SYS-CON

JAVASERVER FACES AND AJAX FOR GOOGLE FANS PAGE 28

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

JDJ

JDJ.SYS-CON.COM VOL.11 ISSUE:4

— SEE PAGES 48-49 AND 60-61 FOR DETAILS —

REAL-WORLD AJAX SEMINAR

April 24, 2006 San Jose, CA

June 5-6, 2006 New York, NY

www.ajaxseminar.com

SOA 10th International WebServices Edge conference+expo

2006 ENTERPRISE> OPENSOURCE CONFERENCE+EXPO

June 5-6, 2006 New York, NY

soaconference.sys-con.com

Debugging JDBC with a Logging Driver

RETAILERS PLEASE DISPLAY UNTIL JUNE 30, 2006

$5.99US $6.99CAN

PLUS...

- Business Intelligence and Reporting with BIRT
- Concurrent Programming and Locking in J2SE 5.0
- All for One and None for All

Java 911

Put out code fires with JProbe®

Tired of constantly fighting fires? Extinguish code issues and prevent future memory flare-ups with Quest Software's JProbe.

Discover the root cause of performance and memory issues in your Java applications. Easily test and fix your applications without any code changes. And even pinpoint performance problems down to the line of code. All with award-winning JProbe.

Stop the fire drills. Gain confidence in your code with JProbe.

Download a trial of JProbe at:
www.quest.com/911

Application Management | Database Management | Infrastructure Management

QUEST SOFTWARE®

Does *i*-Technology Matter?

Jeremy Geelan

When Nicholas Carr posed the question "Does IT Matter?" in his now-famous *Harvard Business Review* essay, he clearly knew that it would provoke discussion. He probably didn't know, on the other hand, that it would eventually cause the world's richest man – whose wealth is derived 100% from IT – to call the essay, during a dinner party at his home, "the dumbest thing I've ever read."

When extended by Nick Carr and published in book form, the essay was subtitled: "Information Technology and the Corrosion of Competitive Advantage." Carr's thesis, simply put, was that since business profits are based on your ability to differentiate yourself, you can only gain an advantage over your competitors by having or doing something that they can't have or do; thus, as IT becomes more and more a commodity, its ability to help your business differentiate itself will decrease.

As Leo Lim has expressed it: "IT is merely a cost of doing business, cut in the same cloth as other innovations that once held a lot of promise like electricity and the railway system. Though no company can ever hope to operate much less compete without the above, its homogeneity and prevalence has somehow blunted its value."

So why would anyone, as I am doing here, go out of their way to ask the same exact question of Internet technologies (i-Technology)? Won't *i*-Technology just follow the same trajectory as its pre-Web predecessor, and end up "not mattering"?

My contention is that it won't.

To back this up, I want to point to the game-changing nature of the Internet as opposed to (merely) the silicon chip. No matter how much Carr might like to suggest that in some way the window of innovation for IT has closed, my counter-argument would be that with the advent of the myriad technologies that the Internet has spawned, the window has blown wide open again. It was blustered ajar by the existence of Web 1.0, and now it's in the process of being blown wide open by the arrival of Web 2.0.

Be in no doubt: i-Technology Does Matter. It matters in ways that IT could only ever dream of. It touches us hourly, daily, weekly. And it touched us in our home lives, at school or in hospitals, as well as in our business lives. In the western industrialized world, at least, i-Technology has altered our entire way of being… and, unlike IT, it is only just getting started.

One tried and true metric of the importance that society attaches to any activity has, throughout the centuries, been vocabulary growth. Just as farming terms multiplied rapidly through the 19th century, the 21st century has so far been characterized by hugely innovative additions to the dictionary, reflecting very real innovations in the real world, including both "blogging" (with its variants like "vlogging" and sub-words like "blogroll" and "blogdropping," and "podcasting" – not to mention the family of words that podcasting too has spawned, such as "podvertising," "nanocasting," "podsafe" and – doubtless soon, alas – "podspamming"). That is before you even begin on acronyms, from the early acronyms like HTML, HTTP, and TCP/IP to more recent ones like DHTML, DOM, and AJAX.

But terminology, even an abundance of it, is what economists would call, at best, a "soft indicator." So next month in this space I will turn from words to numbers and make my argument in terms of solid economic metrics. i-Technology professionals everywhere, stay tuned!

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

jeremy@sys-con.com

ALTOVA® umodel® 2006

Visualize works of software art

Draw on UModel® 2006, and picture better programs based on UML™.

New in Version 2006:

- UML 2.1 support
- C# code generation, reverse engineering, and round tripping
- Sequence diagrams
- Interoperability via XMI 2.1

Altova UModel 2006, the intriguing new entrant in the software design space, is defying convention and earning acclaim as the simple, cost-effective way to draw on UML. Use it to interpret or create software architectures. Decode Java or C# programs into clear, accurate UML2 diagrams, or outline applications and generate code from your plans. With customizable templates, built-in entry helpers, and many other artful features, UModel makes visual software design practical for programmers and project managers. Take the mystery out of UML!

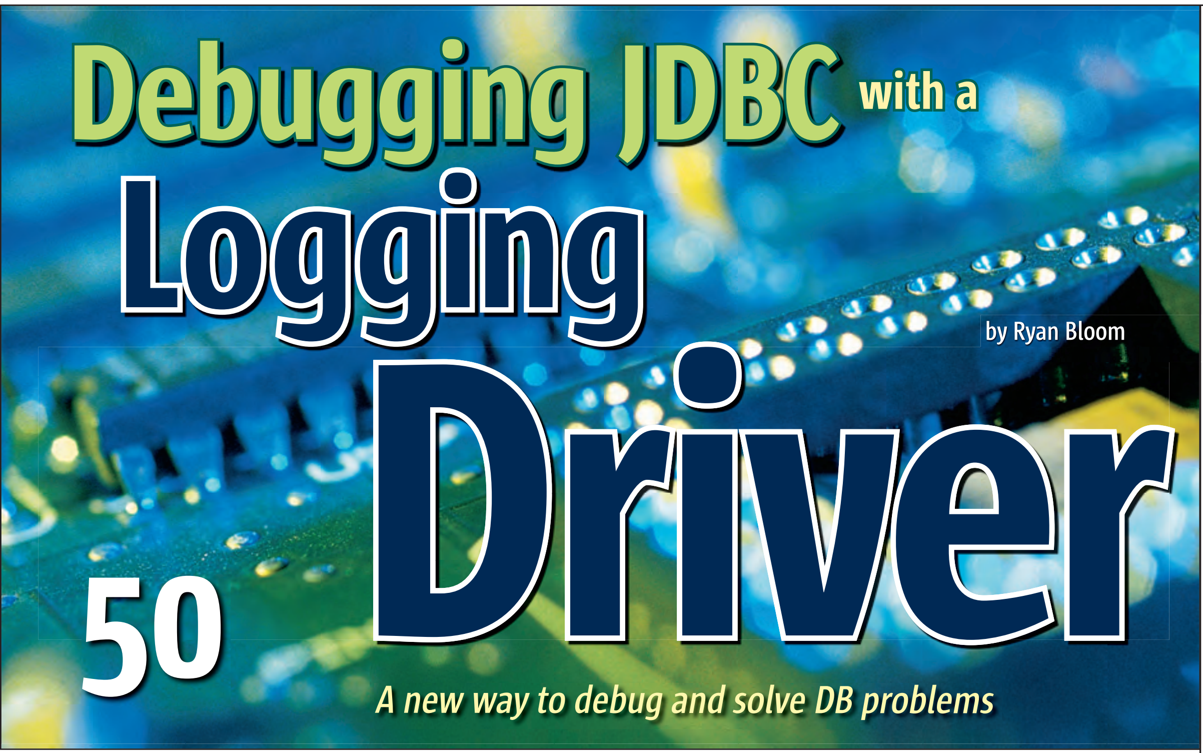
Download UModel® 2006 today: www.altova.com

ALTOVA®

APRIL 2006 VOLUME:11 ISSUE:4

JDJ contents

JDJ Cover Story


Debugging JDBC with a Logging Driver
by Ryan Bloom
50
A new way to debug and solve DB problems

Features

Business Intelligence and Reporting with BIRT
by Chris Beels

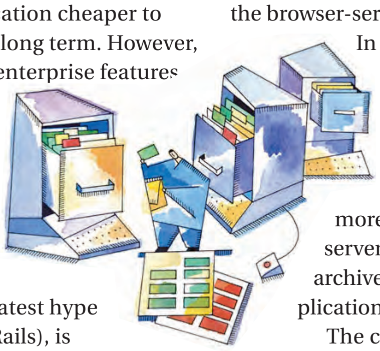
Concurrent Programming and Locking in J2SE 5.0
by Craig Caulfield

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.

Michael Yuan
Enterprise Editor

Future Proof Your Web Application Using Clustered Cache Services

Today's web developers have a lot of choices when it comes to web application platforms. Among them, Java EE has always stood out as a "scalable" solution -- it may not be the easiest platform to prototype a web site, but it protects your software investment over longer terms. For instance, Java EE is a cross platform solution supported by many vendors. That means you can choose from many hardware, software, services vendors to accommodate the current and future needs of your applications. Java is also an OO language with well designed application frameworks / libraries, and a large pool of qualified developers. That makes your application cheaper to maintain over the long term. However, when it comes to enterprise features the single most important differentiators between Java EE and competing solutions (e.g., from Perl, PHP, .NET, to the latest hype such as Ruby On Rails), is that Java EE servers are easy to cluster. You can simply add more hardware to handle more requests, without re-coding the application.



Easy clustering allows you improve your web application performance today, and more importantly, it can future proof your applications for new Internet architectures, and even new business models. For instance, AJAX (Asynchornous JavaScript and XML) technology significantly improves the usability of web applications. But to refactor your existing web applications to use AJAX, you should anticipate several folds of increase in web requests. That is because an AJAX page make multiple HTTP requests to the server to partially update the page, as opposed to one request per page in traditional web applications. Another example is the proliferation of SOA (Service Oriented Architecture) web services. Web applications in a SOA infrastructure are used to deliver component services instead of final UIs. An application typically needs several web service calls across different domains to gather enough information for a web page. As a result, the machine-to-machine interaction in the SOA world generates much more web traffic than the browser-server interaction today.

In general, as we move to Web 2.0, web sites become less document-centric and more application-like. Those future needs call for more powerful application servers, and the best way to archive that is through application server clustering.

The core technology behind a clustered Java EE solution is the distributed in-memory cache. The cache is replicated on all server nodes in the cluster. Hence it acts like a cluster-wide shared memory. You can configure the cache to automatically expire old or less used entries, save its content to the disk when the JVM shuts down, and manage read/write transactions when several different server nodes access the same data concurrently. In a database driven

–continued on page 8

"Database cache is widely used in Java applications regardless whether they are clustered or not"

Michael Juntao Yuan is a member of JDJ's editorial board. He is the author of three books. His latest book, "Nokia Smartphone Hacks" from O'Reilly, teaches you how to make the most out of your mobile phone. He is also the author of "Enterprise J2ME" - a best-selling book on mobile enterprise application development. Michael has a PhD from the University of Texas at Austin. He currently works for JBoss Inc. You can visit his Web site and blogs at www.MichaelYuan.com/.

juntao@mail.utexas.edu

SYS-CON MEDIA

President and CEO:
Fuat Kircaali *fuat@sys-con.com*
Group Publisher:
Jeremy Geelan *jeremy@sys-con.com*

Advertising
Senior Vice President, Sales and Marketing:
Carmen Gonzalez *carmen@sys-con.com*
Vice President, Sales and Marketing:
Miles Silverman *miles@sys-con.com*
Advertising Sales Director:
Robyn Forma *robyn@sys-con.com*
Advertising Sales Manager:
Megan Mussa *megan@sys-con.com*
Associate Sales Manager:
Kerry Mealia *kerry@sys-con.com*

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

Production
Lead Designer:
Tami Lima *tami@sys-con.com*
Art Director:
Alex Botero *alex@sys-con.com*
Associate Art Directors:
Abraham Addo *abraham@sys-con.com*
Louis F. Cuffari *louis@sys-con.com*
Assistant Art Director:
Andrea Boden *andrea@sys-con.com*

Web Services
Information Systems Consultant:
Robert Diamond *robert@sys-con.com*
Web Designers:
Stephen Kilmurray *stephen@sys-con.com*
Wayne Uffleman *wayne@sys-con.com*

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

SYS-CON Events
President, SYS-CON Events:
Grisha Davida *grisha@sys-con.com*
National Sales Manager:
Jim Hanchrow *jimh@sys-con.com*

Customer Relations
Circulation Service Coordinator:
Edna Earle Russell *edna@sys-con.com*
JDJ Store Manager:
Brunilda Staropoli *bruni@sys-con.com*

Innovations by InterSystems

Rapid development with robust objects

Lightning speed with a multidimensional engine

Easy database administration

Massive scalability on minimal hardware

Database For Web-Based Applications.

