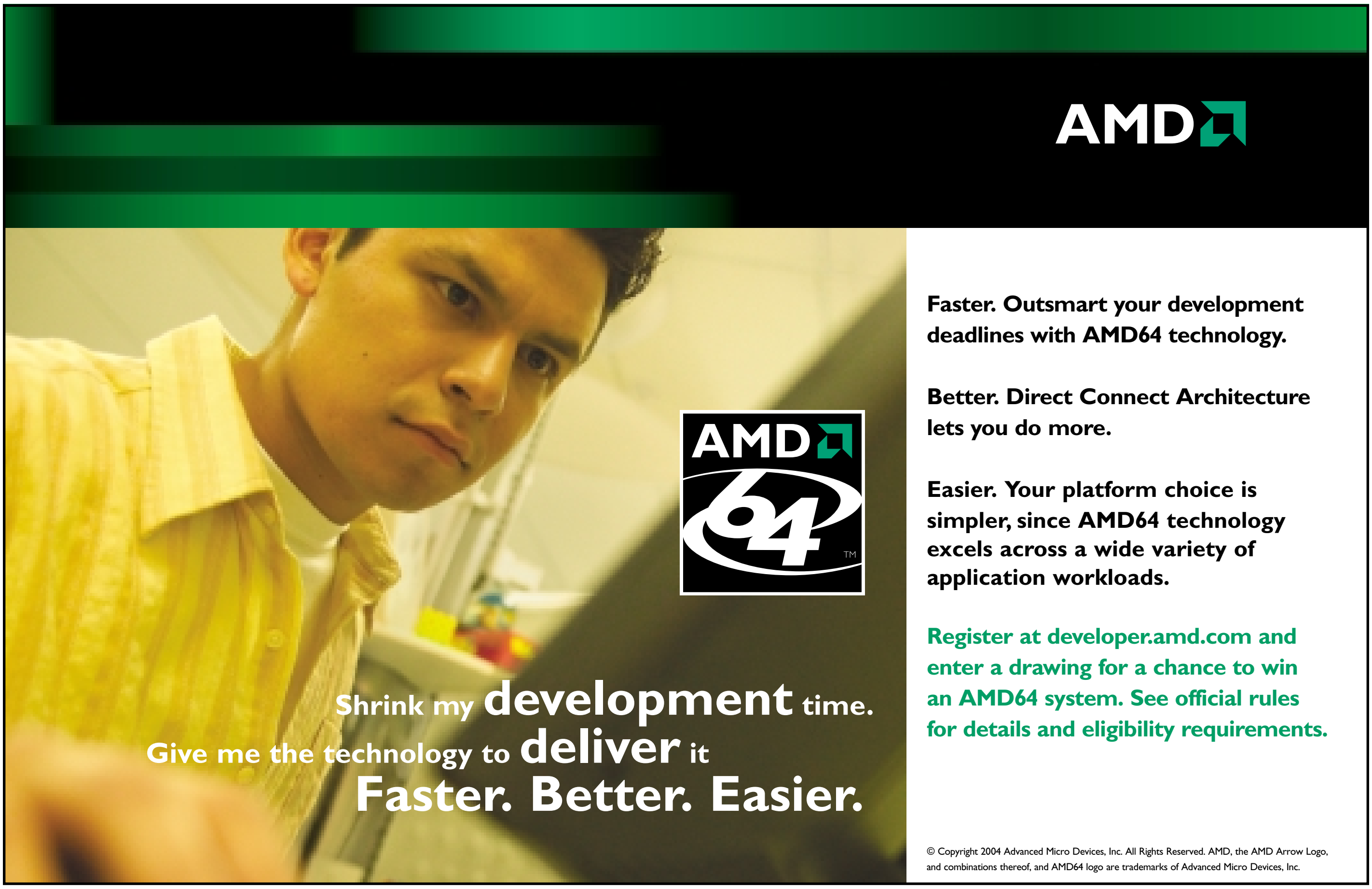
For more information and to download a free evaluation copy .......................... www.inetsoft.com/jdj

InetSoft, Style Report, Style Report/ Enterprise Edition are trademarks or registered trademarks of InetSoft Technology Corp. Java, and Sun are trademarks or registered trademarks of SUN Microsystems, Inc. PDF is a trademark or registered trademark of Adobe Systems, Inc. Excel and Word are trademarks or registered trademarks of Microsoft Corporation.

```
         getPreferredScrollableViewportSize().height ));
      add( pane, c );

      // Place labels.
      _originalLabel = new JLabel( "Original:" );
      _chosenLabel = new JLabel( "Chosen:" );
      c.weighty = 0.0;
      c.gridheight = 1;
      c.gridy = 0;
      c.gridx = 0;
      add( _originalLabel, c );
      c.gridx = 2;
      add( _chosenLabel, c );

      // Place buttons.
      _add = new JButton( "Add" );
      _add.setMnemonic( 'A' );
      _addAll = new JButton( "Add All" );
      _remove = new JButton( "Remove" );
      _removeAll = new JButton( "Remove All" );
      c.weightx = 0.0;
      c.weighty = 1.0;
      c.gridx = 1;
      c.gridy = 1;
      c.anchor = GridBagConstraints.SOUTH;
      add( _add, c );
      c.gridy = 3;
      add( _remove, c );
      c.anchor = GridBagConstraints.NORTH;
      c.gridy = 2;
      add( _addAll, c );
      c.gridy = 4;
      add( _removeAll, c );

      // Attach listeners.
      _add.addActionListener( this );
      _addAll.addActionListener( this );
      _remove.addActionListener( this );
      _removeAll.addActionListener( this );
   }

   /**
    * Action event listener.
    * <p>
    * @param  e  ActionEvent
    */
   public void actionPerformed( ActionEvent e )
   {
      // Get models and command.
      DefaultListModel originalModel =
         _originalList.getModel();
      DefaultListModel chosenModel =
         _chosenList.getModel();
      String command = e.getActionCommand();

      // Add selected objects to the chosen list...
      if( command.equals( _add.getText()))
      {  Object[] values =
            _originalList.getSelectedValues();
         for( int i = 0; ( i < values.length ); i++ )
         {  chosenModel.addElement( values[ i ]);
            originalModel.removeElement( values[ i ]);
         }
      }

      // ...or add all objects to the chosen list...
      else if( command.equals( _addAll.getText()))
      {  Object[] values = originalModel.toArray();
         for( int i = 0; ( i < values.length ); i++ )
            chosenModel.addElement( values[ i ]);
         originalModel.clear();
      }

      // ...or remove selected objects to original list...
      else if( command.equals( _remove.getText()))
      {  Object[] values =
            _chosenList.getSelectedValues();
         for( int i = 0; ( i < values.length ); i++ )
         {  originalModel.addElement( values[ i ]);
            chosenModel.removeElement( values[ i ]);
         }
      }

      // ...or remove all objects to the original list...
      else if( command.equals( _removeAll.getText()))
      {  Object[] values = chosenModel.toArray();
         for( int i = 0; ( i < values.length ); i++ )
            originalModel.addElement( values[ i ]);
         chosenModel.clear();
      }
   }
}
```

*Listing 2*

```
import java.awt.*;
import java.awt.event.*;
import java.util.Vector;
import javax.swing.*;
import javax.swing.event.*;
import javax.swing.table.*;

/**
 * The TableChooser implements the "Add-and-Remove" GUI
 * design pattern and displays the chosen list as a table.
 */
public class TableChooser extends JPanel
   implements ActionListener
{
   /** @serial  Original table. */
   private JTable _originalTable = null;

   /** @serial  Chosen table. */
   private JTable _chosenTable = null;
      .
      .
      .
   /**
    * Constructor.
    * <p>
    * @param  original     original objects
    * @param  chosen       chosen objects
    * @param  columnNames  column names
    */
   public TableChooser( Object[][] original,
      Object[][] chosen, Object[] columnNames )
   {
      // Create Tables and TableModels.
      super();
      setFocusable( true );
      DefaultTableModel originalModel =
         new DefaultTableModel( original, columnNames );
      DefaultTableModel chosenModel =
         new DefaultTableModel( chosen, columnNames );
      _originalTable = new JTable( originalModel );
      _originalTable.setTableHeader( null );
      _originalTable.setShowGrid( false );
      _chosenTable = new JTable( chosenModel );

      // Place tables in scroll panes of appropriate width.
      TableColumnModel tcm =
         _originalTable.getColumnModel();
      TableColumn tc = tcm.getColumn( 0 );
      _originalWidth = tc.getPreferredWidth();
      tc.setMinWidth( _originalWidth );
      _originalTable.setPreferredScrollableViewportSize(
         new Dimension( _originalWidth,
         10 * _originalTable.getRowHeight()));
      _chosenTable.setPreferredScrollableViewportSize(
         new Dimension( tcm.getTotalColumnWidth(),
         10 * _chosenTable.getRowHeight()));
      JScrollPane originalPane =
         new JScrollPane( _originalTable );
      originalPane.getViewport().setBackground( Color.white );
      originalPane.setVerticalScrollBarPolicy(
         JScrollPane.VERTICAL_SCROLLBAR_ALWAYS );
      JScrollPane chosenPane =
         new JScrollPane( _chosenTable );
      chosenPane.getViewport().setBackground( Color.white );
      chosenPane.setVerticalScrollBarPolicy(
         JScrollPane.VERTICAL_SCROLLBAR_ALWAYS );
         .
         .
         .
   }

   /**
    * Action event listener.
    * <p>
    * @param  e  ActionEvent
    */
   public void actionPerformed( ActionEvent e )
   {
      // Get models and command.
      DefaultTableModel originalModel =
         _originalTable.getModel();
      DefaultTableModel chosenModel =
         _chosenTable.getModel();
      String command = e.getActionCommand();

      // Add selected objects to the chosen table...
      if( command.equals( _add.getText()))
      {  int[] indices = _originalTable.getSelectedRows();
         Vector v = originalModel.getDataVector();
         for( int i = 0; ( i < indices.length ); i++ )
            chosenModel.addRow(
               ( Vector )v.elementAt( indices[ i ]));
         for( int i = indices.length - 1; ( i >= 0 ); i-- )
            originalModel.removeRow( indices[ i ]);
      }

      // ...or add all objects to the chosen table...
      else if( command.equals( _addAll.getText()))
      {  Vector v = originalModel.getDataVector();
         for( int i = 0; ( i < v.size()); i++ )
            chosenModel.addRow(
               ( Vector )v.elementAt( i ));
         originalModel.setRowCount( 0 );
      }

      // ...or remove selected objects to original table...
      else if( command.equals( _remove.getText()))
      {  int[] indices = _chosenTable.getSelectedRows();
         Vector v = chosenModel.getDataVector();
         for( int i = 0; ( i < indices.length ); i++ )
            originalModel.addRow(
               ( Vector )v.elementAt( indices[ i ]));
         for( int i = indices.length - 1; ( i >= 0 ); i-- )
            chosenModel.removeRow( indices[ i ]);
      }

      // ...or remove all objects to the original table.
      else if( command.equals( _removeAll.getText()))
      {  Vector v = chosenModel.getDataVector();
         for( int i = 0; ( i < v.size()); i++ )
            originalModel.addRow(
               ( Vector )v.elementAt( i ));
         chosenModel.setRowCount( 0 );
      }
   }
}
```


AMD
AMD 64 TM
Shrink my development time.
Give me the technology to deliver it
Faster. Better. Easier.