Caché, the multidimensional database from InterSystems, can automatically project data and logic in a number of Web-centric forms, such as XML, Web Services, Java, and EJB. These unique capabilities make Caché ideal for rapidly developing Web-based applications.

Caché is the first database to seamlessly combine robust objects and robust SQL, thus eliminating object-relational mapping. Its post-relational technology delivers lightning-fast transaction processing, real-time analytics, and massive scalability on minimal hardware. It requires little administration, and incorporates a rapid application development environment.

These innovations mean faster time-to-market, lower cost of operations, and higher application performance. We back these claims with this money-back guarantee: *Buy Caché for new application development, and for up to one year you can return the license for a full refund if you are unhappy for any reason.** Caché is available for Unix, Linux, Windows, Mac OS X, and OpenVMS – and it's deployed on more than 100,000 systems ranging from two to over 50,000 users. We are InterSystems, a global software company with a track record of innovation for more than 25 years.

InterSystems CACHÉ®

Try an innovative database for free: Download a fully functional, non-expiring copy of Caché, or request it on CD, at **www.InterSystems.com/Cache11P**

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

–continued from page 6

web application scenario, the cache manages the distributed application state and optimizes database access.

First, the cache stores all the HTTP web sessions in the application. So, all the server nodes have access to the same user sessions at any time. The HTTP load balancer can direct any user to any server node. In another the word, the HTTP session cache makes the entire cluster transparent to both web users and developers, as if it is one single application server. The transparent cache increases developer productivity since it allows us to deploy applications written for a single application server to a cluster without any changes to the Java code. The cache also has good performance since it stores data in-memory. In contrast, in a Perl/PHP/Ruby based web framework, you typically have to cluster stateless application servers and save the session state information to the database. For every web request, the application has to load the user session from the database. Hence, in those non-Java EE frameworks, a clustered deployment requires significant changes in both code and database structure from the non-clustered prototype. Under the cache-less scenario, the clustered database has to be carefully partitioned by experienced architects to allow fast access to the frequently requested user session data. In a nutshell, the distributed cache is a critical component that makes Java EE web applications easy to cluster.

Besides HTTP sessions, the distributed in-memory cache can store any transient application state information that do not need to be permanently stored in the database. Examples of such transient data include workflow processes and status, real-time server statistics or sensor readings, incomplete transactions, and unprocessed communication data. The application state cache is especially useful in SOA applications where information displayed to the end users actually come from a variety of different sources that have their own persistence mechanisms. For instance, a web application can display the current weather, traffic condition, stock price, or promotional deals. All those information come from external SOA service components and are transient in the web application. They are best managed in the in-memory cache. With a generic distributed cache, multiple Java EE application servers can be deployed in a computing grid and share computing tasks transparently to the user.

For non-transient application data that do need to be stored in the database, the cache improves the overall application performance. The cache groups multiple database update operations into a single batch for a single trip over the network. It also stores frequently read database objects and queries, and provide the application fast in-memory access to those objects. In both cases, the cache reduces the database load, and more importantly, reduces the slow round trips between the database server and the application server. The database cache is especially important in AJAX applications since the it sends in many more incremental updates to the database (i.e., an auto-saving text editor) and asks for much more frequent database reads to incrementally updates the UI (e.g., an auto-complete text field), compared with traditional document centric web applications.

Database cache is widely used in Java applications regardless whether they are clustered or not. But for clustered applications, a distributed database cache is crucial to archive the best performance. The distributed cache's content is updated when any server node updates the database. Common Java persistence frameworks (e.g., Hibernate and EJB 3.0) allow cache to be transparently plugged in underneath their abstract data access API. You do not need to make any change to your data access code in order to take advantage of the cache. You can simply configure the cache service in your Java EE container.

“Easy clustering allows you improve your Web application performance and more important, it can future proof your applications for new Internet architectures and even new business models”

The transparent distributed cache makes it easy to develop and deploy Java applications in clustered environments. In fact, most Java web applications, even prototypes, are already cluster-ready. You simply need to choose a cache implementation, configure it using XML files, and load it together with your application in a Java EE container. Unfortunately, there is no standard cache API or XML configuration files in Java EE yet (JSR-107 is an effort in this direction but it seemed dormant at this moment). The good news, however, is that you have wide variety of choices for distributed and transactional cache implementations, from the open source JBoss Cache to the commercial Tangosol. Most those cache frameworks can be used in almost any Java EE application server container. Check them out and future proof your Java EE applications!

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

Java on the desktop is back!

RCP Developer™

eclipse READY™

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

STAND ON THE SHOULDERS OF GIANTS

Create blazing fast rich Java clients.

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

Build rich Java client applications faster.

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

Deliver a more consistent user experience.

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

Fully exploit the Eclipse Rich Client Platform.

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

—Mike Milinkovich, executive director of the Eclipse foundation

RCP Developer™

SWT Designer™

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

WindowTester™

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

RCP Packager™

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

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

instantiations
the development tools experts

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

The leader in Eclipse RCP development tools

eclipse FOUNDATION MEMBER

Deadlocks in J2EE

Conventions to follow to avoid the problem

by Michael Nonemacher

Most non-trivial applications involve high degrees of concurrency and many layers of abstraction. Concurrency is associated with resource contention and an increase in deadlock conditions. The multiple layers of abstraction make it more difficult to isolate and fix the deadlock conditions.

Generally, a deadlock happens when two or more concurrent threads of execution each hold a resource and request another resource. Since neither one continues until it acquires the resource, we say each specific thread is blocked; if each thread is blocked on a resource that is held by another thread in the same group, we say the group of threads is deadlocked.

In this article, we'll discuss two broad categories of deadlocks that occur in typical non-trivial J2EE applications: "simple" database deadlocks and cross-resource deadlocks. Although the discussion is based on J2EE, it also applies to other technology platforms.

Database Deadlocks

In a database, one connection can block another if it's holding database locks needed by the other. If two or more connections are blocking each other, none of them can proceed, and they're deadlocked.

Database deadlocks can be tricky because the locks involved often aren't explicit. Typically, updating a row implicitly requires locking that row, doing the update, and then releasing the lock when the enclosing transaction is committed or rolled back. Depending on the database platform, the configured isolation level, and any query hints, the acquired lock can be fine- or coarse-grained and block or not block other queries on the same row, table, or database.

The locks acquired can depend on the internally generated query plan that can change over time as the data size and distribution changes, so a query that acquired one set of locks in one environment can try to acquire a completely different set of locks in another. The database is free to escalate its locks when necessary – instead of locking 10 rows on the same data page, it may choose to lock the entire page, which may block reads or writes to rows that don't need to be locked.

Depending on the database schema, a read or a write can require traversing or updating multiple indexes, validating constraints, executing triggers, etc., each of which can introduce more locks. Further, other applications may be hitting some of the same database schema objects and acquiring locks differently than your application ever does.

All of these factors conspire to make it practically impossible to eliminate the possibility of database deadlocks. Fortunately, database deadlocks are generally recoverable: the database will notice the deadlock, forcibly kill one of the connections (typically the one that has done the least work), and roll back its transaction. This releases all of the locks associated with the terminated transaction, which should allow at least one of the other connection(s) to acquire the locks on which they were blocking.

Because of this typical deadlock-handling behavior, it often suffices simply to retry the entire transaction in the case of a database deadlock. When a database connection is killed, an exception is emitted that can be caught by the application and identified as a database deadlock condition. If that deadlock exception is allowed to propagate out to the layer of code that initiated the transaction, that code can simply start a new transaction and redo all of the earlier work. For this strategy to be correct, the code inside the transaction must have no side-effects until after the transaction has been successfully committed. Note: You'll want to put a limit on the number of times you retry or else a particularly deadlock-prone piece of code may loop forever.

This feels a bit clunky – if something goes wrong, we just try again. However, because of the freedom the database has in the locks that it acquires, it's nearly impossible to guarantee that two threads of execution can't cause a database deadlock. At least this approach guarantees that the application behaves correctly in the rare case that the deadlock happens, and is far less clunky than asking your *users* to retry the operation.

In a J2EE application, developers can set an EJB call to use either bean-managed transactions (BMT), where the developer specifically starts and commits or rolls back the transaction, or container-managed transactions (CMT), where the application server starts a transaction before calling the method and commits or rolls back the transaction after the method completes. It would be nice if the EJB vendors supplied a *retry-on-deadlock* parameter that would do this automatically with container-managed transactions. Without this automated feature, developers end up forcing EJB calls to use bean-managed transactions just to be able to retry on deadlocks. (One of the disadvantages to making your EJB calls use bean-managed transactions is that it's not obvious how to get the same semantics as a container-managed transaction with "RequiresNew" or "NotSupported." See http://www.onjava.com/pub/a/onjava/2005/07/20/transactions.html for how it can be done.)

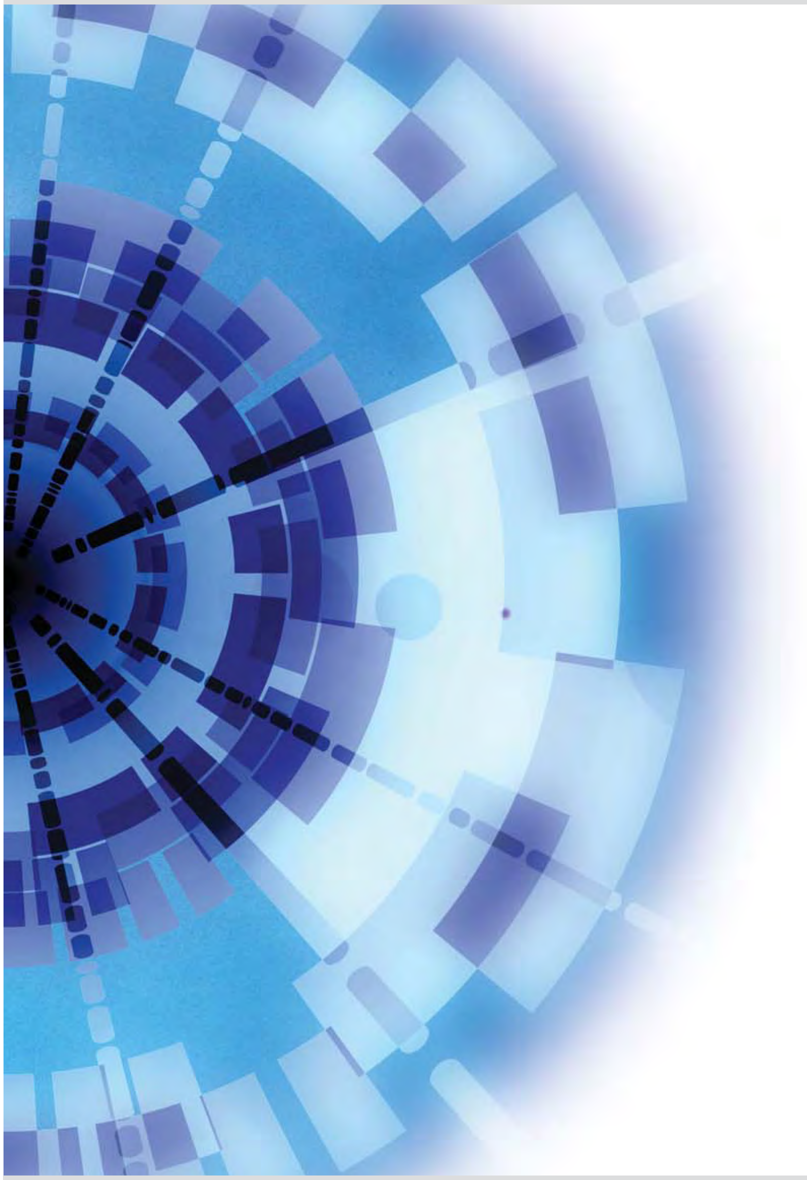
The specifics of how often you'll run into deadlocks and which locks will block other threads depend a lot on your database platform, hardware, database schema, and queries. In databases that use lock-based concurrency control, like MSSQL, uncommitted writes can block reads and uncommitted reads can block writes, which makes them more deadlock-prone. In MVCC (multi-version concurrency control) da-

Michael Nonemacher is a lead software engineer for Lombardi Software. He has worked with Java since 1997, focusing on server-side database interaction and concurrency in Web-based enterprise applications.

michael.nonemacher@lombardisoftware.com

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

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



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

InfoWorld 2006 TECHNOLOGY OF THE YEAR AWARD

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

TIBCO®
The Power of Now®

tabases like Oracle, uncommitted writes won't block reads – the read will simply see the old version of the row. That can introduce other problems but doesn't create as many deadlock opportunities. Familiarize yourself with these database locking schemes and be aware of which type you are using.

There are a number of good references on how to find, fix, and avoid database deadlocks, but none of them will eliminate the possibility of deadlocks.