# Medical Image Conversion

## Converting to supported formats

by Yayati Kasralikar & Pramod Jain

*In this article we'll discuss the conversion of image formats using Sun's Java Image I/O (ImageIO) and the NIH ImageJ APIs. The image formats of interest are DICOM, JPEG 2000, PNG, and TIFF. These formats are widely used in medical applications; however, most of the disciplines in medicine are standardizing on DICOM. A single API, ImageIO or ImageJ, supports the reading and writing of limited image formats. We'll describe the use of these two APIs to support the reading and writing of a larger number of image formats as well as the process of extracting rich metadata from DICOM files.*

In general, medical image files contain the following information:

- *Image data,* which may be in three dimensions and may be raw/unmodified or compressed.
- *Header data,* which contains metadata such as patient information, image information, and equipment information. This information may be stored in different formats using a variety of standards – DICOM and JPEG 2000 standards are described below.

The Digital Imaging and Communications in Medicine (DICOM) standard was created by the National Electrical Manufacturers Association (NEMA) to standardize the imaging and communication format across multiple manufacturers to aid the distribution and viewing of medical images, such as CT scans, MRIs, and ultrasound. DICOM is an industry standard supported by most of the major medical imaging equipment manufacturers. It is a complex standard consisting of many parts, which are specified in the NEMA documents and can be found at http://medical.nema.org/dicom/2003.html.

DICOM image compression can be lossy or lossless variants of the JPEG and JPEG 2000 formats, as well as a lossless Run-Length Encoding format.

JPEG 2000 is the successor of the JPEG standard and offers a better quality wavelet-based compression. It supports both lossy and lossless compression. JPEG 2000 is becoming a popular image format in image servers and medical imaging.

There are various image-conversion tools and software available, but most of them support a limited number of formats. It's often difficult or impossible to customize or integrate these tools with existing applications. ImageIO, Sun's Advanced Imaging API, (JAI), offers an open and extensible framework for image conversion. It supports the reading and writing of many formats such as TIFF, JPEG, and JPEG 2000. At present, it does not support the reading or writing of DICOM and PGM formats, two popular medical formats. ImageJ, a project funded by the National Institutes of Health (NIH), is another open and extensible API for image conversion. It supports the reading of DICOM, PGM, and other formats; however, ImageJ does not support formats such as JPEG 2000.

### ImageIO API

The Image I/O library is one of the standard APIs of the J2SE 1.4 release. The javax.imageio package contains the following classes that are useful for the current application:

- ***ImageReader:*** An abstract superclass for parsing and decoding images
- ***ImageWriter:*** An abstract superclass for encoding and writing images
- ***ImageIO:*** A class containing static convenience methods for locating ImageReaders and ImageWriters and performing simple encoding and decoding
- ***ImageReadParam and ImageWriteParam:*** Classes that describe decoding and encoding, respectively, of data streams

The set of supported image formats is not fixed. By default, the javax.imageio package can read GIF, PNG, and JPEG images and can write PNG and JPEG images. Calling the following static methods of the ImageIO class retrieves the complete list of available readable and writeable formats:

- ImageIO.getReaderFormatNames()
- ImageIO.getWriterFormatNames()

Sun provides JAI Image I/O Tools to support image readers/writers (also called codecs) in addition to the default formats supported in the Image I/O libraries of J2SE 1.4.

JAI Image I/O Tools can be downloaded from http://java.sun.com/products/java-media/jai/downloads/download-iio.html. The readers available with the Image I/O tools are for TIFF, JFIF, WBMP, JPEG-LOSSLESS, JPEG2000, GIF, RAW, BMP, JPEG, PNM, and PNG formats. The writers available are for all the above formats except GIF.

### ImageJ API

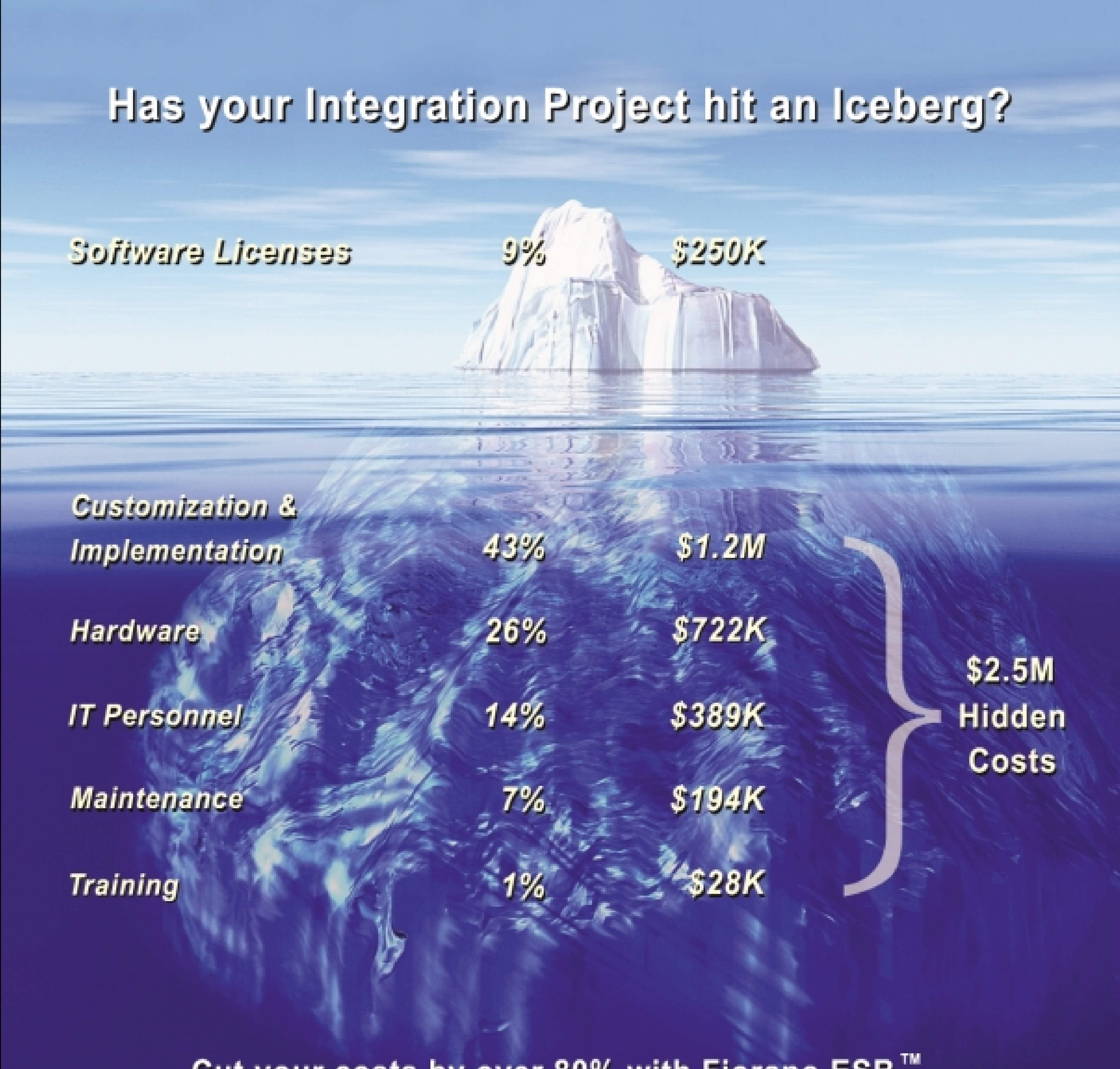
ImageJ is a public domain image processing API in Java, funded by the NIH. It can be downloaded from http://rsb.in

**Yayati Kasralikar** is lead programmer at Innovative Decision Technologies, Inc. (INDENT). He holds an MS from the University of Central Florida, Orlando.

yayati@indent.org

**Pramod Jain** is president, Innovative Decision Technologies, Inc. (INDENT). INDENT has built several large-scale Java-based collaboration portals. Pramod has a PhD from the University of California, Berkeley.

pramod@indent.org



fo.nih.gov/ij/download.html.

It has classes to display, edit, analyze, process, save, and print 8-bit, 16-bit, and 32-bit images. It can read many image formats including TIFF, GIF, JPEG, BMP, DICOM, FITS, PGM, and RAW. It can write image formats such as JPEG, GIF, TIFF, BMP, RAW, etc. Note that some readers, such as DICOM, FITS, and PGM, are not available with ImageIO and some of the writers, such as JPEG 2000 and JPEG-LOSSLESS, that are available with ImageIO are not available with ImageJ.

New plugins can be developed using the ImageJ API to support additional formats, for instance, encoding in JPEG 2000 and JPEG-Lossless formats. A detailed tutorial for writing new plugins can be found at http://mtd.fh-hagenberg.at/depot/imaging/imagej/. This article describes an alternate method that relies on integrating ImageJ and ImageIO instead of extending or writing plugins for ImageJ. As will be evident, this is a simpler and less time-consuming approach because it does not require writing new encoders.

> "There are various image-conversion tools and software available, but most of them support a limited number of formats"

ImageJ supports a wide variety of features such as geometric operations, editing, image enhancements, and analysis. For the purpose of a conversion tool, only input/output (ij.io) and plugin (ij.plugin) packages are used. It contains the following classes and packages that are useful for the current application:

- ***ij.io:*** A package that contains classes for reading/decoding and writing/encoding image files.
- ***ij.io.Opener:*** A convenient class used to open and read a supported image.
- ***ij.plugin***: A package that contains the encoder and decoder classes.
- ***ij.ImagePlus:*** A class that represents an image in ImageJ. It has methods for getting the AWT image and image dimension. Using the AWT image object, a BufferedImage object can be constructed.

## Description of Conversion Code

ImageJ and ImageIO each support various image formats and either of them can be used to develop an image conversion program. However, neither supports the conversion of some important image formats, for instance, the DICOM format cannot be converted to either JPEG-LOSSLESS or JPEG 2000. We will use the two APIs to develop an application to support the conversion from and to a larger set of image formats, particularly the conversion of DICOM to JPEG 2000. The process for image conversion is shown in Figure 1.

The algorithm and high-level steps are as follows:

1. Given an image, check whether it's in a format readable by ImageIO. If yes, create BufferedImage using the ImageIO read method.
2. If ImageIO does not have a reader for that format, try to open using ImageJ and create BufferedImage.
3. If both ImageIO and ImageJ do not have a reader to open the image, report that the image cannot be processed.
4. Use the ImageIO write method to convert the image to a specified format.
5. If ImageIO does not support the writer for the specified output format, report that the image cannot be converted.

The code outline corresponding to these steps is shown in Listing 1. This conversion code outline can handle new formats that may be supported by ImageJ and ImageIO in the future, that is, new formats can be incorporated without the need to change the code because both of these APIs provide extensibility for future compatibility. Both ImageIO read and ImageJ Opener class are capable of reading any format supported by the corresponding APIs.