Cross-Resource Deadlocks

When your deadlock condition isn't completely contained within a database, it can be much more difficult to track down. Since the database is aware of the locks held and requested, it can detect deadlocks that are entirely contained in a database; also, because database transactions provide a nice boundary of what things should and shouldn't be atomic, the database can simply roll back a transaction to recover from a deadlock. Deadlocks that are in other environments, like the JVM, or deadlocks that span environments can be more dangerous because the environment can't (or doesn't) detect them and try to recover. Worse, these deadlocks can have a compounding effect – if two threads are deadlocked while holding some set of resources, any other thread that tries to access one of those resources also becomes blocked, along with any resources that thread has already acquired. Often, these deadlocks can be difficult to track down, but some familiarity with the general patterns usually helps to identify and fix the problem.

There are a few questions to ask when an environment gets into a suspected deadlock condition. The answers to these questions will indicate which of the following scenarios you're dealing with, if any, and give more detail about how to fix the underlying problem. Some of the key things to ask are:

1. Which threads are involved and what are their call stacks? This can take some detailed analysis to separate the threads that are actually deadlocked from those that are simply blocked by the deadlocked threads.
2. Does this deadlock happen consistently in a particular code path (every time a particular operation is performed), or is it dependent on two or more code paths executing at the same time?
3. What database connections are involved? What database locks does each connection hold, and what database locks are each connection trying to acquire? Which JVM thread does each database connection correspond to?

The section below illustrates three commonly occurring cross-resource deadlock scenarios.

Cross-Resource Deadlock #1: Pool Exhaustion with Escalating Clients

The first deadlock scenario we'll look at happens only under load, when a resource pool is too small and each thread needs more than the available resources from the pool. For example, consider an EJB call that uses a database connection, then makes a nested EJB call that uses a separate database connection from the same connection pool. This will happen if the nested EJB call is declared as "RequiresNew," for example.

Under normal load, or with a sufficiently sized connection pool, the EJB call will get a database connection from the pool, and then call the nested EJB. The nested EJB call will get another database connection from the pool, commit the inner transaction, and return the connection to the pool. The outer EJB call will then commit its transaction, and return its connection to the pool.

However, suppose the connection pool has a maximum size of 10 connections, and there are 10 concurrent calls to the outer EJB. Each of those threads acquires a database connection, emptying the pool. Now, they each try to make the nested EJB call, which needs to acquire a second database connection. None of them can proceed, and none of them will give up their first database connection, so all 10 threads are deadlocked.

When investigating a deadlock of this type, you'll see a large number of threads in your thread dump waiting to acquire resources, and the same number of active database connections, all idle and unblocked. If you can inspect the connection pool at runtime while the application is deadlocked, you should be able to verify that it's actually empty.

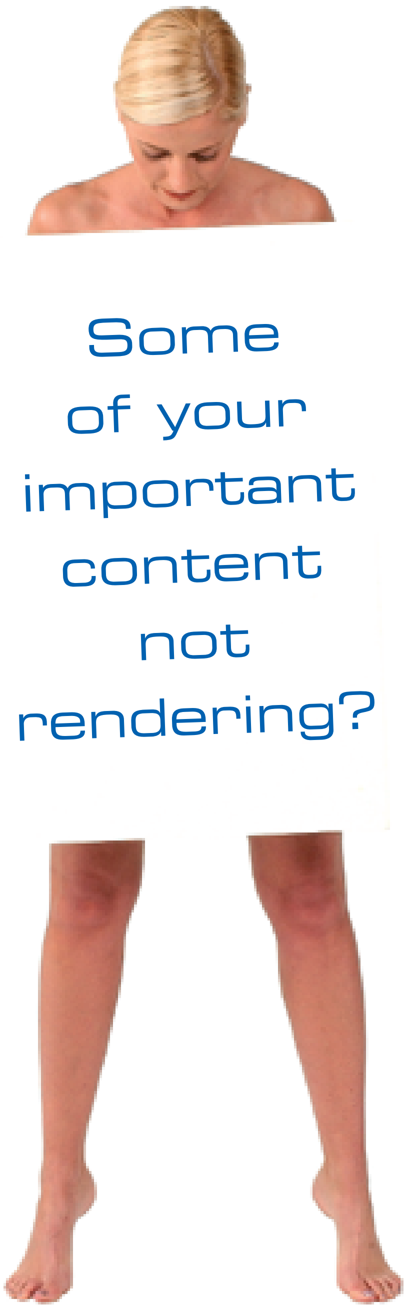
The fix for a deadlock of this type is either to increase the size of the connection pool or to refactor the code so that a single thread doesn't require as many database connections at the same time. If the maximum number of database connections required by a single thread is M, and the maximum number of possible concurrent calls is N, the minimum number of connections required in the pool to prevent this problem is (N*(M-1))+1. Alternatively, you could set up the inner EJB call to use a different connection pool, so that even if the outer call's connection pool is empty, the inner calls will be able to proceed using their own connection pool.

Cross-Resource Deadlock #2: Single-Thread, Multiple Conflicting Database Connections

The second cross-resource deadlock scenario can also arise when making nested EJB calls on the same thread, although this case typically happens even in systems that aren't under load. Just as in the example above, the two EJB calls use different connections to the same database. Since the caller can't continue until the nested call completes, the caller's database connection is effectively blocked by the nested call's database connection, although the database isn't aware of this relationship. If the first (outer) connection has acquired a database lock that the second (inner) connection needs, the second connection will block indefinitely waiting for the first connection to be committed or rolled back and a deadlock arises. Since the database isn't aware of the relationship between the two connections, the database won't detect this as a deadlock.

As a concrete example, consider a data-loading EJB call. This EJB call takes in a large object and persists it to the database in several stages. As it performs the data-load, it updates a separate table that records the state of the pending data-load operation. We'd like the state-update to be visible immediately, but we don't want the loaded data to be visible in an incomplete state, so the state-update is done with a call to a "RequiresNew" EJB. Roughly, our (flawed) data-load method looks like the code in Listing 1.

In the example, the updateBatchStatus method makes a "RequiresNew" EJB



get WebRenderer™

Java Browser SDK

Desktop Edition

Embed WebRenderer into your Java client side application interface to display rich web content.

✓ Swing and AWT support
✓ Extensive DOM
✓ Plugin support
✓ Powerful API
✓ HTTP, SSL, TLS
✓ JavaScript
✓ Complete Event model

Server Edition

Automate Server Side web content reporting, printing and processing with WebRenderer Server Edition.

✓ Headless server optimized
✓ J2EE support
✓ No AWT or Swing dependencies
✓ Complete DOM support
✓ HTML to thumbnail image
✓ Print support
✓ Standards compliant rendering

To download a free 30 day trial visit www.webrenderer.com

JadeLiquid™ Software

call to actually update the batch_status database table so the status change is immediately visible, even though the effects of the current transaction aren't yet visible. The call to executeUpdate isn't an EJB call, so it executes in the same transaction as the rest of bulkLoadData.

As described, this code will cause a deadlock even in the absence of concurrency. When bulkLoadData calls the executeUpdate method, it updates an existing database row, which involves acquiring a write-lock on that row. The nested EJB call to updateBatchStatus will execute on a separate database connection and try to execute a very similar query, but it will block because it can't acquire the necessary write-lock. From the database's perspective, this second connection will be allowed to proceed as soon as the first connection's transaction is committed or rolled back, which is why the database doesn't detect this as a deadlock. The VM, however, won't allow the call to bulkLoadData to complete before each call to updateBatchStatus completes, so we have a deadlock.

This example shows one update blocking another update, so it causes a deadlock on any database. If the initial update query was a simple select query instead, this example would only cause deadlocks on databases that use lock-based concurrency control, where one connection's read-lock can block another connection from acquiring a write-lock. In any case, a deadlock of this type is neither timing-dependent nor load-dependent, and the thread dump will show one Java thread waiting for a database response, but that thread will be associated with two active database connections. And one of those database connections will be idle, but blocking the other connection.

This scenario has many subtle variations, where more than one thread and more than two database connections may be involved. For example, the outer EJB call's database connection may have acquired a database lock that blocks another unrelated database connection from proceeding, but that unrelated database connection has acquired a lock that blocks the nested EJB call's database operation. This particular case is timing-dependent, and will show several Java threads waiting for database responses. At least one of those Java threads will be associated with two active database connections.

Cross-Resource Deadlock #3: JVM Locks Crossed with Database Locks

A third deadlock scenario occurs when mixing database locks with JVM locks. In this case, one thread holds a database lock and is trying to acquire a JVM lock (trying to enter a synchronized block), and another thread holds that JVM lock and is trying to acquire a database lock. Again, the database sees one connection blocking another, but nothing is stopping that connection from proceeding, so it won't detect a deadlock. The JVM sees one thread inside a synchronized block and another trying to enter, so even if the JVM could detect deadlocks and handle them somehow, it wouldn't detect this case.

For an example that illustrates this deadlock scenario, consider a simple (flawed) read-through cache. This cache is a HashMap backed by a database table. If a cache hit occurs, it will simply return the value from the HashMap; on a cache miss however, it will read the value from the database, add it to the HashMap, and return it as shown in Listing 2.

It's a simple read-through cache. Note that the get() method is synchronized, since we access a non-thread safe container, and we want the containsKey/put combination to be atomic in the case of a cache miss.

This cache is fairly straightforward: the contract is that if we change the data in the table backing the cache, we should call clearCache(), so the cache can avoid handing back stale data. The resulting cache misses will repopulate the cache appropriately.

Let's now consider some code that changes this data and clears the cache:

```
public void updateData(String key, String
value) {
 executeUpdate("update cache_table set
value='" + value +
   "' where key='" + key + "'");
 SimpleCache.getInstance().clearCache();
}
```

In a simple case, this will work with no problems. However, on a database that uses lock-based concurrency control, the query in updateData will prevent the select query in queryForValue from executing, because the update statement will have acquired a write-lock that prevents the select query from acquiring a read-lock on the same row. If the timing is just right, one thread can try to read a given value out of the cache and get a cache miss while another thread updates that value in the database. If the database executes the update statement first, it will block the select statement from continuing. However, the thread executing the select statement came through the synchronized get method, so it has acquired the lock on the SimpleCache. For the thread in updateData to return, it has to call clearCache(), but it can't acquire the lock (clearCache() is synchronized).

When dealing with an instance of this scenario, there will be one Java thread waiting for a response from the database, and one waiting to acquire a JVM lock. Each thread will be associated with a database connection, with one connection blocking the other. The fix is to avoid doing database operations while holding JVM locks: the get() method of our SimpleCache could be rewritten like this:

```
public Object get(String key) {
 synchronized(this) {
  if (cache.containsKey(key)) {
   return cache.get(key);
  }
 }
 Object value = queryForValue(key);
 synchronized(this) {
  cache.put(key, value);
 }
 return value;
}
```

Since we now know that this deadlock case can happen, we can add a check to our queryForValue method to try to avoid the deadlock condition by using Thread.holdsLock():

```
private Object queryForValue(String key) {
 assert(!Thread.holdsLock(this));
 return executeQuery(...);
}
```

While Thread.holdsLock() can be useful in this case, it only works if we know which locks we're specifically worried about. It would be useful to have a similar method that could determine whether the current thread

Bridge the Gap Between Windows and Java

Give your users the freedom to use Microsoft Windows based reporting, analysis, and development applications against your Java data source

OpenAccess™ SDK lets you quickly implement a custom ODBC driver with full SQL support for any data source or application.

Key OpenAccess™ SDK Features & Benefits

- Write the ODBC driver with SQL support in just weeks - 99% of the required code is contained in the SDK.
- Use Java to implement the data source specific code.
- Guaranteed to work with ODBC compliant applications – hundreds of organizations have used OpenAccess™ to set their data free.
- Fully Unicode enabled for globalization.
- When required easily add JDBC, OLE DB, or .NET drivers.
- Designed from ground up for reliability, scalability, and portability.

OpenAccess can also be used to add ODBC support on top of your JDBC driver and to add a security layer in between a client application that uses ODBC or JDBC and the data source. Please call us or visit our web site to learn how different problems have been solved using OpenAccess.

To learn more about OpenAccess™ SDK for Java or to try it out, go to **www.OpenAccessSoftware.com/JDJ or call 1.888.805.ODBC**

OpenAccess SOFTWARE

OpenAccess Software, Inc. 2001 Gateway Place, Suite 100 San Jose, CA 95110 www.OpenAccessSoftware.com holds *any* JVM locks. Then, any piece of code that made any kind of RPC call, database access, etc., could throw an exception or log a warning to indicate that these operations can be dangerous while holding a JVM lock.

Note: even though we've fixed the deadlock problem in this example, it's still flawed because the cache is cleared before updateData's transaction has been committed. If a cache miss happens after the clearCache call but before updateData's transaction is committed, the cache will load the old data because the new data isn't visible yet. The fix here is to clear the cache only after the changes have been committed. Observe that this would only happen in an MVCC database; in a lock-based database, the pending update would block the cache's read, so the cache would read the correct value after the update's transaction was committed.