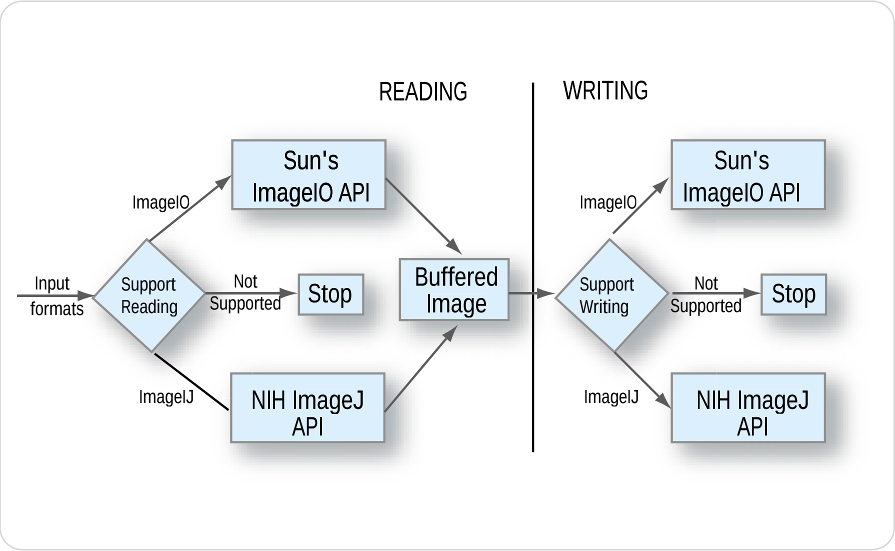


**Figure 1** Image conversion process

## Reading Metadata from DICOM Images

The DICOM file contains image data as well as image metadata such as patient information, image information, equipment information, etc. This metadata is stored in the file header. Each metadata field has a code associated with it, for instance, the patient name field has a code of (0010,0010). An outline of the program for extracting metadata is shown in Listing 2. Code corresponding to the metadata fields of interest are stored in the dicomKeys array. Information corresponding to this code is retrieved and stored in a hashtable.

## Future Enhancements

- The metadata extraction code should be extended to store the metadata informa-

tion in XML files or in a database.
- The DICOM writer/encoder should be supported. Writing a DICOM encoder as a new plugin in ImageJ can achieve this.
- Retrieving and storing metadata from images with formats other than DICOM should be supported.

## Conclusion

The ImageJ API from the NIH is used with the ImageIO API from Sun to convert formats of medical images. The approach presented increases the number of supported medical image formats for conversion, without writing new encoders/decoders. The conversion code may be downloaded from www.indent.org/jdj/imageConversion/htm.

## References

1. *Medical Image Format FAQ:* www.dclunie.com/medical-image-faq/html/
2. *Advanced Imaging Image I/O API RC 1.0:* http://java.sun.com/developer/technicalArticles/Media/AdvancedImage/
3. *ImageJ Homepage:* http://rsb.info.nih.gov/ij/
4. *DICOM Homepage:* http://medical.nema.org/

### Listing 1

```
Iterator iter = ImageIO.getImageReadersBySuffix(fileExt);
if(iter.hasNext()) {
// Input format is readable by ImageIO. Use ImageIO to read it.
BufferedImage bi = ImageIO.read(new File(inFileName));
}
else {  // Try opening the input image using ImageJ
       try {
      Opener o = new Opener();
      ImagePlus imp = o.openImage(inFileName);
      BufferedImage bi = new BufferedImage(imp.getWidth(),
      imp.getHeight(), getType(imp));
      Graphics g = bi.createGraphics();
      g.drawImage(imp.getImage(), 0, 0, null);
      g.dispose();
      }
       catch(Exception e) {
System.out.println(“ImageIO and ImageJ both cannot read the input
image”);
             }
}
ImageIO.write(bi, outFormat, outputFile); // Use ImageIO to write
to output format
```

### Listing 2

```
String s = (String) imp.getProperty("Info"); //imp is the
ImagePlug object
Hashtable ImageMetaData = new Hashtable();
for (int i=0;i< dicomKeys.size();i++) {
String value =
s.substring(s.indexOf(dicomKeys.get(i).toString()));
ImageMetaData.put(dicomKeys.get(i).toString(),value.substring
  (value.indexOf(": ")+2,value.indexOf("\n")).trim());
}
```

# JDJ Editors' Choice **Awards**

*The editors of JDJ are in a unique position when it comes to Java development. All are active coders in their "day jobs," and they have the good fortune of getting a heads up on many of the latest and greatest software releases. They were asked to choose three products from the last 12 months that they felt had not only made a major impact on their own development, but also on the Java community as a whole.*

*The following is a list of each editor's selections and the reason why they chose that product.*

## Joe Ottinger

*Editor-in-Chief*

### WSAD 5.1

WebSphere Studio Application Developer is a comprehensive integrated development environment for visually designing, constructing, testing, and deploying Web services, portals, and J2EE applications. www-306.ibm.com

*WSAD 5.1 is IBM's beefed-up version of Eclipse, meant to put J2EE development into overdrive. While it has some odd quirks and is primarily aimed at WebSphere development and lock-in, it definitely shows off what Eclipse can be, given time and resources, and the WAS integration certainly enables rapid development on the platform. On the whole, I've been very impressed with WSAD 5.1 when compared to the Eclipse platform, and I find its quirks quite acceptable.*

### JUnit

JUnit is a regression testing framework written by Erich Gamma and Kent Beck. It's used by the developer who implements unit tests in Java. JUnit is open source software released under the Common Public License version 1.0 and hosted on SourceForge. www.junit.org

*Even if you don't buy into the entire test-driven development paradigm, it's very difficult to argue what a useful tool JUnit has proven to be in terms of providing verifiability for Java. It's been used everywhere, from system and unit testing onward. I'd be honestly surprised at any major project in use today that hasn't been using JUnit or something influenced by it internally.*

### Borland Optimizeit

Borland Optimizeit Profiler for Java supports the development of fast, scalable, and reliable applications, tracking memory and CPU performance issues throughout the development life cycle. www.borland.com

*Optimizeit is a profiling tool from Borland, and in my opinion is excellently made. I find it very useful for zeroing in on performance issues in code, and producing artifacts that can be used to track performance over time. While there are other profilers in popular usage, I find that Optimizeit suits my needs best at the moment.*

## Kirk Pepperdine

*Java Enterprise Editor*

### IntelliJ IDEA 4.0 from JetBrains

IntelliJ IDEA is an intelligent Java IDE intensely focused on developer productivity. It provides a robust combination of enhanced development tools including refactoring, J2EE support, Ant, JUnit, and CVS integration. www.jetbrains.com/idea/index.html

*Earlier this year I commented on a discussion in which people were passionately speaking for their choice of IDE. Since the discussion centered on Eclipse and NetBeans, I didn't speak of IntelliJ. Instead I noted that a developer's relationship with an IDE is a very personal one. Now it's true that many (if not all) of the features found in IntelliJ are present in many of the other IDEs. That said, beyond a certain subset of features that are used 99% of the time, the others become curiosities. What is more interesting is the way in which these features are presented to the user. In real terms, how much time do I spend thinking about the mechanics of using the IDE as apposed to actually thinking about what I'm trying to achieve. If this were the only criterion, then IntelliJ would still be a great product. What really puts this product over the top is that it does not interfere with my ability to structure a project the best way I see fit as some other IDEs unfortunately do. It is these combinations of usability (or noninterference of my thoughts) that make IntelliJ IDEA 4.0 my choice of IDE.*

### HP JTune

HP JTune is a core component of the first generation of the reference platform Itanium 2 processor - HP-UX version that bundles HP-UX 11i v1.6 with essential HP-UX tools, compilers, and technical documentation into a single source to speed your next development project. www.hp.com

*JTune is a tool that is offered freely by HP. It offers its users a simple visualization of the logs produced by garbage collection when the –verbose:gc flag has been set. Though there is enough information in logs to diagnose many memory utilization problems, much of that information must be massaged out of*

# Royalties Fees are a Thing of the Past

Paying OEM royalty fees to application adapter vendors is now a thing of the past. Librados offers application adapters on a low-cost royalty free source code basis. This along with Librados' pure Java, J2EE standards-based architecture eliminates the proprietary vendor lock-in that has been a key factor in the high cost and limited adoption of old fashioned adapter solutions.

Librados delivers over fifty adapters for enterprise information systems such as SAP R/3, PeopleSoft, Siebel, Oracle Applications, JD Edwards, file systems, databases, and mainframe application environments. These adapters can expose back-end enterprise system functionality in a wide variety of formats, including XML and Web services.

LIBRADOS

FREEDOM TO INTEGRATE

SAP Certified Integration

Call 888 341 4258 or visit www.librados.com

*the raw data. JTune not only does a wonderful job of culling this information, it does a nice job of presenting the data in an understandable format. What makes this tool even more interesting is that if an HP JVM has produced the logs, you can use that data to run a simulation under different memory configurations. I have found the results of these simulations to be a fairly reliable prediction of how a particular configuration will perform in production. In this editor's humble option, JTune is an essential tool to have when you are trying to resolve memory problems.*

### Ant

Apache Ant is a Java-based build tool. In theory, it's kind of like Make but without Make's wrinkles. Instead of a model where it is extended with shell-based commands, Ant is extended using Java classes. Instead of writing shell commands, the configuration files are XML-based, calling out a target tree where various tasks get executed. Each task is run by an object that implements a particular Task interface.
http://ant.apache.org/

*What is there to say about Ant that has not already been said. It is YAT (yet another tool) that has proven itself time and time again. Ant has proven that you can take the most horrible activity you could describe, scripting in XML, and make it look like a work of art.*

## Karl Avedal

*Core and Internals Co-editor*

### IntelliJ IDEA 4.0 from JetBrains

IntelliJ IDEA is an intelligent Java IDE intensely focused on developer productivity. It provides a robust combination of enhanced development tools including refactoring, J2EE support, Ant, JUnit, and CVS integration.
www.jetbrains.com/idea/index.html

*IDEA is simply the best development environment I have ever used. It meant a paradigm shift for my development when it arrived.*

### YourKit Java Profiler

YourKit Java Profiler is a new generation of profiling tools. With the help of YourKit Java Profiler, you can easily solve a wide range of CPU and memory-related performance problems.
www.yourkit.com/home/index.jsp

*For some reason, I really like Java profilers. I love JProfiler and I've used and liked Optimizeit a lot in the past. But this time I'm going to select YourKit Profiler. It still has quite a few bugs, but has the potential to fundamentally change how and when you profile your installed applications.*

### Java HotSpot Virtual Machine

The Java HotSpot Virtual Machine is a key component in maximizing deployment of enterprise applications. It's a core component of J2SE software, supported by leading application vendors and technologies. The Java HotSpot VM supports virtually all aspects of development, deployment, and management of corporate applications.
http://java.sun.com

*It's sort of ironic. When I first started using Java the biggest problem was the JVM. I loved the language, but the JVM was incredibly slow and easy to crash. After many years of trying to make a better JVM, Sun eventually succeeded, with help from competitors like IBM and JRockit making good JVMs that in many cases outperformed HotSpot. Today I would say that the JVM is one of the best things about the Java platform. It's now so good that new languages, like Groovy, choose Java as their standard runtime platform.*

## Joe Winchester

*Desktop Java Editor*

### Swing 1.4.2 Release

The Swing toolkit is a fully featured UI component library implemented entirely in the Java programming language. The J2SE platform's javax.swing package uses the windowing functionality of AWT and the rendering capabilities of the Java 2D API to provide sophisticated and highly extensible UI components that comply with the JavaBeans specification.
http://java.sun.com

*The 1.4 performance improvements were great, and then with the inclusion of the XP look and feel in 1.4.2 this has been a huge rebirth for Swing. I get great feedback from customers and it's a huge boost toward helping Java remain and grow on the client.*