Rules of Thumb

These guidelines should help you avoid these problems, or at least diagnose and fix them if they do happen.

- Keep your transactions short and simple.
- Understand the locking behavior of your database (and your transaction isolation level).
- Assume any database access has a chance of failing with a database deadlock, and retry correctly.
- Don't update any non-transactional state (in-memory state, caches, etc.) until after the transaction has completed.
- Make sure resource pools are large enough for your peak concurrency.
- Try not to acquire multiple resources at the same time. If you must, acquire them in the same order each time.
- Learn how to get a full thread dump from your application server and a list of database connections from your database (including which connections are blocked by each), and how to know which Java thread each active database connection is associated with. The easiest way to understand the mapping between Java threads and database connections is to add some logging to your connection pool access code.
- When making nested EJB calls, understand which ones will use the same database connection as the caller, and which ones will use new connections. Even if the nested call runs in the same global transaction, it may still use a different database connection, and this can cause cross-resource deadlocks.
- Avoid making database calls and EJB calls, or doing other off-JVM operations while holding JVM locks. Use assert(!Thread.holdsLock(...)) if there are specific JVM locks you're concerned about to prevent future code changes from violating this rule unintentionally.

Conclusion

Cross-resource deadlocks in a J2EE application can be a big problem – they can cause the entire application to grind to a halt. They can also be difficult to isolate and fix, especially if developers aren't familiar with how to analyze a deadlocked environment. The scenarios we've discussed should help you understand a few common deadlocks, and give you some idea of what to look for. Even better, the rules of thumb we've outlined should give you a few conventions to follow in your code to avoid problems like this altogether.

Resources

- How to get "RequiresNew" and "NotSupported" transaction behavior when using bean-managed transactions (BMT): http://www.onjava.com/pub/a/onjava/2005/07/20/transactions.html
- Database deadlocks in J2EE applications: http://www.theserverside.com/patterns/thread.tss?thread_id=14482
- WebSphere documentation's discussion of scenario #1: http://publib.boulder.ibm.com/infocenter/wasinfo/v5r1/index.jsp?topic=/com.ibm.Websphere.base.doc/info/aes/ae/cdat_conpool.html
- WebSphere discussion of connection sharing across EJB calls: http://www-128.ibm.com/developerworks/Websphere/library/techarticles/0404_tang/0404_tang.html

Listing 1

```
public void bulkLoadData(DataBatch batch) {
 int batchId = batch.getId();

 // Since this executeUpdate call doesn't happen in a separate
 // transaction, it wouldn't be visible anyway, but the effect is
 // far worse: a cross-resource deadlock.
 executeUpdate("update batch_status set status='Started' " +
  "where batch_id=" + batchId);
 validateData(batch);
 updateBatchStatus(batchId, "Validated"); // RequiresNew EJB call
 loadDataStage1(batch);
 updateBatchStatus(batchId, "Stage 1 complete"); // RequiresNew
EJB call
 loadDataStage2(batch);
 updateBatchStatus(batchId, "Stage 2 complete"); // RequiresNew
EJB call
 finalizeDataLoad(batch);
 updateBatchStatus(batchId, "Complete"); // RequiresNew EJB call
}
```

Listing 2

```
public class SimpleCache {

 private Map cache = new HashMap();

 public synchronized Object get(String key) {
  if (cache.containsKey(key)) {
   return cache.get(key);
  } else {
   Object value = queryForValue(key);
   cache.put(key, value);
   return value;
  }
 }

 private Object queryForValue(String key) {
  return executeQuery("select value from cache_table " +
   "where key='" + key + "'");
 }

 public synchronized void clearCache() {
  cache.clear();
 }

 // other methods omitted for brevity
}
```

Fiorano SOA™ 2006

The Quickest Path to an SOA

Fiorano provides the industry's first framework enabling developers to rapidly create SOA applications in well-formed J2EE code.

Key Features:

- JCA-compliant Service-component Framework
- JMS based messaging
- BPEL support for composite components and applications
- Synchronous and Asynchronous Services in a single framework
- Support for multiple languages : Java, C, C++, C# and others

With Fiorano you can:

- Directly deploy Service Components and BPEL processes to any standard J2EE container
- Rapidly deliver ACID transactions across front-end (JSP) and backend processes (Databases, ERPs, etc.)
- SOA enable your existing J2EE application (EJB, JCA, JMS) without reengineering or code-generation
- Easily debug complex flows spanning multiple applications and middleware
- Perform runtime service deployment, service updates and application changes

Meet us at:

JavaOne Sun's 2006 Worldwide Java Developer Conference

May 16-19, 2006
Moscone Center - San Francisco

Download your copy of Fiorano today!

http://www.fiorano.com/downloads

Fiorano®

Enabling Change at the speed of thought

Look Mom, No Application Servers, Look...MOM!

by Yakov Fain

In the unlikely event that you're not familiar with my gas station, you can find my previous essays at http://jdj.sys-con.com/read/category/1142.htm. Recently, I've conducted a small survey among my truck drivers. I asked them just one question: "What do you think of application servers?" The most popular answer was, "I don't need no stinkin' application server." And truck drivers usually know what they're talking about!

You may think that now I'll start selling one of the popular application frameworks. Wrong! The idea of these frameworks was nice: get back from complex containers to programming POJOs. But while trying to provide alternatives to container services, each of these frameworks ran into the short-blanket syndrome: something is always sticking out. XML is sticking out big time!

To simplify Java programming, developers are paying the high price of adding unmanageable amounts of XML descriptors, mappings, wirings, and other plumbing. Twenty years ago .ini files were widely used, 10 years ago .properties replaced them, and after the Y2K problem was solved, people were bored and started investing their time in XML.

It started quite innocently: XML is better than HTML, then DTD came about, XSLT, XPATH, XQuery, XML farms, XML processing hardware, and on, and on, and on. Basically, the proper way to name today's POJO is XPOJO. You know what I mean. Hopefully, Java annotations will slow down the X-hype. I'm already using Java 5 in my agile business, and as soon as Mustang is released, big corporations will switch to Tiger.

Let's return to the main subject. One of my readers asked if I was planning to write a piece on how to select an application server. I answered, "No, because I'm not sure if they're needed."

I believe that at least a half of today's business applications don't need one. In my gas station, I'm going to implement a set of client/server applications with rich clients talking to each other using Java messaging, and most likely, I'll need to implement an Enterprise Service Bus (ESB) to communicate with a couple of old applications that I inherited from the former owner of my gas station.

My newly developed applications will use rich application clients living in a Web browser, but they'll run in their own virtual machines (no AJAX by the pumps!). I've already started working on a more serious manuscript on RIA (see http://theriabook.com). To make a long story short, I'm planning to use/implement SOA, RIA, MXML , JWS, JMS, MOM, JNDI, ESB, JTA, DAO, JDBC, DBMS, and a Web Server. Raise your hand if you know

how to spell out each of the acronyms used in this article so far.

I know, I know. Ten years ago, people who could spell out VB and SQL were able to find a nice paying job in a heartbeat. Forget about it. I can recommend a handy Web site called acronymfinder.com. In some cases it gives you too many versions for your acronym, but key in JTA and you'll find that it stands for Java Transaction API (it's right above Japan Toilet Association). Since I'm not going to use the J2EE application server, I have to find transaction support somewhere else.

Consider the use case of a typical business application without an application server. The front-end reacts to various events by putting messages (Java value objects or key-value pairs) into a remote message queue (orders, requests, etc.). A server-side JMS listener picks them up from the queue and starts business processing the messages received. This is where transaction support comes in handy. For example, from a business perspective saving a customer's order in a database and making a JMS call for further processing can be considered one unit of work, which has to be rolled back if any of these actions fails. That's why a JMS listener has to be able to begin, commit, and roll back transactions.

Communicating with the database can be done using a simple DAO/JDBC combo. Create a tiny .properties file with a dozen parameters (the data source to use, maybe some external URLs, and a couple of others). That's it. This is what I call real POJO programming. Did I miss anything? Oh yeah, JMS is just an API, so we need a transport/storage for our messages, and this is what Message-Oriented Middleware (MOM) is for.

While there are several Open Source JMS/MOM implementations on the market, I'm planning to use ActiveMQ from Logic Blaze. The product has been on the market for years, has good reputation, supports JTA/XA, and has a large user community. Look at the ActiveMQ list of features, and you'll see that it offers more than any small business needs. Another pro is that the same vendor offers an Open Source JBI-based ESB implementation called ServiceMix. But this is a topic for future discussions. And the best part is that this middleware is available for free.

Not everyone may like this architecture, but let me remind you, we're talking about a small business here. One of my readers sent me an e-mail asking, "What if this application is supposed to be used by thousands users?" I wish… This is just a gas station! When you're architecting your next application, try not to get into a Google/Amazon state of mind unless you work for them. If you're not dealing with thousands of users, just use a simple Data Access Object, write a couple of SQL statements here and there, and make a dozen JDBC and JMS calls. And MOM will be a good, reliable foundation for creating loosely coupled applications with robust transactions, persistence, and failover support.

Yakov Fain is a senior technical architect at BusinessEdge Solutions, a large consulting and integration firm. He authored several Java books, dozens of technical articles, and his blog is hugely popular. Sun Microsystems has nominated and awarded Yakov with the title Java Champion. He leads the Princeton Java Users Group. On Sundays, Yakov teaches Java in NYC.

yakovfain@sys-con.com

Endless re-working on your reports?

Empower your users to create their own reports!

IntelliVIEW is designed to allow users to create their own reports, giving you the time to focus on strategic activities.

- Designed for non-technical users
- Zero-setup analytics
- Ad hoc reporting & analytics
- Offline analytics
- WYSIWYG printing

To know more visit **http://www.intelliview.com/jdj**
Email: sales@intelliview.com
Call toll-free 1-866-99IVIEW

IntelliVIEW
Reporting & analytics made easy!

product of Synaptris

What Issues to Look Out For as You Move from Java to Web Services

by Adam Kolawa

Legacy integration is one of the biggest challenges in building Web Services

The creation and popularity of Web Services are growing rapidly in every industry. With this continued growth, more and more programmers find themselves writing code that, even if it's not currently packaged as a Web Service, will eventually be exposed as one.

As more enterprises move toward an e-business strategy, communication and integration of existing information systems is key. When integrating existing information systems with Web Services, enterprises will usually face one of the following two scenarios:

- The enterprise information system is comprised of legacy systems. To offer services and share data with business partners, customers, and other information systems, businesses must update these legacy systems with current technology and expose them as Web Services.
- The enterprise information system may have middleware already in place, but this middleware needs to be exposed as Web Services.

Regardless of which of the above scenarios a business may face, developers have the option of two distinct approaches to exposing information systems as Web Services – Sun Microsystems' Java 2 Platform Enterprise Edition (J2EE) and Microsoft's .NET platform. Although both architectures offer a great deal of technology and standards, the strengths of Java, at least on the surface, seem to outweigh those of .NET.

However, despite Java's strengths, the devil is in the details – there will always be a number of drawbacks that developers must face when building and deploying Web Services, regardless of architecture. This article will discuss the strengths that Java has over .NET and address some of the pitfalls that developers should be wary of when exposing legacy systems and middleware as Web Services.

Exposing Legacy Systems as Web Services

Many of today's large businesses have existing legacy systems written in different languages, such as COBOL and C++. These companies have spent enormous sums of money and have collected huge amounts of data maintaining these legacy systems. Therefore, it's crucial that companies find a quick and efficient way to preserve and reuse these legacy systems to integrate and expose them as Web Services. This legacy integration is often one of the greatest challenges to tackle when building a Web Service. However, Java provides a relatively simple solution for legacy integration.

The J2EE Connector Architecture (JCA) can be used to integrate legacy systems. The JCA is a specification for creating resource adapters that understand how to communicate with existing legacy systems, such as those written in COBOL, C++, etc. These resource adapters are reusable in any container that supports JCA. Currently there are a large number of major vendors in the marketplace that produce adapters and support JCA.

.NET also offers legacy integration through the Host Integration Server 2000. However, there's limited connectivity to legacy systems through the Host Integration Server because there's little other support for it besides Microsoft. Since so many vendors already support JCA and offer resource adapters, integration with legacy systems becomes much easier.

Java's portability plays a large role in developing Web Services too. The programs created in Java are portable in a network and can run on various platforms and operating systems such as Win32, Unix, and mainframe systems. This is a big advantage since most e-businesses have clients and customers that exist on a variety of platforms.

On the other hand, .NET's portability is at best limited since it only runs on Windows machines. Although the Mono project is attempting to create an Open Source implementation of .NET, the status of the project is still uncertain. For example, Microsoft has recently applied for a patent on the .NET Framework classes. Regardless of Microsoft's true intentions, the question still remains – how much of the .NET Framework will Microsoft allow to be supplied on other platforms?