### Eclipse 3.0

Eclipse is an open platform for tool integration built by an open community of tool providers. Operating under an open source paradigm, with a common public license that provides royalty-free source code and worldwide redistribution rights, the Eclipse platform provides tool developers with flexibility and control over their software technology.
www.eclipse.org

*The M8 release is the one I'm using now and it's just incredible. The Java development tooling continues to impress me more and more each release. I was at EclipseCon in February in Anaheim and I felt this was the start of something very big. There is a lot of optimism and enthusiasm from developers across a wide set of companies and institutions, and a lot of the big names in software are moving toward Eclipse as being their foundation for tooling and beyond.*

### SWT

Standard Widget Toolkit is the software component that delivers native widget functionality for the Eclipse platform in an operating system–independent manner. It's analogous to AWT/Swing in Java with a difference – SWT uses a rich set of native widgets.
www.eclipse.org

*The Standard Widget Toolkit provides a rich, native cross-platform GUI toolkit. Recent improvements have included Swing interoperability, a control that uses the platform browser on each operating system (IE on Windows, Mozilla on Linux), as well as irregular-shaped windows, toolbar integration, and support for large datasets on trees.*

## Jason Bell

*Contributing Editor*

### Eclipse 3.0

Eclipse is an open platform for tool integration built by an open community of tool providers. Operating under an

open source paradigm, with a common public license that provides royalty-free source code and worldwide redistribution rights, the Eclipse platform provides tool developers with flexibility and control over their software technology.
www.eclipse.org

*It's been interesting watching this IDE mature over the east 12 months. There's still some way to go for the team but every release has some very cool features in it.*

### Jtest5 from Parasoft

Jtest is an Automated Error Prevention product that automates Java unit testing and coding standard compliance to help developers produce reliable code in record time. Jtest analyzes classes, then generates and executes JUnit-format test cases designed to achieve maximum coverage, expose uncaught runtime exceptions, and verify requirements that were expressed using Design by Contract.
www.parasoft.com

*This code coverage and testing tool has saved my programming life many times over. Autogenerating JUnit tests create a much more robust solution than any I could possibly put together. It costs a lot but I believe this money will be recouped in less than 12 months.*

### jEdit 4

jEdit is a mature and well-designed programmer's text editor that has been in development for over five years. The core of jEdit is primarily developed by Slava Pestov, and jEdit plugins are being written by a large and diverse team of programmers from around the world.
www.jedit.org

*Another product that gets better from release to release. I've been doing a lot of JSP work with jEdit and it has yet to let me down. Once again it's the community effort with the plugins that makes the icing on the cake.*

## Calvin Austin

*Core and Internals Co-editor*

### Project Looking Glass

In the Project Looking Glass prototype, windows displaying applications are no longer stacked on each other with flat icons and buttons to represent them; they are viewed in a 3D environment and manipulated as 3D objects. Project Looking Glass is moving beyond the boundaries of old environments to revolutionize the use of the desktop.
wwws.sun.com/software/looking_glass/

*It is rare to find a product that grabs your attention like Project Looking Glass. Zooming into a mountain range to reach your desktop files, rotating stacks of windows with a wave of a mouse, and scribbling notes on the back of your browser – all simple and yet intuitive tasks. However, when you learn that this is all the work of a Java application, then you can't help but smile. Java on the desktop is back.*

### Tomcat 5.0

Tomcat is the servlet container that's used in the official Reference Implementation for the Java Servlet and JavaServer Pages technologies. The Java Servlet and JavaServer Pages specifications are developed by Sun under the Java Community Process.
http://jakarta.apache.org/tomcat/

*Tomcat needs no introduction and is the well-respected workhorse of many Java application environments. The 5.0 release builds on that reputation with support for JavaServer Pages 2.0, the Java Servlet 2.4 API, and some nice performance improvements. If that wasn't reason enough to try the download, you can now monitor your faster servlets using JMX.*

### *The Java Developer's Guide to Eclipse*

Shavor, S., et al. (2003). Addison-Wesley.

Drawing on their considerable experience teaching Eclipse and mentoring developers, the authors provide guidance on how to customize Eclipse for increased productivity and efficiency and how to avoid common pitfalls.
www.aw-bc.com

*Books about open source projects have a mixed record. Many suffer by being out of date by the time they are published or skim over difficult areas.* Java Developer's Guide to Eclipse *sets that bar high. It's easy to read, not afraid to tackle the more complex areas, and, although it's based on Eclipse 2.0, has many examples still relevant for current builds.*

# JReport

## by Jinfonet Software, Inc.

Reviewed by
Dave Glende

**Dave Glende** is the vice president of strategy and chief technology officer for Unify. In his 19 years with Unify, Dave has guided Unify's product strategy from its beginnings in relational database management systems, to multitier distributed systems, to application development tools, and today information delivery and management solutions. His expertise lies in information delivery systems, process-centric computing solutions, and Web application development. Glende holds a BS in computer science from California State University, Sacramento.

*dmg@unify.com*

The value of a single application platform that connects people, information, and systems, both within and across organizational boundaries, is self-evident. The critical component to developing such a powerful enterprise solution is selecting the right business-process automation tools that can deliver on the promise of a connected enterprise.

At Unify, we recognized the value and challenges presented by the development and distribution of an integrated application platform. Because decision-makers at all levels of an organization depend on timely, customized reports to provide actionable business information, choosing the right information delivery component for this platform was critical. For the Unify NXJ application platform, we needed a reporting solution that was easy enough for all types of developers to install, deploy, and manage; powerful enough to support business; and open enough to integrate into any IT environment.

JReport was a good fit for Unify NXJ because it is a 100% J2EE solution that delivers highly usable business information without the complexities of integration, management, and usage inherent in traditional analytical reporting solutions. In this article, we'll look at how JReport meets the challenges of enterprise information delivery.

### Actionable Information

The key to JReport is the ability to provide actionable information through Web-based report deployment and ad hoc report creation. JReport provides a wide range of functionality to dynamically modify report views via the Web. Data can be sorted, filtered, grouped, navigated, searched, and drilled-down on to view detailed information. JReport allows users to customize their view of data in order to glean valuable information.

Creating ad hoc reports is simple, making JReport a highly usable tool across the enterprise. A wizard prompts users through the process of selecting a data source, choosing a report type (JReport offers over 40 chart types, tabular reports, and crosstabs), defining fields to display, and setting grouping and sorting levels. Parameterized reports can be created by applying filters on the generated result. The ease and flexibility of creating reports on the fly is an important element of Unify NXJ.

### Easy, Open, and Flexible Report Design

JReport is an open system that offers a high degree of creative flexibility. JReport presents an interactive design environment that puts a host of report design features at your fingertips. Our developers were able to start designing report templates for our customers quickly, and a handy "View" option allowed them to see the actual report as it was being built. This helped speed the process of designing and testing the report templates that were included in NXJ or built by Unify Professional Services for customer-specific solutions.

Another nice feature of JReport is the Report Inspector, which displays each report object on a hierarchy tree in a separate window. Clicking on the object, either from within the layout interface or within the Report Inspector, displays properties and values for that object, which can then be edited in point-and-click fashion. A property can be either a static value or a Java function.

Users can also define style sheets to streamline formatting of font size, typeface, alignment, and other parameters, allowing them to design more reports faster, while also making it easier to implement a standardized look. In addition, custom objects can be created by using an API to convert any Java class into a JReport object that can be seamlessly integrated into any report.

Jinfonet Software, Inc.
9400 Key West Avenue - Suite 250
Rockville, MD 20850
Web: www.jinfonet.com
Phone: 301 838-5560
Fax: 240 465-0355
E-mail: info@jinfonet.com

One main advantage of JReport is the ability to share resources/objects among reports. Once a parameter, formula, or summary has been created, it's stored in a catalog and can be used in any report. Since certain elements within a report, such as address fields, queries, etc., are often repeated in reports, this feature saves developers duplication of effort while enhancing uniformity of design.

JReport supports conditional formatting, by which formulas can be used to calculate summaries, control layout, condition records, etc. For example, to set a field's font properties or control page breaks dynamically, a user simply creates a formula and sets the associated property. Formulas can be executed at the record, group, or section level. JReport provides more than 160 built-in functions as well as an API to write custom Java formulas.

JReport users can set runtime parameters to dynamically control query values and report layout. In addition, grouping and sorting criteria can be selected at runtime. JReport can export data to a number of different

Invoice

Introducing

# SOAPscope 3.0 Debug Test Tune

## 4 Ways to Know Your Web Services

Whether you are learning how a Web service works, or troubleshooting a tough problem, you need the help of a "smart" tool. SOAPscope lets you dig deeper, faster.

**1. Try It** Solve problems by testing your Web service with different inputs without writing any code.

**2. See It** View WSDL and SOAP to understand what's happening. Capture from any toolkit, and see just the right detail for the task at hand.

**3. Diff It** Compare a problem message or WSDL with a similar, working one.

**4. Check It** When the problem's not obvious, rigorous interactive analysis finds inconsistencies, errors, and interoperability problems.

## Look What's New in 3.0

- Integrates with Microsoft® Visual Studio® .NET and Eclipse
- Graph Message Statistics
- Interactive Message Analysis
- Interoperability Testing System
- SSL Support
- HTTP Authentication Support
- HTTP Compression Support
- Support for multi-byte encoding
- Best usability of any Web Services tool

> *"SOAPScope 3.0 is easily the most addictive piece of software I've encountered since Halo. When does the multi-player version come out?"*
>
> **Don Box**
> XML Messaging Architect,
> Microsoft and co-inventor
> of SOAP

The most comprehensive Web services diagnostic system available, and still **only $99!**

**Try It** online at XMETHODS.net or download a trial at Mindreef.com

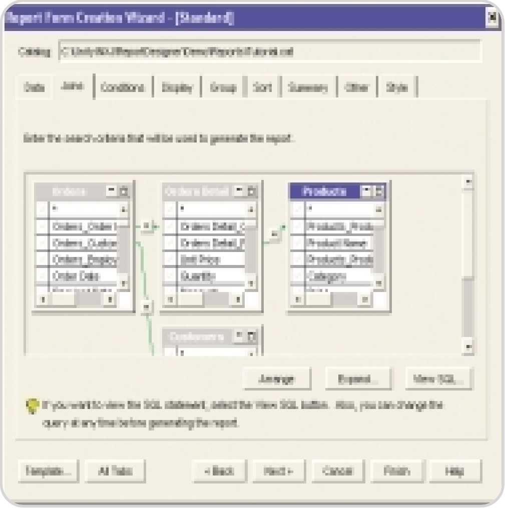
"The Web Services Diagnostic Experts" Mindreef formats, including HTML/DHTML, e-mail, PDF, Excel, Fax, RTF, and XML. Users can preview reports in these various formats while they are designing the reports.

## High Performance

Unify needed its reporting component to be flexible yet powerful enough to support sophisticated enterprise reporting demands. At the heart of JReport is a high-performance engine that can handle any workload demand and easily scales from small deployments to large multi-CPU environments. In addition, JReport supports remote administration, enabling server configuration and management from any Web browser. It also offers a clustered server option that provides high availability, load balancing, and failover protection to ensure uninterrupted access to critical business information.

JReport has several features that optimize performance by making the most efficient use of system resources. JReport provides the ability to fine-tune data buffers usage during report generation and to determine the use of temporary files. JReport is also able to cache connections, allowing incoming reports to utilize existing connections. This helps to streamline the process of executing reports, allowing the system to run more efficiently. To optimize performance in a multiple CPU environment, JReport provides “report bursting” that divides a large report into concurrently running smaller pieces of reports that are later automatically merged into a final report.

Once a report is run, that report and all its resources can be cached for future use. This greatly enhances performance in situations where the same reports need to be generated at regular intervals. Reports can also be archived with their results for future reference.



C:\Unify\NXJ\ReportDesigner\Demo\Reports\Tutorial.cat

Enter the search criteria that will be used to generate the report.

If you want to view the SQL statement, select the View SQL button. Also, you can change the query at any time before generating the report.

## Enterprise Deployment

Deploying reports to an enterprise server is fast and intuitive through JReport’s Deployment Wizard. Users can choose a catalog, select any number of reports from the catalog, select the desired destination within the server directory, and view a list of the connection properties for that catalog. With just a few mouse clicks, the reports and their resources are deployed for production.

Enterprise-wide deployment is also enabled through JReport’s support for any data source, including user-defined data sources. In addition, JReport can automatically generate reports based on the data structure inherent in hierarchical data sources like XML and EJBs.

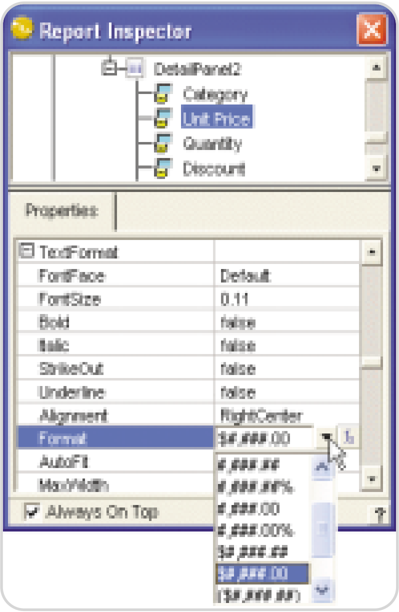
The JReport Enterprise Server API provides an easy-to-use interface for resource management, task scheduling, version control, caching, connection pooling, and security. JReport organizes report resources, including JReport catalogs, report templates, and report results into a logical hierarchy that can be accessed using a Web browser. Users don’t need to know the details of where a report catalog, template, or result is physically stored in order to access it. Security can be controlled at the record, row, or page level to authorize end-user access to personalized information, enabling a single, large report to serve any number of individuals across an enterprise.

With JReport’s National Language Support, the same report can be viewed in different languages depending on the end user’s locale. JReport incorporates a Java mechanism called “resource bundle,” which separates language-dependent strings into isolated files to simplify the translation process.

## Seamless Integration

Unify required a flexible, open report deployment and delivery system that was easily integrated – and branded – so that it could be embedded seamlessly into the NXJ application platform. Once again, JReport delivered. Its 100% J2EE architecture does not require its own server environment and can be run as Java servlets, allowing for quick and seamless integration into any enterprise environment.

JReport is easily implemented in a typical three-tier J2EE environment, and utilizes native Java connectivity to access data sources and the adapters needed to retrieve data on demand. While JReport can be implemented in a number of configurations within the J2EE environment, our needs required integration as an EJB. In this embedded model, Unify NXJ calls J2EE components provided by JReport, and JReport calls J2EE components provided by NXJ in order to customize specific functions.



| | |
|---|---|
| TextFormat | |
| FontFace | Default |
| FontSize | 0.11 |
| Bold | false |
| Italic | false |
| StrikeOut | false |
| Underline | false |
| Alignment | RightCenter |
| Format | $#,###.00 |
| AutoFit | |
| MaxWidth | |

#,###.##
#,###.##%
#,###.00
#,###.00%
$#,###.##
$#,###.00
($#,###.##)

JReport allows users to design their own JavaServer Pages (JSPs) to customize the look and feel of their reporting system interface. JReport integrates with our existing system and application-level security, so users can access business information with a single sign-on.

JReport can also notify a designated user of report completion or report failure by e-mail. The user receives an e-mail message containing a URL link to the report. In addition, reports can be sent in the body of an e-mail or as an attachment.

## Summary

JReport is a powerful, open, easy-to integrate, and easy-to-use reporting solution. With an intuitive and feature-rich design interface, JReport makes designing customized reports quick and easy. Its servlet architecture provides high performance, security, scalability, and seamless integration into any J2EE environment.

Training requirements were minimal, though we did find documentation to be somewhat complicated. For NXJ customers, we simplified documentation, helping to establish JReport as a more seamless out-of-the-box reporting solution.

CUSTOMER SUCCESS BULLETIN

# Quest Software's J2EE Health Check Proves Diagnostic Prowess at Blue Agave

It's pretty hard to deny the real value of a diagnostic solution when it saves you two man-days the very first time it's used. But that's precisely what happened when Blue Agave Software took Quest Software's J2EE Health Check.

Blue Agave develops real-time business activity monitoring software for the consumer goods and consumer electronics manufacturing industries. Privately held, Blue Agave has had success already with its "Active Performance Management" solution, working with manufacturers to improve their operating performance with leading retailers. Blue Agave's solution leverages a company's ERP and other back-end systems along with retailer-provided sales and inventory data to improve response to unexpected business problems like inventory shortages and excess, late orders and supply/demand mismatches.

> *"Quest PerformaSure showed us the root cause of a bottleneck in just 30 minutes… the same one took our engineer two days to track down manually."*
>
> **Mark Wick**
> Senior VP Engineering
> *Blue Agave Software*

## Real Results:

- Improved performance of one transaction 800 percent
- Faster problem diagnosis from system-wide to code-level analysis provided by integration of Quest JProbe

## The Pain of an Unscalable Web-Based Application

After the beta version of Blue Agave's solution was installed in a customer environment, the Blue Agave team started noticing performance issues in high-volume usage scenarios. Although experienced with WebLogic and J2EE, Blue Agave's developers couldn't predict how the application would perform under heavy real-world use. The application was slowing down in ways they couldn't explain.

Mark Wick, Blue Agave's senior VP of engineering, summed up the problem this way: "We could not live with this level of uncertainty as we ramped up our business. We have strict service level agreements with our large clients, and if we can't reliably manage the performance of our bread-and-butter application, we're dead."

## "Do We Really Need Another Tool?" – Enter the J2EE Health Check

Blue Agave doesn't have a big IT budget. Wick's philosophy is to run IT lean and mean, no shelfware. The team already had some tools, but after a week spent poring through log files and metrics dumps, they were no closer to isolating the root cause of their performance bottlenecks.

Taking a recommendation from a Quest Software sales rep, Blue Agave decided to try Quest's new J2EE Health Check program, which uses Quest PerformaSure™ to find, diagnose and resolve complex J2EE application issues.

A Quest system consultant began the Health Check by installing PerformaSure remotely on a spare Blue Agave computer. The consultant showed Wick and his team how easy it is to capture data from the application running under load. The Blue Agave team was stunned when PerformaSure isolated the root cause of a bottleneck in only 30 minutes. Wick remarked that an engineer had recently diagnosed this particular bottleneck after two whole days of sleuthing.

## Quest PerformaSure Improves Transaction Performance

Wick and his team were impressed with the holistic view of their application provided by PerformaSure. For the first time, they could see how a single transaction request was stressing different aspects of the system. Using PerformaSure and Quest Software's JProbe® together, Wick's team tuned the code to improve performance of one transaction by 800 percent, from 16 seconds down to two seconds.

Wick realized that he really needed Quest PerformaSure to manage the performance of Blue Agave's critical solution and chose Quest PerformaSure over other solutions. He's now also using PerformaSure earlier in development to tune performance before the application hits live production.

## Platform and Environment:

- One WebLogic 8.1 instance
- 2 CPU Xeon server (2 GB RAM) running Windows 2000
- App has approx 55 different types of EJBs, heavy use of JMS and memory
- Quest PerformaSure 2.3

Visit us at: www.quest.com/jdj_advertorial and learn more about ensuring better performing J2EE applications with Quest Software's JProbe and PerformaSure. Quest Software, a leading provider of application, database and Windows management solutions, provides Application Confidence to 18,000 customers worldwide. Quest's products help customers develop, deploy and maintain enterprise applications without expensive downtime or business interruption. Headquartered in Irvine, Calif., Quest Software can be found around the globe and at www.quest.com

PAID SUBMISSION BY QUEST SOFTWARE, INC.

# User Experience **Matters**

By Christophe Coenraets

**Christophe Coenraets** worked at Powersoft, which then became part of Sybase, from 1994 to 2000. He started working with Java in 1996 and became the technical evangelist for the company's Java and Internet application division. Christophe then joined Macromedia as the technical evangelist for JRun, the company's J2EE application server. In this position, Christophe started working on rich Internet applications and on ways of integrating Flash front ends with J2EE back ends. Christophe is currently the senior evangelist for Macromedia's new developer-centric rich Internet applications initiative.

ccoenraets@macromedia.com

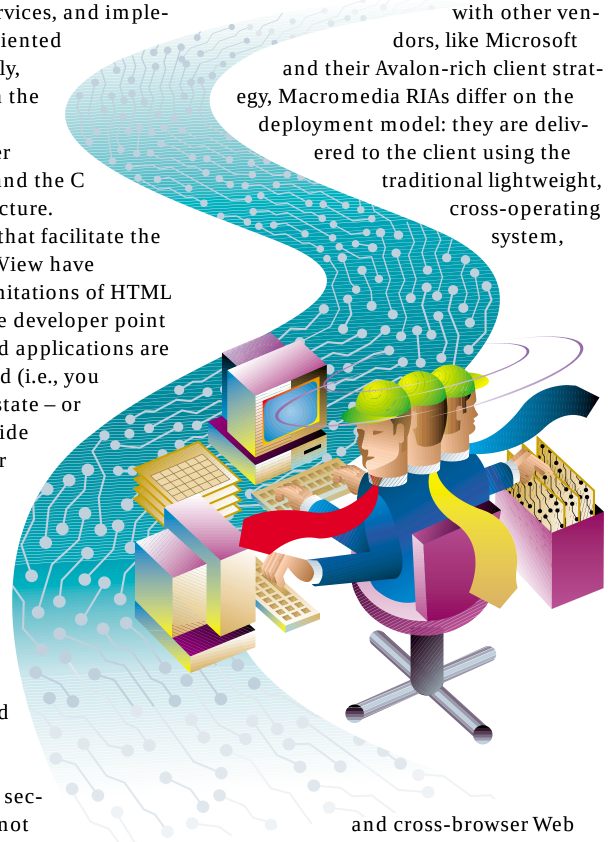
Much has been said about the limitations of HTML, coupled with the HTTP request/response model, for delivering the user interface of Web applications. These limitations revolve around the page-centric and stateless nature of the Web, HTML's limited number of user interface components and metaphors, and the absence of smart client-side data manipulation.

While it's a well-known problem, not much has been done about it. For the past few years the focus of many IT organizations has been on the back end: consolidating business and data-access logic, exposing business processes as Web services, and implementing a service oriented architecture. Similarly, many innovations in the Java APIs have been focused on the server side, around the M and the C in your MVC architecture.