Exposing Middleware as Web Services

Besides the integration of legacy systems, another scenario that an enterprise may face is the integration of middleware as a Web Service. The intermingling of Web Services and Java is only natural, since services communicate with other services via middleware applications and most enterprise systems today already have middleware written in Java. If this is the case, then the exposure of Java middleware as Web Services is relatively easy with the use of any Java library.

A number of third-party libraries (such as Apache Axis) are readily available to accelerate the development of Web Services. These libraries store

Dr. Adam Kolawa is cofounder and CEO of Parasoft, a vendor of automated error-prevention software and services based in Monrovia, CA. Dr. Kolawa, who is the coauthor of Bullet-proofing Web Applications (Wiley, 2001), has written and contributed hundreds of commentary pieces and technical articles for publications such as The Wall Street Journal, CIO, Computerworld, Dr. Dobb's Journal, and IEEE Computer. He has also authored numerous scientific papers on physics and parallel processing. He holds a PhD in theoretical physics from the California Institute of Technology.

ak@parasoft.com

object-oriented routines and class definitions that can be "called" instead of having to write new code or make any modifications to the original legacy environment – a process that can cause innumerable complications.

Although companies can tap the strengths of Java without having to reinvent all of their existing systems, Java isn't the cure-all it may seem. Using Java to expose legacy systems and middleware as Web Services isn't difficult to do, but it is difficult to do right.

Issues of Concern

Despite the seemingly simple process of converting Java classes into XML for Web Services, it's still more difficult than developers realize. Regardless of architecture, be it Java or .NET, there will always be a number of details that developers must consider when building and deploying Web Services. And as said before, the devil is in the details. The following illustrates some examples of these details.

SOAP Message Structure

If you're trying to integrate systems of disparate backgrounds, you must resolve the mismatches of the SOAP message structures in both systems. For example, if you're integrating a Java implementation with a .NET implementation, the Java implementation may use a RPC style with SOAP encoding, while the .NET implementation may use a document style with literal encoding.

This difference has to be resolved to ensure interoperability. Using literal XML (no encoding) instead of SOAP encoding may be of greater benefit since SOAP encoding interferes with schema validation. Indications that the industry is abandoning SOAP encoding can be seen from its omission from the WS-I basic profile.

The question of RPC versus document style isn't as clear-cut. Microsoft's .NET prefers document style and most other toolkits prefer RPC. From a schema validation standpoint, document style is superior. Furthermore, by abandoning SOAP encoding, some of the benefits of RPC style are diminished. However, RPC is common enough that integration projects will often involve some RPC services. Keep in mind that RPC messages have an additional "wrapper" element that's the name of the operation.

Also, although the message declarations in the WSDL are, in theory, not specific to a particular binding and don't commit to RPC versus document style, in practice there's a strong correlation between the way messages are declared and the bindings that are used. A common pattern is that messages with parts that are declared in terms of elements are intended to be used bound to a document style while those with parts declared in terms of types are intended to be used by the RPC style.

Example message declaration (intended for document style):

```
<message name="echoStringSoapIn">
    <part name="parameters" element="s0:
echoString" />
 </message>
 Example message declaration (intended for
rpc style)
  <message name="echoStringSoapIn">
    <part name="parameters" type="xsd:
string" />
  </message>
```

Serializing and Deserializing

Serializing and deserializing is the process by which data is converted from programming language-specific data structures (such as Java classes) to and from SOAP-compliant XML. For different machines to understand the serializer/deserializer translations, you must decide how to lay out the Java objects in each machine. The meaning an object has in one machine may not necessarily have the same meaning in another machine. Dealing with primitive types is rather simple; however, when dealing with more complex types, serialization becomes more of an issue.

For example, a complex class may have several strings. If two of these strings are identical, you must decide whether to treat these strings as two references to one string, or to treat them as two separate strings. This decision ultimately affects how the strings are translated on either side of the Web Service.

Note that the reference/value distinction is only available in SOAP encoding. A common practice in Java is to form Bean classes, where accessor methods follow a simple pattern. This allows reuse of a common serializer/deserializer pair, decreasing the amount of extra code to be maintained. An example of this is the following simple class for an employee data structure:

```
public class Employee {
    private String first;
    private String last;
    private int number;
    public Employee() {
    }
    public Employee(String first, String
last, int number) {
        this.first = first;
        this.last = last;
        this.number = number;
    }
    public String getFirst() {
        return first;
    }
    public void setFirst(String first) {
        this.first = first;
    }
    public String getLast() {
        return last;
    }
    public void setLast(String last) {
        this.last = last;
    }
    public void setNumber(int number) {
        this.number = number;
    }
    public int getNumber() {
        return number;
    }
}
```

Note that the accessor methods follow a get/set naming convention that lets the bean serializer/deserializer recognize them automatically. Also, a public no argument constructor must be available for deserialization even though, in this case, the class would be better off without it from a type safety point-of-view. Keep in mind that you will either need to maintain some bean classes like this for structured data to serialize or write custom serializers and deserializers.

Conclusion

The future of e-business undoubtedly lies in Web Services. For organizations that are building and developing Web Services, or are preparing for a future of Web Services, their underlying architecture must have strong Web Services support. Java provides this support with its J2EE Connector Architecture, portability, and extensive class libraries – all of which allow for translating to Web Services without any major rewrites. Organizations building and developing Web Services will do well by moving forward in this direction, while avoiding the pitfalls that inevitably arise when working with new technologies.

Business Intelligence and Reporting with BIRT

A first experience account

by Chris Beels

Open Source Business Intelligence software is finally coming into its own, with three major players coming to the fore: JasperReports, Pentaho, and BIRT (Business Intelligence and reporting tools). Business Intelligence technology comprises solutions for delivering enterprise data in the form of customizable reports, facilitating such practices as data mining and decision support systems. Of the three leading projects, both JasperReports and BIRT are commercial Open Source; they are backed by JasperSoft and BI veteran Actuate respectively. Pentaho uses the more traditional collaborative Open Source model. One of BIRT's distinguishing features is that it's an official Eclipse project, giving it a strong endorsement by one of the leading forces in the Java community. Our team chose BIRT after some due diligence on the other two because of our prior experience with Actuate and deep appreciation of all things Eclipse.

This article presents a "first experience" approach to BIRT. We developed a simple BIRT report table of "sales broken down by region and salesperson" using a hypothetical database. The final PDF report preview is reproduced in Figure 1. BIRT has a rich set of layout capabilities that are only hinted at in this article, but will allow report designers the freedom to provide richly designed formatted reports. The actual report generation is done via BIRT's Report Engine API, which can be combined with a report designed to generate BIRT PDF reports from any Java Application.

Chris Beels is Vice President of Technology Management at Merrill Lynch and has eight years of experience in using Open Source tools like Eclipse & BIRT to do J2EE Application Design and Development. He is a Sun Certified Enterprise Architect.

chris.beels@gmail.com

Getting Started

BIRT 2.0 has a bit of a tricky install process, which makes one wonder if it's harder than needs be to "advertise" for the commercial Actuate version whose only additional virtue, other than support, is a standard complete installer. In any case, carefully follow the instructions at www.eclipse.org/birt to get it up and running in Eclipse. If installing a new version of Eclipse with BIRT, we recommend that you install it to a new directory instead of over an old Eclipse version...this caused us problems in the past.

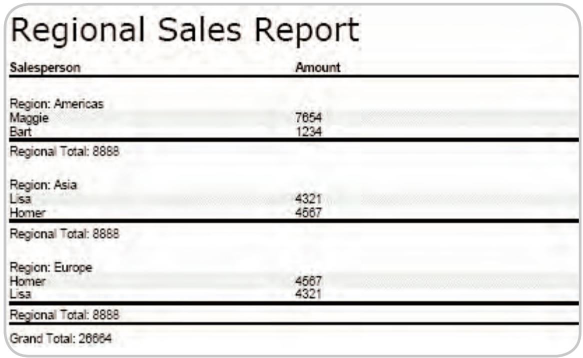
You'll want to create a small database table with some test data. You can download our sample set at the link at the end of the article. When you restart Eclipse, create a new project by clicking "New Project...Business Intelligence and Reporting Tools...Report Project." Then to create a new report, click Window...Open Perspective and select Other...Report Design, then select File...New...Report and Report templates: Blank Report to set up a new report.

Setting Up Database Access

The first step in setting up a report is telling BIRT how you want to do the JDBC access to your database. To do this, BIRT needs to know about your database driver and login information. Click the "Data Explorer" tab and right-click Data Sources...New Data Source and select JDBC Data Source. On the next page, click Manage Drivers...and Add the jar that contains the JDBC drivers for your database. Click the Drivers tab to verify that the drivers from the selected JAR file have been added then click OK. Select your driver from the Driver Class dropdown, enter the DB URL (e.g., "jdbc:mysql://localhost:3306/birt_sample"), username, and password. Click "Test Connection" to verify and now you have a source for data for your report.

Getting Data

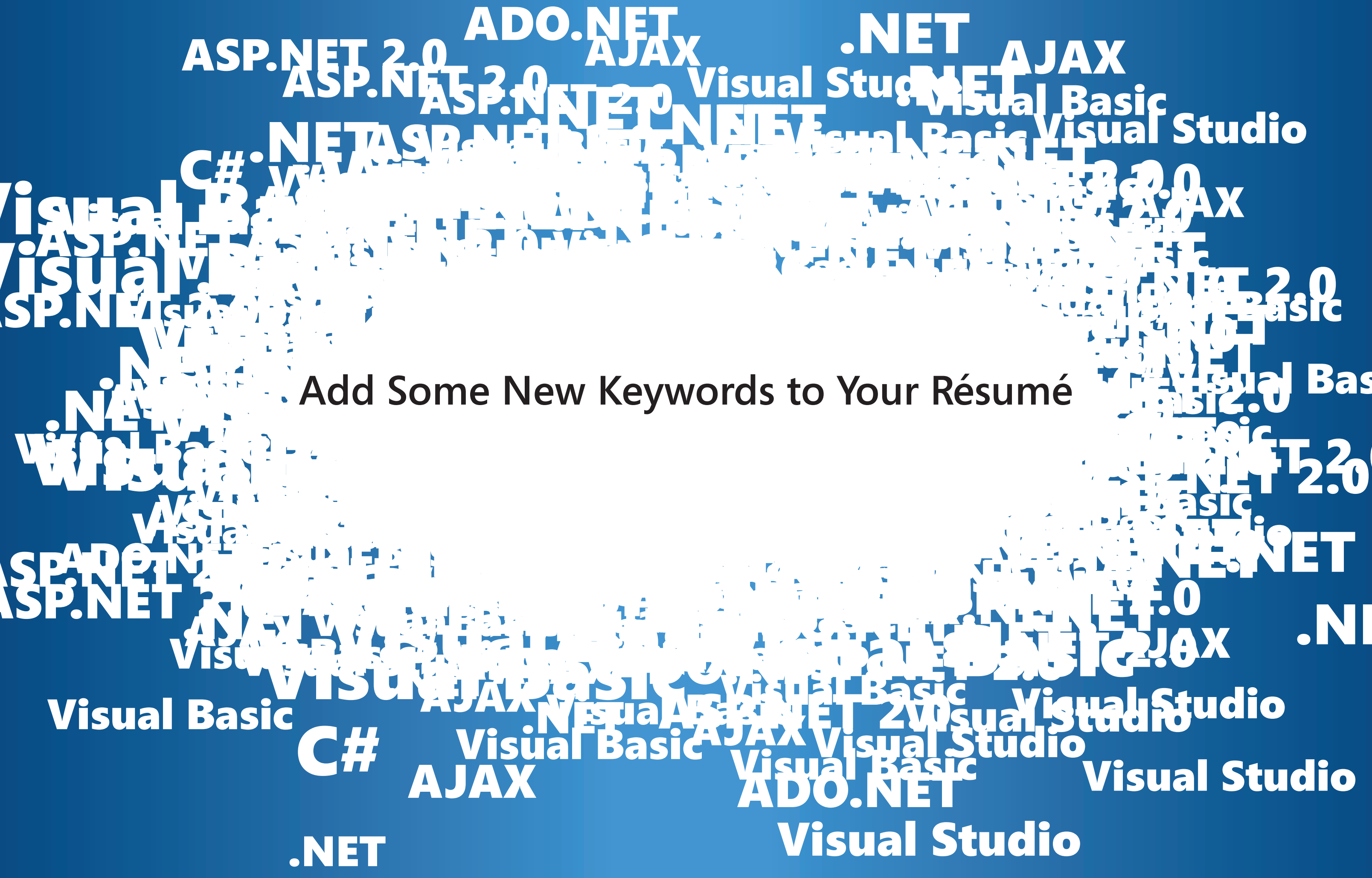
Now that you have a Source, you'll need to set up a Data Set, which is the set of data that will be used for the current report. This is typically an SQL query on your Data Source. Right-click Data Set...New Data Set and give your data set an appropriate name. Select the Data Source you just set up, if it's not already selected. When you hit OK, BIRT will bring up the Data Set wizard, giving you a window to enter your SQL query.