Yes, frameworks that facilitate the development of the View have emerged. But the limitations of HTML are twofold: from the developer point of view, HTML-based applications are cumbersome to build (i.e., you maintain the client state – or even handle client-side events – at the server side). From the end-user point of view, HTML-based applications are cumbersome to use. Current frameworks do a great job of solving the first problem, but the end result is still an HTML-based user interface. Hence the second problem is still not solved.

This also comes at a time when people in charge of the business side of applications realize that these applications don't always deliver on the promised ROI: e-business applications' completion rates are still low, and inside the firewall internal applications' utilization rates could use a boost as well.

In general, there is an increasing sense that the back end is currently in fairly good shape, and that we are at a stage where the client side needs to catch up to allow the advances made at the server side to fully deliver on their promise.



Macromedia has been working on rich Internet applications for a long time, and while sharing a common vision for a desktop application–like user experience with other vendors, like Microsoft and their Avalon-rich client strategy, Macromedia RIAs differ on the deployment model: they are delivered to the client using the traditional lightweight, cross-operating system, and cross-browser Web deployment model. These RIAs run on the ubiquitous Flash Player platform that enables this universal deployment.

In addition to this ubiquitous virtual machine and a lightweight deployment model, developers also need a programming model that supports established methodologies and design patterns and that integrates with existing IDEs and version control systems. To address this requirement, Macromedia recently released Flex, a platform that enables enterprise developers to build rich Internet applications using a familiar programming model. The Flex programming model builds on the strengths of two popular development paradigms: markup languages and traditional object-oriented programming languages. Use the Flex XML markup language (MXML), much like HTML, to declaratively lay out the user interface of your application. MXML includes a much richer set of tags than HTML and allows you to create your own components. The first time the MXML application is requested, it's compiled into Flash bytecode much like a JSP is compiled into a servlet. The major difference is that the generated bytecode (the View) is executed at the client side by Flash Player, providing the users with a much more engaging experience than traditional HTML-based applications.

Building rich Internet applications is often a liberating experience: the limitations of HTML disappear, and it's hard not to get excited by the new possibilities and opportunities. You can finally build real applications for the client tier. These applications are stateful, and are neither cluttered with page refreshes nor limited to a handful of user interface controls. They can expose rich user interface metaphors such as drag and drop, support smart client-side data manipulation, and even access a local data storage area to work in an occasionally connected environment. But most important, they integrate naturally with your existing middle tier business logic and frameworks.

OPEN
24 HRS
What//: are your Web applications saying to your customers?

# Talk of Two Communities

## The JCP and Jini

Onno Kluyt

This month I have a new format for my *JDJ* column. Due to the recent organizational changes, a second effort now reports to me: the Jini technology. And so, starting with this issue, you'll be able to catch up on the efforts of two communities: the JCP and the Jini community at large.

### A Handful of Proposed Final Drafts

Several JSRs have reached the Proposed Final Draft stage, which is the last step in a JSR's life before attempting to become final through the Final Approval Ballot. A JSR's Proposed Final Draft will be, as the name suggests, a draft spec that the Spec Lead expects to be very close to the final document. It provides developers with a chance to catch and query any remaining ambiguities. JSR 200 (Network Transfer Format for Java Archives) posted a second Proposed Final Draft. This API, in the javax.pack name space, defines a dense archival format that delivers substantial savings compared to JAR files. The SAMS Messaging JSR led by Nokia (JSR 212) is also close to finalizing. The JSR defines an API for servers providing SMS (Short Message Service) and MMS (Multimedia Messaging Service) functionality. This API will work on top of J2SE and J2EE implementations. JSR 133, led by Sun, revises the Java Memory Model and Thread Specification. The JSR focuses on volatile variables, final variables, immutable objects, class initialization, asynchronous exceptions, and thread interrupts. This JSR is part of the upcoming J2SE platform version 1.5.

> "Proposed Final Draft...provides developers with a chance to catch and query any remaining ambiguities"

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

*onno@jcp.org*

### A New JSR for Device Management

There is one new JSR that I'd like to draw your attention to this month – Siemens has proposed JSR 246, Device Management API. This optional package for CLDC- and CDC-based platforms allows the developer to manage a device through the device's natively implemented device management protocols; this management can be performed remotely. The JSR will focus on several existing protocols such as SyncML/OMA DM and the WAP OTAP protocol and aims to integrate with management frameworks such as OSGi's MEG specification. The Spec Lead hopes to complete the JSR in about 12 months.

### Jini Technology, An Introduction

The Jini technology is an architecture for things that connect to a network. Those things can be devices (card readers, printers) or software application components. It takes a strong look at Deutsch's network fallacies (e.g., "the network is secure," "there is one administrator," "the network is reliable," etc.), makes the developer think about them, and helps you create services and clients that participate successfully in distributed, federated systems. You can read more about the Jini technology, and its use and adoption at http://Jini.org.

### Jini Technology @ the JavaOne 2004 Conference

The Jini technology and the community are on display and in action several times during the week in San Francisco. First, on Tuesday evening the Jini community is one of the hosts of the "Java Communities in Action" event in the Argent Hotel. There are four technical sessions and three Birds of a Feather sessions. On Tuesday, Brian Murphy of Sun will be giving a practical guide to the Jini network technology version 2 (TS-1075). On Thursday, Leon Chism and Steven Hoffman of Orbitz will be presenting on this technology for use in a service oriented architecture as a low-cost alternative to the Enterprise JavaBeans technology (TS-2614). On Monday afternoon Ronald Simmons (Invesco) will be talking about Wall Street's use of grid computing and the Jini technology for large-scale deployment of computational resources (TS-2387). In the fourth technical session on Wednesday, Kevin MacDonald and Larry Mitchell (Sun) will discuss the use of the Jini architecture in combination with RFID and Auto-ID technology (TS-1313). There are two Jini-related BOFs scheduled for Monday evening: Brian Murphy and Keith Thompson will host a general conversation on the latest release of the Jini technology (BOF-1076), followed by Peter Jones' and Nigel Daley's (Sun) Q&A session on Jini Extensible Remote Invocation and Java Remote Invocation Method (BOF-1134). Then on Tuesday evening Cameron Roe (PsiNaptic) and Vladimir Rasin (Ford) will be discussing the integration of in-vehicle components (BOF-3807).

That's it for this month. I'm very interested in your feedback. Please e-mail me with your comments, questions, and suggestions.

NEW – JReport 7
See us at JavaOne
Booth 1624

# Yeehaw?

## Some reporting solutions are pretending to be something they're not.

When you look under the fancy packaging, there's only one Java reporting solution with years of proven, real-world experience and thousands of successful deployments worldwide – JReport.

JReport is a comprehensive, 100% J2EE reporting solution that is intuitive in nature and easy to integrate, eliminating the need for costly professional services and training. From departmental applications to enterprise-wide deployments, JReport scales to meet your reporting needs, ensuring uninterrupted access to critical business information, so you get the answers you need, when you need them.

Whether you need to deploy sophisticated reports via the Web or enable users to create ad hoc reports on demand, JReport delivers on the promise of self-serve reporting. And, with JReport's *one-click analysis*TM, any user can slice-and-dice, pivot, and drill-down on a report, from any Web browser.

Visit **www.jinfonet.com/yeehaw** or call **301-838-5560** today, and discover why, when it comes to Java reporting, JReport is the real McCoy.

JINFONET
SOFTWARE

# Letters to the Editor

## A Generic Concept

*[“Service-Oriented Architecture” by Ted Farrell, Vol. 9, issue 4]*

This article clearly explains the need for a service abstraction layer for building robust applications in an SOA-enabled environment. As Ted Farrell points out, SOA is a generic concept that goes well beyond Web services and extends to include EJBs, other XML-based communications, JMS, database access, JavaBean objects, and so on. Essentially, the best practice for accessing the service boils down to defining a unified service interface that is instantiated and executed by the client in the same fashion regardless of the underlying source of the business service.

*—Arun Candadai*

## AOP Involves a Leap of Understanding

*[“Investing in ‘Professional Open Source’” – An interview with David Skok, Vol. 9, issue 4]*

AOP is definitely a revolution along the same lines that OOP was a revolution. Like OOP, it will only simplify development for expert programmers who learn it well, not for the less expert audience. Like OOP, poorly written AOP will be even more difficult and error-prone than poorly written J2EE.

Even today, I know many people who misunderstand OOP and, heck, OOP has been mainstream for 10 years. AOP involves a similar leap of understanding and it will take years for the majority of programmers to accept it. Even then, like OOP, most programmers will not dedicate themselves to become experts in it and, thus, only gain limited benefits.

It is highly doubtful that AOP will do anything to help that 60%, ever.

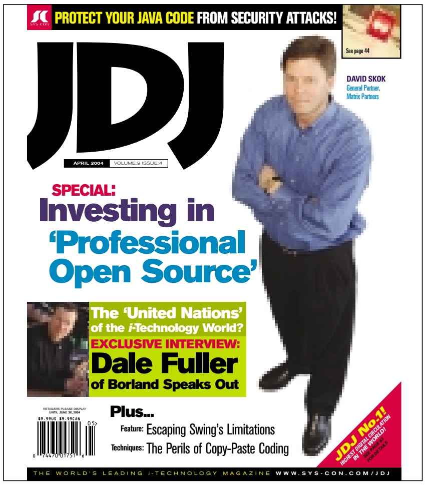
*—Daniel Howard*

## Will This Become a Minefield?

*[“The Commercialization of Open Source” by Kirk Pepperdine, Vol. 9, issue 4]*

Two-tier licensing sets one clear difficulty – how does the community deal with selling freely donated code? I suspect that this may be a minefield in the future. Another more general issue is that some form of funding may be necessary to sustain projects in maintenance mode – not many people will do maintenance programming for nothing. You can already see this on Source Forge, which has a fair number of stalled projects. Perhaps this is a Darwinian trait but I do think it will become a feature of many open source projects where sexy new features will be added but annoying bugs and poor documentation will not be fixed. There is also the frequently raised argument that if everything is open source, who pays the developers who write open source – what do they do as their day job?

*—Ken*


PROTECT YOUR JAVA CODE FROM SECURITY ATTACKS!
See page 44
JDJ
APRIL 2004
VOLUME:9 ISSUE:4
DAVID SKOK
General Partner, Matrix Partners
SPECIAL:
Investing in ‘Professional Open Source’
The ‘United Nations’ of the i-Technology World?
EXCLUSIVE INTERVIEW:
Dale Fuller of Borland Speaks Out
RETAILERS PLEASE DISPLAY UNTIL JUNE 30, 2004
$9.99US $9.99CAN
Plus...
Feature: Escaping Swing’s Limitations
Techniques: The Perils of Copy-Paste Coding
JDJ No.1! HIGHEST DIGITAL CIRCULATION IN THE WORLD! SEE PAGE 63 FOR DETAILS
THE WORLD’S LEADING i-TECHNOLOGY MAGAZINE WWW.SYS-CON.COM/JDJ

Open source has always felt like an extension of grad school for students/programmers who can’t seem to break away. Some day the open source programmers/hobbyists are going to get a life and wonder why they’ve been programming for free in their spare time after a full day’s work. An infusion of cash can only feed the development and make it grow. The open source model is an anomaly in the business world and the better projects will inevitably become commercialized software. Paying the programmer isn’t a sin; neither is buying commercial software. Programmers have to eat too.

*—JP*

## Ensure Portability Through Standards Compliance

*[“Scalability of J2EE Applications” by Stefan Piesche, Vol. 9, issue 4]*

I recently investigated several open source and commercial caching products and hit somewhat of a brick wall with the J2EE and EJB specs. I needed a cache to serve read-mostly data in a clustered EJB application. The application runs on several app servers, so portability was a key requirement.

In my experience, the key element to ensuring portability is standards compliance. The J2EE and EJB specs forbid EJBs from using certain APIs, namely those that create threads, block on socket operations, etc. They go even further by explicitly naming forbidden APIs, such as JMS’s setMessageListener(). However, pretty much all the caching products I investigated were using these APIs.

In one example, I queried the vendor (who shall remain nameless!) about the use of setMessageListener() in their product. They replied by stating that (1) it works and (2) it was fine to use those APIs in classes that the EJB *uses*, just not in the *actual* EJB bean class.

My feeling on (1) was that if it contravenes the spec, then working on AppServer version *X* doesn’t guarantee it will work on version *X*+1. It also doesn’t mean it will work on any other app server. On (2) I don’t see that it makes any difference whether the call is in class A or in class B, which gets called from A (in the same thread)?!… The vendor in question had only recently started marketing their product as a J2EE cache (previously it had been best known as a servlet cache). I don’t know if those restrictions apply to the Web tier.

In the end, due to portability concerns, we decided to roll our own. Our invalidation strategy was simple and was achievable using JMS and MDBs (no spec contravention required).

*—John Segrave*

## ULC vs Droplets

*[“UltraLightClient by Canoo Engineering AG” reviewed by Peter Leitner, Vol. 9, issue 4]*

UltraLightClient reminded me of Droplets, which were reviewed in the February issue (Vol. 9, issue 2). So I tried to figure out what the differences are. Here’s the result: (1) ULC integrates into J2EE servers while Droplets need their own application server; ( 2) ULC uses Swing on the client while Droplets uses AWT; (3) ULC’s server-side API is Swing-like while the Droplet widgets come with their own API; (4) the ULC client is pure Java while the Droplet client has some native code; and (5) ULC uses J2EE communication infrastructure (HTTPS or RMI and IIOP) while Droplets use tunneling.

*—Richard Ballestero*

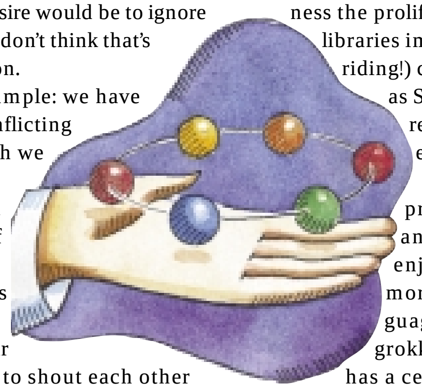
SOAPtest®
Jtest®
.TEST®

–continued from page 6

## State of the Union

Of course, that "cross-platform" means "…as long as you're running Windows," and for those of you who are thinking of Mono right now, I'd like to point out that I had to massage Mono to get it to install on any OS besides…Windows.

Remember, though, that our first example showed us a core group of users and developers for whom Java's cross-platform quality is irrelevant for all intents and purposes. My personal desire would be to ignore them as outliers, but I don't think that's really a valid conclusion.



In yet another example: we have a whole gamut of conflicting mechanisms by which we can persist data, process data, present data, or even think of data. Each of these different mechanisms have their proponents, all waving their flags like mad, trying to shout each other down.

They represent a tyranny of choice and, in the end, the array of options actually makes doing something in Java harder rather than easier. What's more, this problem shows no signs of going away.

In (thankfully) my last example: we have the JVM itself, slowly morphing and growing into a behemoth all its own. Far from UCSD's concept of p-Code, the VM now far supersedes anything we're likely to deploy on it (unless we're BEA, IBM, or Sun), and even there it's not sufficient: witness the proliferation of third-party libraries implementing (and overriding!) core mechanisms, such as SWT and Thinlets, in a reference to my very first example.

As an administrator, programmer, architect, and user, I like Java. I enjoy working with Java more than I have any language since I first really grokked C, honestly. Java has a certain simple clarity that allows and encourages me to focus on the problems at hand, and a lot of that can be credited directly to Sun's stewardship. I can't say I think everything Sun's done has been "right" or "good for Java," necessarily, but I think Sun's done a bang-up job, all things considered.

Sun doesn't exist in a vacuum, however. There are a lot of forces at play, some of which I've tried to enumerate, and I wonder now if Java hasn't started to go past the point of maturation to the point where it'll be spoken of the same way COBOL is today: in use in the sweatshops, still carrying its load, but no longer glamorous or entirely desirable, no longer really growing and improving publicly and measurably. Sure, improvements are made, even important ones – but Java's only keeping the mind share it has, not really garnering new converts in large quantities.

My problem is that I don't know what will replace it. There are lots of technologies out there, but right now they seem to be all glitz, all flash-in-the-pan technologies that impress you but don't make you want a second date.

What do you think?

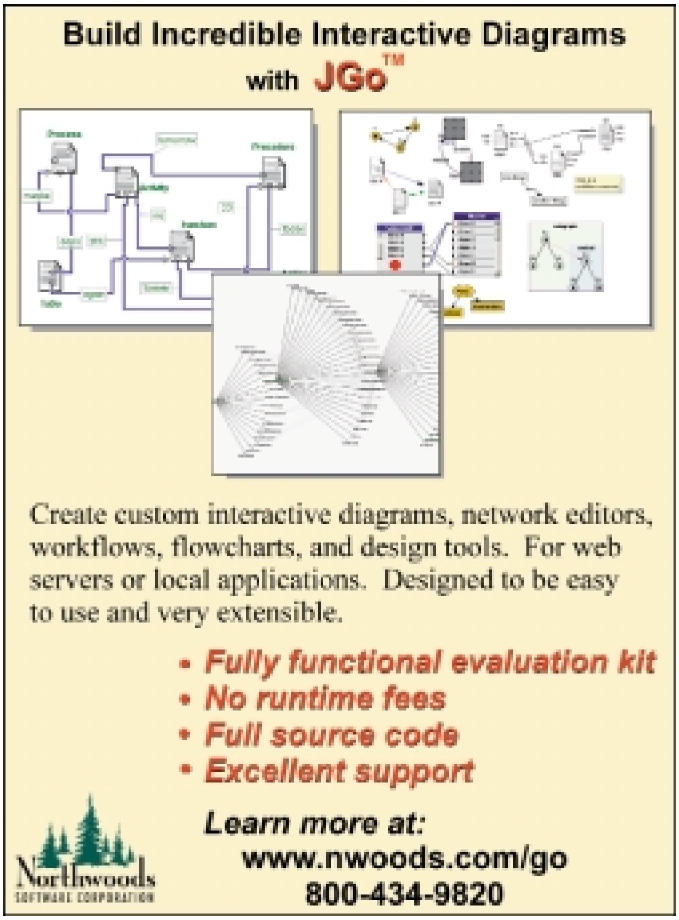
Think long term, and enjoy!

# JAVA DEVELOPER'S JOURNAL Advertiser Index

| Advertiser | URL | Page |
|---|---|---|
| Agitar | www.agitar.com | 39 |
| Altova | www.altova.com | 13 |
| AMD | www.developer.amd.com | 51 |
| Borland | www.go.borland.com/j6 | 7 |
| Business Objects | www.businessobjects.com/v10/047 | 29 |
| Canoo Engineering AG | www.canoo.com/ulc | 35 |
| ClearNova | www.clearnova.com/thinkcap | 37 |
| Compuware | www.compuware.com | 45 |
| DataDirect | www.datadirect.com/jdj | 4 |
| Enerjy | www.enerjy.com | 21 |
| ESRI | www.esri.com/develop | 27 |
| Fiorano | www.fiorano.com/downloads | 53 |
| Google | www.google.com/cacm | 50 |
| GreenPoint | www.webcharts3d.com/demo | 55 |
| H&W Computer Systems | www.hwcs.com/jdj01.asp | 65 |
| Identify Software | www.identify.com | 47 |
| ILOG, Inc. | www.jviews-into-kit.ilog.com | 9 |
| InetSoft | www.inetsoft.com/jdj | 49 |
| InferData | www.inferdata.com | 41 |
| InterSystems | www.intersystems.com/match1 | 19 |
| Jinfonet | www.jinfonet.com | 67 |
| Librados | www.librados.com | 57 |
| Mindreef | www.mindreef.com | 61 |
| Northwoods Software Corporation | www.nwoods.com/go | 71 |
| Oak Grove Systems | www.oakgrovesystems.com | 59 |
| Oracle | www.oracle.com/platform | 2 |
| Parasoft | www.parasoft.com/achievequality | 69 |
| Parasoft | www.parasoft.com/jtest | 15 |
| Quest Software | www.quest.com/jdj | 76 |
| Quest Software | www.quest.com/jdj_advertorial | 63 |
| Rascal Software | www.rascalsoftware.com/jdj | 33 |
| Reporting Engines | www.reportingengines.com | 31 |
| SAP | www.sdn.sap.com | 25 |
| Scientific Toolworks, Inc. | www.scitools.com | 73 |
| Sleepycat Software | www.sleepycat.com/bdbje | 43 |
| SoftwareFX | www.chartfx.com | 75 |
| Spirent Communications | www.spirentcom.com | 70 |
| Tangosol | www.tangosol.com | 17 |
| Visual Paradigm | www.visual-paradigm.com | 11 |
| Web Services Edge East | www.sys-con.com/edge | 71 |

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

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



# Industry News

## InetSoft Releases Style Report 6.0 and Style Report Analytic Edition

(*Piscataway, NJ*) – InetSoft Technology has announced the general availability of Style Report 6.0. A new member of the Style Report family, Style Report/Analytic Edition is introduced with a host of ad-hoc reporting and analytic functionalities.

Style Report/Analytic Edition is built on top of the Enterprise Edition. The design goal for Style Report/Analytic Edition is to solve the conflict between data analysis and ease-of-use in a zero client Web environment. Style Report/Analytic Edition breaks up analytic needs into small, easy-to-learn, and easy-to-use components.

www.inetsoft.com/inetsoft/index.html

## GreenPoint Releases WebCharts3D v5.0

(*New York, NY*) – GreenPoint, Inc., a developer of the XML-enabled interactive Web visualization solutions, has released WebCharts3D version 5.0. New features include a chart designer with integrated image server, DTD and XSD schema production, an API for Swing designer components, multiview charts, animated Flash generation, new shading and gradient effects, and more.

WebCharts3D can produce complex interactive charts in a number of different formats (PNG, JPEG, PDF, SVG, SWF, GIF, TIFF, WBMP) and deliver them to browsers and mobile devices.

A free 30-day fully functional evaluation copy of WebCharts3D v5.0 is available at www.gpoint.com/demo.