Regional Sales Report

Salesperson	Amount
Region: Americas	
Maggie	7654
Bart	1234
Regional Total: 8888	
Region: Asia	
Lisa	4321
Homer	4567
Regional Total: 8888	
Region: Europe	
Homer	4567
Lisa	4321
Regional Total: 8888	
Grand Total: 26664	

Figure 1 Final PDF Report



Add Some New Keywords to Your Résumé

Learn new skills by attending the ASP.NET 2.0 Webcast Series.

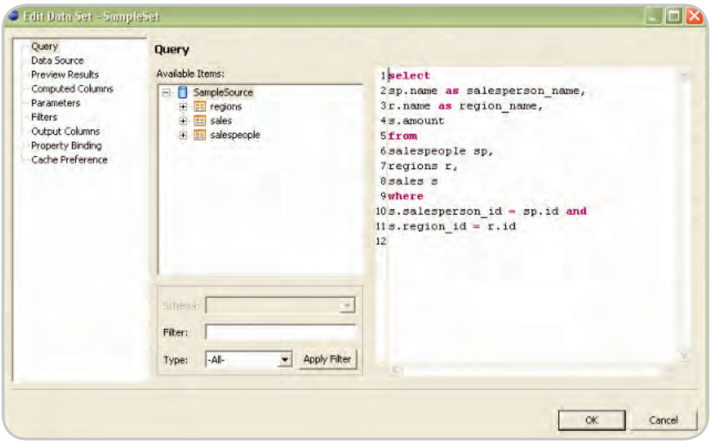
Attend 3 webcasts and you'll receive:

- Microsoft® Visual Studio® 2005 Standard Edition (Not for Resale)
- Programming ASP.NET 2.0 Core Reference (5 Chapters)
- And more—a $400 value*

> Register now at **www.learn2asp.net/webcasts.**

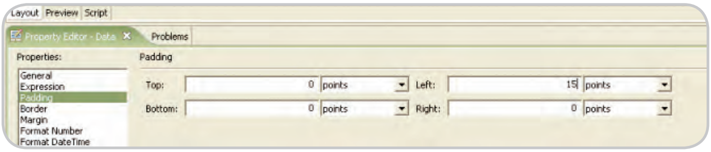
Microsoft

*Offer good in the United States and Canada only through April 17, 2006, while supplies last. This offer is available only to professionals employed in the area of software or Web development who watch three live or on-demand webcasts as part of the ASP.NET 2.0 Webcast Series. This offer is nontransferable. Microsoft is not responsible for nondelivery of gift due to incorrect postal information provided by respondent. Limit: one per customer. Please allow 6–8 weeks for delivery.



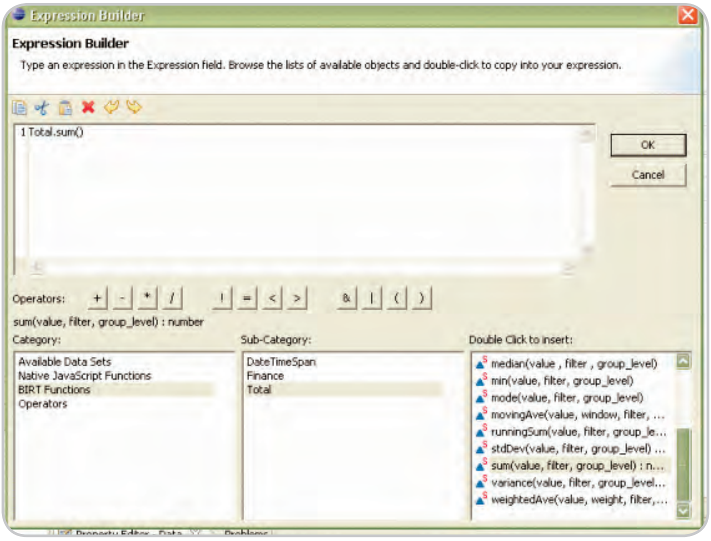
```
select
sp.name as salesperson_name,
r.name as region_name,
s.amount
from
salespeople sp,
regions r,
sales s
where
s.salesperson_id = sp.id and
s.region_id = r.id
```

Figure 2 Query editor



Top: 0 points
Left: 15 points
Bottom: 0 points
Right: 0 points

Figure 3 Row Padding Properties



Expression Builder

Type an expression in the Expression field. Browse the lists of available objects and double-click to copy into your expression.

```
Total.sum()
```

sum(value, filter, group_level) : number

median(value , filter , group_level)
min(value, filter, group_level)
mode(value, filter, group_level)
movingAve(value, window, filter, ...
runningSum(value, filter, group_le...
stdDev(value, filter, group_level) ...
sum(value, filter, group_level) : n...
variance(value, filter, group_level...
weightedAve(value, weight, filter,...

Figure 4 Total Expression Dialog

Try to choose a query that has both hierarchical and additive information. For this article, we'll assume if you have a table called "Sales"; you might write a query to get all regional and salespersons' totals (see Figure 2).

Your Data Set will now be listed under Data Sets. Expand it and you'll see the fields that BIRT has found in the database, in our case SALESPERSON_NAME, REGION_NAME, AMOUNT. Let's go ahead and create a report table using this data.

Setting Up the Report

Our business requirements for this report are to have a "Region" header, an indented list of salespeople and their individual totals, a regional total at the end of each Region section, and a grand total at the end of the report.

Let's start with the creation of the report header. Click the Palette tab in the upper left corner. The Palette shows you all the tools that will be used for creating report objects. Start with a Text object to be the header. Click "Text" and drag the object to the Report Workspace in the middle of the screen. This will bring up the "Edit Text Item" window that gives you a full set of CSS-compliant HTML tags to format your text. Switch from "Plain Text" to "HTML/Dynamic Text" to enable the HTML tags. Since this is the report header, let's make it H1. Note that BIRT inserts the open and close tags for you when you click the H1 button. Let's also choose a sans-serif font like Verdana by clicking the FONT tag. Enter Verdana for face (leave the size and color tags blank...they'll be ignored). Input a title for your report like "Regional Sales Report" inside the FONT tag.

To get the appropriate hierarchical totaling behavior for our report, we're going to use a set of "Grouped" tables. In the palette window, select a Table to hold the regional level of data. Click and drag the table object to the report workspace in the middle of the screen. Every table is made up of three components, the Header, Footer, and Details (see also the "Reporting Terminology" box). In our case, the table will have a Region header, a details section with individual salespeople, a Region subtotal footer, and a Grand Total footer. Note that only one footer is allowed for a Table object so we'll be grouping multiple tables to achieve the desired behavior.

The intuitive way to design a multi-level table is to start from the most detailed level and work your way out. In our case, that's the individual salesperson. Drag a table object into the workspace and select two columns (which will be Name and Total) and one Detail. The table will be created with Header, Detail, and Footer sections. Go to Data Explorer and drag the "Salesperson's name" into the left Detail cell. BIRT helpfully adds the database column name into the Header row, which can be edited by double-clicking. We want this level of detail to be indented from the Regional totals. To do this, click the left detail cell. The Properties tab at the bottom of the screen will change to show only properties that all selected elements share, which includes Padding, the one we need for our indentation. Set the Left Padding to 15 points (see Figure 3).

Grouping and Aggregating Data

Now we want to tell BIRT to divide up our salespeople by region. This is done by "Grouping" them, much in the same way you would using a "GROUP BY" clause in SQL. Mouse over the table object in the worksheet area and a tab marked "Table" will appear in the lower left corner of the object. Right-click on this tab to get a menu of Table-level commands. Since we're moving from more granular information (salespeople) to less granular (regions), select Insert Group. This opens the Grouping dialog box. I named the group "Region" and chose to group on the REGION_NAME field from my Data Set. I left group header and group footer checked, since the header will have the Region name and the footer will have the Region subtotal. BIRT automatically puts the grouping data element in the Group Header. I'd like to label it as a "Region" so I highlight it and click "Expression" to change this text from a simple

Complex JAVA J2EE Hosting made easy.

WebAppCabaret℠

http://www.webappcabaret.com/jdj.jsp

1.866.256.7973

JAVA J2EE-Ready Managed Dedicated Hosting Plans:

Xeon III SAMEDAY SETUP

Dual 3.2 GHz Xeons
4GB RAM
Dual 73GB SCSI
1U Server
Firewall
Linux
Monitoring
NGASI Manager

$399 *monthly*

Pentium 4 I SAMEDAY SETUP

2.4 GHz P4
2GB RAM
Dual 80GB ATA
1U Server
Firewall
Linux
Monitoring
NGASI Manager

FREE SETUP

$199 *monthly*

2nd month FREE

4Balance I

1 Database Server and 2 Application Servers connected to 1 dedicated load balancing device. Dual Xeons. High-Availability.

$1724 *monthly*

NEW! Geronimo 1.0 Hosting . Virtual Private Servers(VPS) starting at $69/month

At **WebAppCabaret** we specialize in **JAVA J2EE Hosting**, featuring **managed dedicated servers** preloaded with most open source JAVA technologies.
PRELOADED WITH:
JDK1.4 . JDK1.5 . Tomcat . Geronimo . JBoss . Struts . ANT . Spring . Hibernate
Apache . MySQL . PostgreSQL . Portals . CRM . CMS . Blogs . Frameworks
All easily manage via a web based control panel.
Details:
-All Servers installed with the latest Enterprise Linux
-Firewall Protection
-Up to 60 GB daily on site backup included at no extra charge per server.
-Database on site backup every 2 hours
-Daily off site database backup
-A spare server is always available in case one of the server goes down
-Intrusion detection.
-24x7 Server and application monitoring with automatic self healing
-The Latest Bug fixes and Security updates.
-Tier 1 Data Center. 100% Network Uptime Guarantee
-Guaranteed Reliability backed by industry-leading Service Level Agreements

Log on now at **http://www.webappcabaret.com/jdj.jsp** or call today at **1.866.256.7973**

NGASI POWERED

WebAppCabaret℠

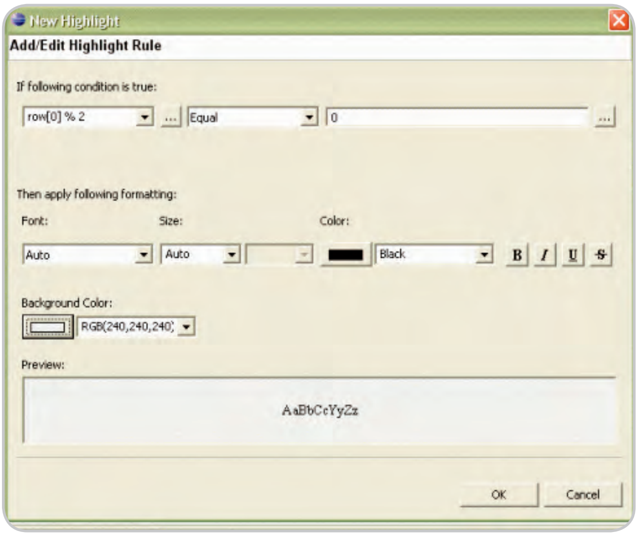
JAVA J2EE Hosting

data item to a calculated expression. BIRT uses standard JavaScript as its expression language, but in this case, it's a simple String concatenation that's the same in Java and JavaScript: "Region: + row["REGION_NAME"].

In the group footer, we want to have the Regional Total. To insert the calculated information, go to the Palette and drag a Data item into the Group Footer Table Cell. A Data item is the same as a text item, but it brings up the Expression Builder window by default. We want the sum of the TOTAL column so select "BIRT JavaScript Objects...Total (Aggregate) Functions... sum." This puts the "Total.sum()" method into the builder. We want to pass the AMOUNT field as an argument so go select it from "Available Data Sets." We want to label our total so the final expression looks like: "Regional Total: " + Total.sum(row["AMOUNT"]). Copy this onto the clipboard and put a new Data item into the lowest Footer row: "Grand Total: " + Total.sum(row["AMOUNT"]) (see Figure 4).

Reporting Terminology	
Rolling	Where does the text *roll* to when this column ends? The top of the next column? The next page? Most Business Intelligence tools handle page rolling well enough, but have limited (or no) support for magazine-style column rolling.
Widows & Orphans	A related issue to rolling...when you roll unexpectedly (that is, you designed a table to have 20 rows and end at the bottom of the page, and some sneaky data analyst puts a 21st row in the database), you end up with a widow (the original table) and orphans (the rows being *rolled* onto the next page). This is one of the toughest issues in producing robust reports for variable length data. The report designer must be careful at all times to ensure that pages designed for fixed-length data guarantee that the data is really fixed length, or designed for rolling, including table header and footer repetition.
Header & Footer Hierarchy	When doing a complex table of hierarchical data (as many Business Intelligence projects do), you'll have headers and footers that need to be repeated not just for the page, but for the table, the table's sections, and even the table's subsections. A good Business Intelligence tool will have a way of differentiating between initial headers and "rolled" headers so that you can add some indicator [e.g., (continued)] showing that the data is a continuation of the previous page.