## Sabre Airline Solutions and Sun Take Airlines to New Heights

(*Santa Clara, CA/Southlake, TX*) – Sun Microsystems, Inc., and Sabre Airline Solutions have teamed to design and implement a next-generation architecture for Sabre Airline Solution's application service provider (ASP) portal offering, the Sabre eMergo Web-enabled and dedicated network solutions. Based on Java Enterprise System components and Sun Fire servers, the Sabre eMergo offering has reduced IT costs by as much as 50 percent for participating airline carriers worldwide.

www.sabre-holdings.com

## Versant Joins Telecom Leaders in OSS Through Java Initiative

(*Fremont, CA*) – Versant Corporation, a provider of data management and integration for the real-time enterprise, has joined the OSS through Java (OSS/J) Initiative. OSS/J provides Java, XML, and Web services application program interfaces (APIs) and Design Guidelines in the OSS (Operation Support System) area, which provides key systems for telecom carriers to operate and monitor their services. As a member of OSS/J, Versant joins a distinguished group of telecommunications industry leaders who combine resources in a collaborative effort to facilitate integration and fast deployment of OSS components, and speed the development of innovative telecommunications solutions.

www.versant.com

## ICEsoft Announces Java Browser Support for ADA/Section 508

(*Calgary, Alberta*) – ICEsoft Technologies Inc., a provider of Java browser and Java Web access software, has announced that the new release of ICEbrowser is compliant with Section 508 of the Federal Rehabilitation Act.

ICEbrowser 6.0 allows impaired users to make full use of ICEsoft Java client technology. ICEbrowser-rendered applications configured to support accessibility in the rendering panel can now be navigated with keyboard commands using commercial text reader software that can also verbalize content to impaired users. ICEbrowser's Swing-based HTML rendering component supports the Accessible Hypertext interface, which is part of the Java Accessibility package. Through this API, applications have a standard mechanism for accessing the content of the rendered page through attributes and spatial location.

www.icesoft.com

## Nokia Shows Multiplayer Java Games with Key Community Features at E3

(*Los Angeles*) – Nokia has premiered the first multiplayer Java games based on its SNAP Mobile solution at this year's Electronic Entertainment Expo (E3). Developed together with Sega Mobile, the SNAP Mobile demonstration features multiplayer gaming for Java games, in addition to key community features such as friends lists, presence, and instant messaging. The Sega Mobile game demos are the first example of how SNAP Mobile brings the technology utilized in the N-Gage Arena gaming community to mass market Java terminals.

www.nokia.com

## Quest's New Release of JProbe Profiler Expedites Diagnosis of J2EE Performance Bottlenecks

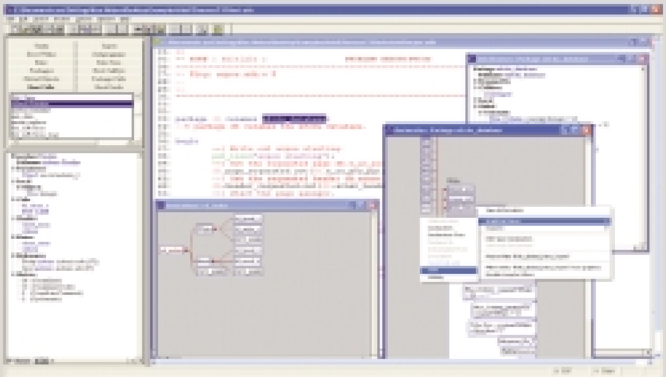
(*Irvine, CA*) – Quest Software, Inc., a provider of application, database, and Windows management solutions, has released a freeware edition of its Java development solution, Quest JProbe Profiler. A Java profiling tool, JProbe Profiler helps developers diagnose performance bottlenecks in Java code. Offered free of charge to the Java development community, the freeware edition of JProbe Profiler makes professional-quality Java development easier and more accessible to Java developers.

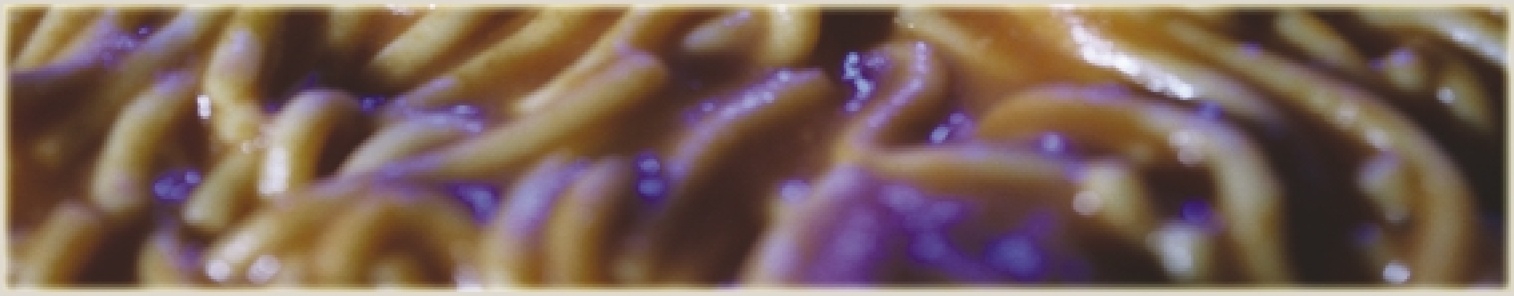
www.quest.com

## Xythos Chooses Dralasoft Workflow for Combined Document/Workflow Management Solution

(*Westminster, CO*) – Dralasoft, Inc., a provider of Java technology for Business Process Management (BPM), has announced that Xythos Software, a developer of simple document and file management software, has chosen to partner with Dralasoft to provide integrated workflow/document management solutions for enterprise use. The solutions will provide a new way for organizations to access, distribute, route, process, and review documents and other file-based information according to the user's preset rules and standards.

www.dralasoft.com





# What's New Is **Old Again**

by Margaret Leber

*I do strongly think that people, when they start throwing computers at something, they think that it's a whole new ballgame, so why should they study the past. I think that is a terrible mistake.*
—Donald Knuth

*Those who do not remember the past are condemned to repeat it.*
—Santayana

One thing that's always struck me throughout my career as a professional in computing has been how little regard or study we give to the history of our profession.

I suppose this situation has been engendered to an extent by the rapid growth in the technologies we work with. It hardly seems possible sometimes that lessons learned long ago (when FORTRAN pterodactyls roamed the Earth, or when PL/I T. Rex feasted upon COBOL flesh) could have any relevance in a Web services world or help us deal with adapting object orientation to a playing field where databases are by and large relational.

Of course, being a certified old-school fogie myself, I'm certainly guilty of holding forth on how romantic and primitive things were back in the day ("walking 10 miles to school in the snow, uphill both ways, with a Teletype strapped to my back and we *liked* it!"); that sort of nostalgia doesn't sell well to the current population of the community. That said, I really do believe that we don't spend as much time as other engineering disciplines do on understanding how things used to be done and, more important, why.

Knowing this will help us understand why things are as they are now, and how they got to be that way, and might well shed some light on the way forward.

By way of an illustration, think for a moment about the perennial debate between adherents of C/C++ language technology and advocates of Java (or other virtual machine–based languages). One often heard plaint from the C people is that hand-crafted C code is inherently more performant than anything possible through "an interpreter," and therefore whatever benefits the newer approach might have aren't worth the cost of not being able to wring every last cycle out of the machine, runtime optimization possibilities notwithstanding.

A "new" debate? Not exactly. Consider this reply:

*What about the classical objections to such a tool? There are three: It doesn't let me do what I want. The…code is too big. The…code is too slow.*

*As to function, I believe the objection is no longer valid. All testimony indicates that one can do what he needs to do, but that it takes work to find out how, and one may occasionally need unlovely artifices. As to space, the new…compilers are beginning to be very satisfactory, and this improvement will continue. As to speed, [optimizers] now produce some code that is faster than most programmer's handwritten code…*

Perhaps you can see I've edited the above just a little to avoid tipping my hand. As familiar as it may sound to today's ear, this isn't a recent apologia for Java technology. The author was the venerable Fred Brooks, and he was talking about the debate over using assembler code versus PL/I in the Operating System/360, developed for IBM mainframes, consuming (by his estimate) 5,000 man-years of effort from 1963–1966.

Now that's old school, right? Brooks was writing in *The Mythical Man Month* of lessons learned in a project completed two years before I wrote my first computer program. *MMM* was written in 1974, around the time I entered the industry on a paid basis. This is fully four years before the 1978 copyright date of my copy of Kernighan and Ritchie's *The C Programming Language*, so we can say that the "don't use {higher_level_language_X} because it isn't performant" debate is not only older than Java, it's actually older than C. Originals of both books live in a spot of honor on my bookshelf, next to their 20th anniversary editions, themselves now getting a bit long in the tooth.

That particular debate is now fully thirty-some years old, would you say? Not quite…

Here's another quote from a different source, and I won't monkey with this one at all, since you know something of what I'm up to now:

*The first programming system to operate in the sense of a modern compiler was developed by J. H. Laning and N. Zierler for the Whirlwind computer at the Massachusetts Institute of Technology in the early 1950s…it was, in John Backus' words "an elegant concept elegantly realized"…he also remarked that it was all but ignored because it threatened what he called the "priesthood" of programmers, who took perverse pride in their ability to work in machine code using techniques and tricks few others could fathom, an attitude that would persist well into the era of personal computers.*

*Donald Knuth…saw another reason in the allegation that the Laning and Zierler system was slower than a factor of ten than other coding systems for Whirlwind.…Closing that gap between automatic compilers and hand coding would be necessary to win acceptance for compiler systems and break the priesthood of the programmers.*
—*A History of Modern Computing* by Paul Ceruzzi

Now this is recognizably the same debate, taking place nearly two decades before Brooks' project.

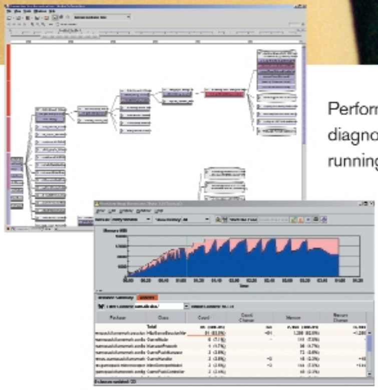
The writing has a consciously historical bent, and it offers some insight into reasons why the newer technology was rejected by the establishment…reasons that perhaps didn't have as much to do with the technology, but more to do with "programming as a human activity" – a delicious turn of phrase we owe to Jerry Weinberg's classic *The Psychology of Computer Programming*, a work of the same era as Brooks' book. More understanding of the "paradigm-shift" pattern at work here (and please pardon my use of the word "paradigm," oft-abused by New Age management theorists of the 1990s) is available in another book from that shelf, Thomas Kuhn's *The Structure of Scientific Revolutions*.

Perhaps if today's developers had a higher awareness of this kind of historical background, we'd spend less time on old debates and more time thinking about newer issues. Goodness knows we have enough of those to deal with.

**Margaret Leber** has been involved in various aspects of computing technology since 1968, and with Java since 1996. She holds SCJP and SCWCD certifications, and is currently studying for the new SCBCD exam.

maggie@voicenet.com

SPEND LESS TIME PROBLEM SOLVING… AND MORE TIME DEVELOPING APPLICATIONS.



PerformaSure – a system-wide performance diagnostic tool for multi-tiered J2EE applications running in test or production environments.

JProbe – a performance tuning toolkit for Java developers.

**Join The Thousands of Companies Improving Java Application Performance with Quest Software.**

Whether it's a memory leak or other performance issues, Quest Software's award-winning Java products — including JProbe® and PerformaSure™ — help you spend less time troubleshooting and more time on the things that matter. Quest's Java tools will identify and diagnose a problem all the way down to the line of code, so you no longer have to waste time pointing fingers or guessing where the problem lies. Maximize your team's productivity with Quest Software by downloading a free eval today from **http://www.quest.com/jdj**.

QUEST SOFTWARE®