Table 1 Reporting terminology



row[0] % 2 | Equal | 0
Font: Auto; Size: Auto; Color: Black
Background Color: RGB(240,240,240)

Figure 5 Highlight Rule Dialog Box

Formatting Text

As a final touch before we preview our report, let's set the fonts settings for our table. Click in the empty lower right-hand cell and drag a selection box into all the table cells. Choose the same font that was used in the header, Verdana, this time using the Font properties tag in the Property Editor at the bottom of the screen. Highlight the Group Footer and Header rows and make them bold and give them some additional top and bottom spacing to separate the report sections. Highlight the top Header row and make it Bold, and give it a double lower border in the Borders properties tab (Style: [double line], click bottom border button).

Highlights

A very common request in Business Reports is to alternate the background colors for each line in the report. To accomplish "dynamic" behavior like this we use Highlights, or value dependent properties. Select the detail row in the table, click the Property Editor then click Highlights in the bottom tabs. Click "Add" to add a highlight condition. Under "If the following condition is true:" put "row[0] % 2" (row[0] is the row count variable in BIRT), leave "Equal" in the middle box, and put "0" for the value. Change the "Background Color" from Auto to a suitable alternate color. This will cause all even rows to be highlighted with the indicated color value (see Figure 5).

Previewing Your Work

There are two ways to preview your work. BIRT provides a simple "Working Preview" in the "Preview" tab at the base of the report. Click it now to see your report. If you want to see a report with full pagination, select File...Preview as PDF from the top Eclipse menu. This will bring up an Eclipse PDF viewer window showing the fully rendered report with all headers/footers and pagination.

Achieving Open Source Business Intelligence

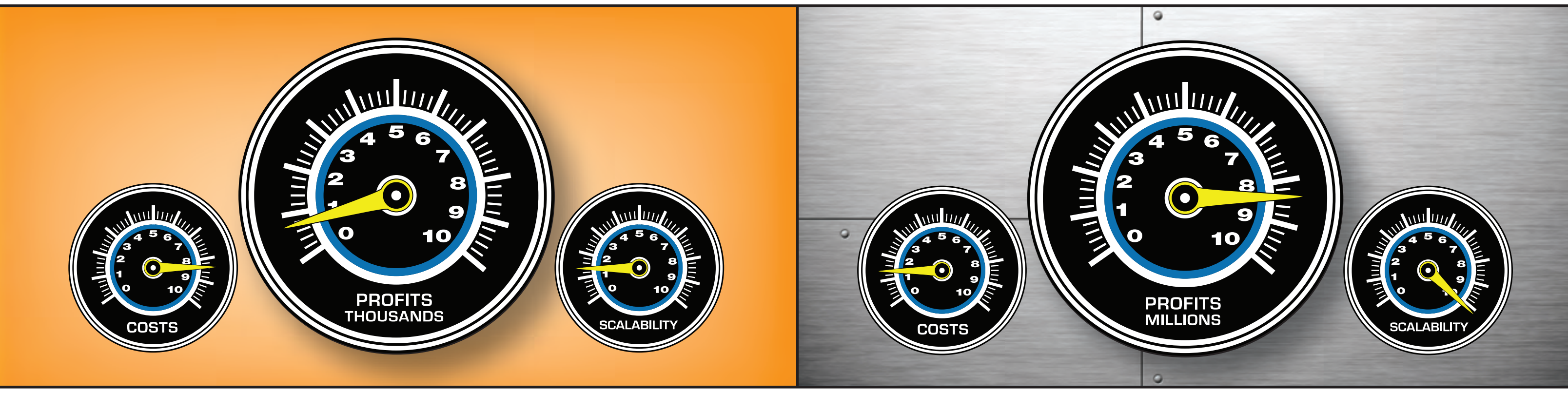
At this point, you should have a good initial understanding of the BIRT suite and its use. There are more advanced tutorials (creating sub-reports and scripting) available on the BIRT site at http://www.eclipse.org/birt and some excellent documentation in the "BIRT Developer Guide" in the Eclipse help at http://help.eclipse.org. Look there for next steps with your current report, like "Deploying a BIRT report to an application server."

BIRT isn't quite at the same level as a commercial solution like Business Objects, Microsoft Reporting Services, or Actuate, but it's encouraging how much one can do with this initial version. With sufficient grassroots support from loyal users, BIRT should become one of the most important and visible elements of the Eclipse suite, and rightly so.

Resources

- BIRT: www.eclipse.org/birt
- Pentaho: www.pentaho.org
- JasperSoft: www.jaspersoft.com

A slapdash approach won't get you there...



ExtenXLS will.

"ExtenXLS makes it possible to re-use spreadsheets representing a tremendous investment in time and business logic and to 'connect' those spreadsheets to various SQL data sources to provide access to usable data in a way never before possible."

- Project Leader, AT&T

"We use ExtenXLS for 'pushing' accounting data into an Excel template. A key customer wanted web access to Excel format reports. We tried .NET but had problems with preserving charts & formatting. ExtenXLS provides a flexible, cost effective solution that gives us the ability to stay ahead of user requirements."

- Project Leader, John J. McMullen Company

It's What's behind the Dashboard that Counts!

Slapping a dashboard onto a static spreadsheet file may **look** good, but it's like installing a flashy dashboard on an outdated clunker. Install that same front-end on ExtenXLS — **a scalable, server-based spreadsheet engine** — and watch your flashy dashboard come alive.

Now with Round Trip Reporting™

Reuse your existing spreadsheets as a data-entry tool. Modified data is extracted from your spreadsheets and turned into Java Data Objects for reuse in all your programs.

Reach New Heights without the Learning Curve

With the familiar look and feel of Excel™, ExtenXLS eliminates the learning curve imposed on your end-users by other reporting tools. ExtenXLS unlocks the business logic stored in static spreadsheets throughout your organization, saving time and money.

Escape the Gravitational Pull of Obsolete Reporting!

Output to customized HTML for maximum compatibility or to XML for further processing. Keep your users happy with native Excel™ output which preserves the VB macros, images, charts, and other features that transform live data into actionable reports. You can even embed a familiar spreadsheet component in your Swing applications.

**Don't rely on a slapdash approach for your mission-critical reporting needs.
Put a rocket under the hood and achieve escape velocity.**

Visit the mother ship at: www.extentech.com/jdj and take your free evaluation copy into orbit for a test flight.

EXTENXLS4™ JAVA | XLS REPORTING TOOLKIT

extentech™
Call Us: 415-759-5292

JavaServer Faces and AJAX for Google Fans

Create your own custom components and build RIAs

by Jonas Jacobi and John Fallows

This is our last article in a series of four that have been introducing the concepts of creating AJAX-enabled JavaServer Faces (JSF) components. In this article we are going to summarize and encapsulate the concepts that were introduced in the three previous *JDJ* articles starting with the "Rich Internet Components with JavaServer Faces" (Vol. 10, issue 11), and design a Google-like JDJ InputSuggest component.

We will show you how to use Mabon to create a simple and powerful input component with built-in suggest functionality similar to what Google Suggest provides. To make it easy for application developers to use our JDJ InputSuggest component, we are going to use the Weblets open source project to bundle external resources, such as icons and JavaScript libraries, into a Java archive (JAR) that represents our JSF component bundle.

Jonas Jacobi is a principal product manager and evangelist for Oracle's Java/J2EE tool offering, JDeveloper, and over the past three years has been responsible for JavaServer Faces, Oracle ADF Faces, and Oracle ADF Faces Rich Client development features within Oracle JDeveloper. Jonas has been in the software business for 15 years. Prior to joining Oracle, he worked at several software companies in Europe, covering many roles including support, consulting, development, and project team leadership.

jonas.jacobi@oracle.com

Creating an AJAX-Enabled JSF Input Suggest Component

The JSF AJAX input suggest solution consists of four classes as shown in Figure 1.

These classes are as follows:

- The HtmlInputSuggest is the renderer-specific subclass.
- The HtmlRenderer superclass provides some convenient methods for encoding resources.
- The HtmlInputSuggestRenderer is your new custom Renderer, which is in charge of the markup rendered to the client, including resources needed such as JavaScript libraries and style sheets.
- The HtmlInputSuggestTag is the tag handler.

The part of our input suggest solution that will provide the richness is a JavaScript library – inputSuggest.js – that contains functions needed to

leverage Mabon to retrieve data from the application developer's backing bean. We'll focus on the following artifacts – the inputSuggest.js file and the HtmlInputSuggestRenderer – both impacted by Mabon and provide the input field with type-ahead and suggest list functionality.

The Input Suggest JavaScript Library

Since we use Mabon, there is no need to worry about fetching data from the backing bean. We can leave this to Mabon. What we do need to be concerned about, though, is how to handle the returned data on the XMLHttpRequest object, how to populate the actual suggest list, and how to handle user interactions. The inputSuggest.js library contains a number of functions

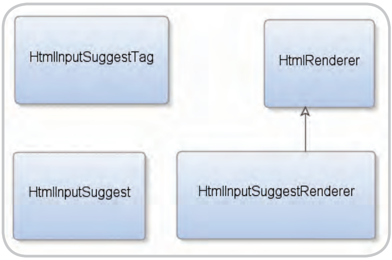


Figure 1 Class diagram showing classes needed for the input suggest component

that are used to handle keyboard navigation and mouse interactions, and for space limitations we'll be focusing on the functions that have the most impact on the JSF HtmlInputSuggest component.

The doKeyPress Function

The doKeyPress function, as shown in Listing 1, handles keypress events and checks whether the user pressed the TAB key or not. The TAB key would, under normal circumstances, navigate out of the input field and raising the blur event. For an input suggest solution, a TAB key can also be used to select an active row in the list of suggestions and as such we need to trap the TAB key and select a row in the suggest list and add the value to the input field, or, if no list is available, navigate out of the input field. If navigation occurs, the doBlur() function will be invoked and close the list of suggestions.

The doKeyUp Function

This function is invoked on any keyup event, and, depending on which key was activated, it will perform certain actions. Even though the TAB key has been managed by the doKeyPress function, we still have to make sure to catch it and terminate the process.

Another aspect of key strokes is the UP and DOWN arrow keys. Normally these keys will cause the cursor to navigate left and right in a regular input field. With an input suggest component a user is expecting these keys to navigate the list of suggestions.

The doKeyUp function's most important part, for this article, is to catch all key strokes that end up with a new character entered in the input suggest field, and is not a Backspace (see Listing 2). In this case the doKeyUp function will invoke the blur() function, which evaluates the value entered by the user and if changed raise a change event. We also have to reset the focus to the input suggest field, input.focus(), so the user can continue to enter more text.

If a user enters a new value, the doChange() function is invoked (see Listing 3). The doChange() function will call the mabon.send() function, passing a Map containing information about the value entered by the user, the JSF managed bean to invoke, and the callback function for this solution – _callback().

The _callback function, as shown in Listing 4, is responsible for handling the returned result from the response object and creating a list with suggestions according to the value entered by the user. Since we asynchronously communicate with the server, there is a chance that the user has entered a new character before the response comes back. To ensure that we can handle this, we have added a timer that delays the type-ahead feature until a certain time has passed.

If the value has changed since the request was made or the value is the same as the suggested value, we do nothing (see Listing 5). If the value is the same as the initial value entered, we add the first suggested value in the list to the input field and highlight the part that is appended to the value entered by the user.

The HtmlInputSuggestRenderer

It's time to have a look at how we can leverage the client-side functionality provided by the inputSuggest.js library and encapsulate it into one single JSF component. In this sample the main player is the JSF Renderer – HtmlInputSuggestRenderer. The Renderer is responsible for making sure that the correct markup is written to the client including any references to additional resources such as JavaScript libraries, CSS, icons, etc.... As we described in our previous *JDJ* article – "JSF and AJAX: Introducing a new open source project" (Vol. 11, issue 1) – you can use Weblets to package these resources into the same library as your JSF components.

Using Weblets

The open source Weblets project (http://weblets.dev.java.net) aims to solve the resource-packaging problem in a generic and extensible way so that all JSF component writers can leverage it, and it places no additional installation burden on the application developer.

A Weblet acts as a mediator that intercepts requests from the client and uses short URLs to serve resources from a JAR file. Unlike the servlet or filter approach, a Weblet can be registered and configured inside a JAR file, so the component library renderers, their resource files, and the Weblet configuration file (weblets-config.xml) can all be packaged together in the same JAR file. You don't need to separately deploy additional installables when the component libraries are upgraded to new versions. For the application developer, no configuration steps are needed.

It's important to note that all resources served up by Weblets are internal resources, used only by the Renderer. Any resources, such as images, that are provided by the application are supplied as component attribute values and loaded from the context root as external resources.

The HtmlInputSuggestRenderer Class

The two most important methods in this HtmlInputSuggestRenderer are the encodeBegin() and encodeEnd() methods. The encodeBegin() method, as shown in Listing 6, is responsible for writing out the needed resources for this component. The writeScriptResource() method and writeStyleResource() method are convenience methods provided by the HtmlRenderer and provide "write-only-once" semantics, preventing the same library from being written multiple times in case the application developer adds more than one input suggest component to the page.

Using Mabon

Mabon is an open source project hosted on the http://mabon.dev.java.net Web site. Mabon offers a convenient way to hook in a specially designed life cycle that is ideal for AJAX-enabled components that need to fetch data directly from a backing bean, without the overhead of a full JSF life cycle. It also provides a Mabon protocol – mabon:/ – that is used to reference the backing bean and a JavaScript convenience function that is used to send the target URL and any arguments needed and then asynchronously receive data from the managed bean.

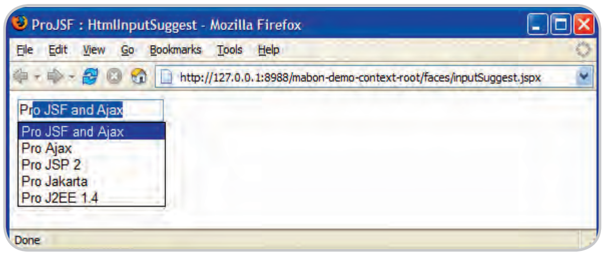

ProJSF : HtmlInputSuggest - Mozilla Firefox
File
Edit
View
Go
Bookmarks
Tools
Help
http://127.0.0.1:8988/mabon-demo-context-root/faces/inputSuggest.jspx
Pro JSF and Ajax
Pro JSF and Ajax
Pro Ajax
Pro JSP 2
Pro Jakarta
Pro J2EE 1.4
Done

Figure 2 The HtmlInputSuggest component rendered in a browser

Mabon and JSON

As you know, the XMLHttpRequest provides two response types – responseText and responseXML – that can be used to fetch data. The question to ask is, when should I use each? Answers to this question can differ depending on whom you ask, but we can recommend one rule. Ask yourself whether you control the syntax of the response.

The responseXML type returns a complete DOM object (which gives you ample ways of walking the DOM tree), allowing you to find the information needed and apply changes to the current document. This is useful when your component will impact surrounding elements and you don't control the response (for example, when you're communicating with a Web service).

For the input suggest component, you do control the response and you are looking at only fetching data for your component, not modifying the whole page's DOM structure. The responseText type returns plain text, which allows you to leverage JSON syntax for the response. For components leveraging AJAX, JSON is an extremely useful data-interchange format, since it can be easily parsed with the eval() function.

The eval() function takes one argument, a string of JavaScript code, and parses and executes this string in one go rather than trying to process each part separately. This is significantly faster than any other type of parsing, such as XML DOM parsing.

This is the reason why Mabon implements JSON – you control the response, and JSON syntax is easy and fast to parse.

The encodeEnd() Method

The real "work" is done in the encodeEnd() method, as shown in Listing 7. In the encodeEnd() method we get the Map of attributes from the HtmlInputSuggest component. One of the attributes on this component is the doSuggest attribute. From this attribute

John Fallows is a consulting member of technical staff for server technologies at Oracle Corporation, and has been working in distributed systems for over a decade. During the past five years, he has focused on designing, developing, and evolving Oracle ADF Faces, and is now lead developer for Oracle ADF Faces Rich Client.

john.r.fallows@gmail.com

we can get hold of the MethodBinding, if any, and from the MethodBinding object we can get hold of the actual MethodBinding expression defined by the application developer, for example, #{backingBean.doSuggest}. We then trim the #{} from the expression and concatenate the remainder of the string with the mabon:/ protocol-like syntax. The MabonViewHandler will recognize the string and return a resource URL that will be written to the client (for example, /context-root/mabon-servlet-mapping/backingBean.doSuggest).

Using the Input Suggest Component

Creating an AJAX solution is not a simple task, although there are several AJAX toolkits available, such as the Dojo Toolkit (http://www.dojotoolkit.org), that make it a lot easier. What JSF can offer is an even simpler programming model and one that is known by millions of developers: JSP and Java. To finish this article off in a fashion that suits a proper Ajax solution, let's look at how you can use this input suggest component in a JSF application, as shown in Listing 8.

This page contains one HtmlInputSuggest component, <jdj:inputSuggest>, that has the value attribute set to a value binding expression. This expression is pointing to a value property on the backing bean. The doSuggest attribute contains a method binding expression pointing to a doSuggest() method on the same backing bean. Let's have a look at the backing bean, as shown in Listing 9.

The value property is just a plain old JavaBean property. The doSuggest() method, as shown in Listing 10, is a bit more interesting. This method takes the initial value entered by the user and passed to it via Mabon from the doChange() function (see Listing 3). The doSuggest() method then returns an Array filtered based on the users initial value to the client. It is important that the returned value conforms to the supported JSON syntax.

The end result of this HtmlInputSuggest component looks like Figure 2.

Conclusion

From this article, we hope you have gained an understanding of how to Ajax-enable data fetch for your JSF components using Mabon, and how you can package external resources needed for your JSF component into the same archive as your Java classes leveraging the Weblets project.

Finally, now that you know how to create reusable rich Internet components with JSF and AJAX, we hope you will apply the techniques you have learned in this article series to create your own custom components and build Rich Internet Applications (RIAs) with JSF.

THIS ARTICLE IS BASED ON, AND CONTAINS EXCERPTS FROM, THE BOOK *PRO JSF AND AJAX: BUILDING RICH INTERNET COMPONENTS* BY JONAS JACOBI AND JOHN FALLOWS, PUBLISHED BY APRESS.

Listing 1: the doKeyPress function

```
projsf.jdj.doKeyPress = function(
  event)
{
  var input = (event.srcElement || event.target);
  var inputId = input.id;
  var div = document.getElementById(inputId + "$suggest");
  var divStyle = (div.currentStyle || div.style);
  if (event.keyCode == 9 && divStyle.display == "block")
  {
    div.style.display = "none";
    var activeRow = projsf.jdj._findActiveRow(div);
    input.value = activeRow.innerHTML;
    return false; //Cancel Tab out
  }

  return true; //Proceed with Tab out, which will invoke the doBlur()
}
```

Listing 2: The doKeyUp function

```
projsf.jdj.doKeyUp = function(
  event)
{
  var input = (event.srcElement || event.target);
  var inputId = input.id;

  var div = document.getElementById(inputId + "$suggest");
  if (event.keyCode == 9)//Tab key
  {
    return false;
  }
  else if ((div.style.display == "block" || div.childNodes.length >
0) &&
           (event.keyCode == 40 || event.keyCode == 38))
  {
    if (div.style.display == "none")
    {
      div.style.display = "block";
    }
    else
    {
      var activeRow = projsf.jdj._findActiveRow(div);

      switch (event.keyCode)
      {
        case 40: // Arrow Down
          if (activeRow.nextSibling)
          {
            activeRow.className = "HtmlInputSuggestRow";
            activeRow = activeRow.nextSibling;
            activeRow.className = "HtmlInputSuggestActiveRow";
          }
          break;

        case 38: // Arrow Up
          if (activeRow.previousSibling)
          {
            activeRow.className = "HtmlInputSuggestRow";
            activeRow = activeRow.previousSibling;
            activeRow.className = "HtmlInputSuggestActiveRow";
          }
          break;
      }

      input.value = activeRow.innerHTML;
    }

    return false;
  }

  if (event.keyCode != 8)//Not a Backspace
  {
    input.blur();
    input.focus();
  }

  if (input.value.length <= 2)
    div.style.display = "none";
}
```

Don't Miss the 2006 JavaOne℠ Conference

More than 300 technical sessions and Birds-of-a-Feather sessions will be offered at the 11th Annual JavaOne℠ Conference. See and hear from the Industry leaders and technology experts over four content-rich days. Included are:*

PLATINUM COSPONSORS

IBM

ORACLE

GOLD COSPONSORS

JUSTSYSTEM

SAP

SILVER COSPONSORS

QUEST SOFTWARE

TERRACOTTA

Introduction to AJAX

Ben Galbraith, *Consultant* | Dion Almaer, *Adigio, Inc*

This session provides an introduction to AJAX and an orientation to the state of the AJAXian universe. It demonstrates the basic AJAXian techniques through live coding and demonstrates and deconstructs more-advanced examples of AJAX.

Effective Java™ Reloaded

Joshua Bloch, *Google, Inc.*

It has been five years since *Effective Java*™ was released. The Java platform has evolved, and we've learned more about how to use it to best effect. Therefore, a second edition of *Effective Java*™ is being released to coincide with the 2006 JavaOne conference. This presentation covers new material that has been added to the second edition, material that should be useful to every working Java technology programmer.

Spring Framework Update

Rod Johnson, *Interface21*

Rod Johnson, the father of Spring, brings attendees up to date on some of the many significant new features in the Spring 1.3 and 1.4 releases. He discusses what's new and cool in the Spring world and examines the implications of these new features for best practice in developing applications with the Spring Framework. Johnson shows code examples throughout the presentation, leaving attendees ready to try these features out for themselves.

To see more information on the conference offerings visit java.sun.com/javaone/sf

*Content subject to change.

Register by April 14, 2006 and SAVE $100.

java.sun.com/javaone/sf

Sun microsystems

JavaOne℠ Conference | May 16-19, 2006

JavaOne℠ Pavilion: May 16–18, 2006, Moscone Center, San Francisco, CA

JavaOne

Sun's 2006 Worldwide Java Developer Conference℠

Listing 3: The doChange function

```
projsf.jdj.doChange = function(
  event,
  doSuggestURL)
{
  var input = (event.srcElement || event.target);
  var inputId = input.id;

  var context = { _inputId: inputId };
  net.java.dev.mabon.send({ url: doSuggestURL,
                            args: [input.value],
                            callback: function(result) {
                              projsf.jdj._callback.call(context,

 result);} });
  return true;
}
```

Listing 4: The _callback function

```
projsf.jdj._callback = function(
  results)
{
  var inputId = this._inputId;
  var input = document.getElementById(inputId);

  var div = document.getElementById(inputId + "$suggest");

  if (results.length <= 1)
  {
    div.style.display = "none";
    return;
  }

  // get the input from the context
  var input = document.getElementById(inputId);
  div.style.width = input.offsetWidth;
  while (div.firstChild)
  {
    div.removeChild(div.firstChild);
  }

  for (var i=0; i < results.length; i++)
  {
    var row = document.createElement("div");
    var span = document.createElement("span");
    var text = document.createTextNode(results[i]);
    row.className = "HtmlInputSuggestRow";
    row.appendChild(text);
    row.onmouseover = new Function("event",
                                  "projsf.jdj._doMouseOver(event
||
                                                  window.
event)");
    row.onclick = new Function("event",
                              "projsf.jdj._doMouseClick(event ||
                                                  window.
event)");
    div.appendChild(row);
  }

  div.firstChild.className = "HtmlInputSuggestActiveRow";

  div.style.display = "block";

  window.setTimeout("projsf.jdj._selectText('" + inputId + "', " +
                                        "'" + input.value +
"', " +
                                        "'" + results[0] +
"')",
                   200);
}
```

Listing 5: The _selectText Function

```
projsf.jdj._selectText = function(
  inputId,
  initialValue,
  suggestion)
{
  var input = document.getElementById(inputId);
  if (input.value != initialValue)
    return;

  if (input.value == suggestion)
    return;

  if (input.createTextRange)//IE Specific
  {
    var selectionStart = input.value.length;
    input.value = suggestion;
    var range = input.createTextRange();
    range.moveStart("character", selectionStart);
    range.moveEnd("character", input.value.length);
    range.select();
  }
  else //DOM Compliant
  {
    var selectionStart = input.value.length;
    input.value = suggestion;
    input.selectionStart = selectionStart;
    input.selectionEnd = input.value.length;
  }
}
```

Listing 6: The HtmlInputSuggestRenderer encodeBegin() Method

```
package com.apress.projsf.jdj.render.html;

Import ...

/**
 * HtmlInputSuggestRenderer renders a traditional HtmlInputText field
 * with auto-suggest behavior.
 */
public class HtmlInputSuggestRenderer extends HtmlRenderer
{
...
  public static String TITLE_ATTR = "title";
  public static String DO_SUGGEST_ATTR = "doSuggest";

  public void encodeBegin(
    FacesContext context,
    UIComponent  component) throws IOException
  {
    writeScriptResource(context,
                        "weblet://org.dojotoolkit.browserio/dojo.js");
    writeScriptResource(context,
                        "weblet://net.java.dev.mabon/mabon.js");
    writeScriptResource(context,
                        "weblet://com.apress.projsf.jdj/inputSuggest.
js");
    writeStyleResource(context,
                       "weblet://com.apress.projsf.jdj/inputSuggest.
css");
}
```

Listing 7: The HtmlInputSuggestRenderer encodeEnd() Method

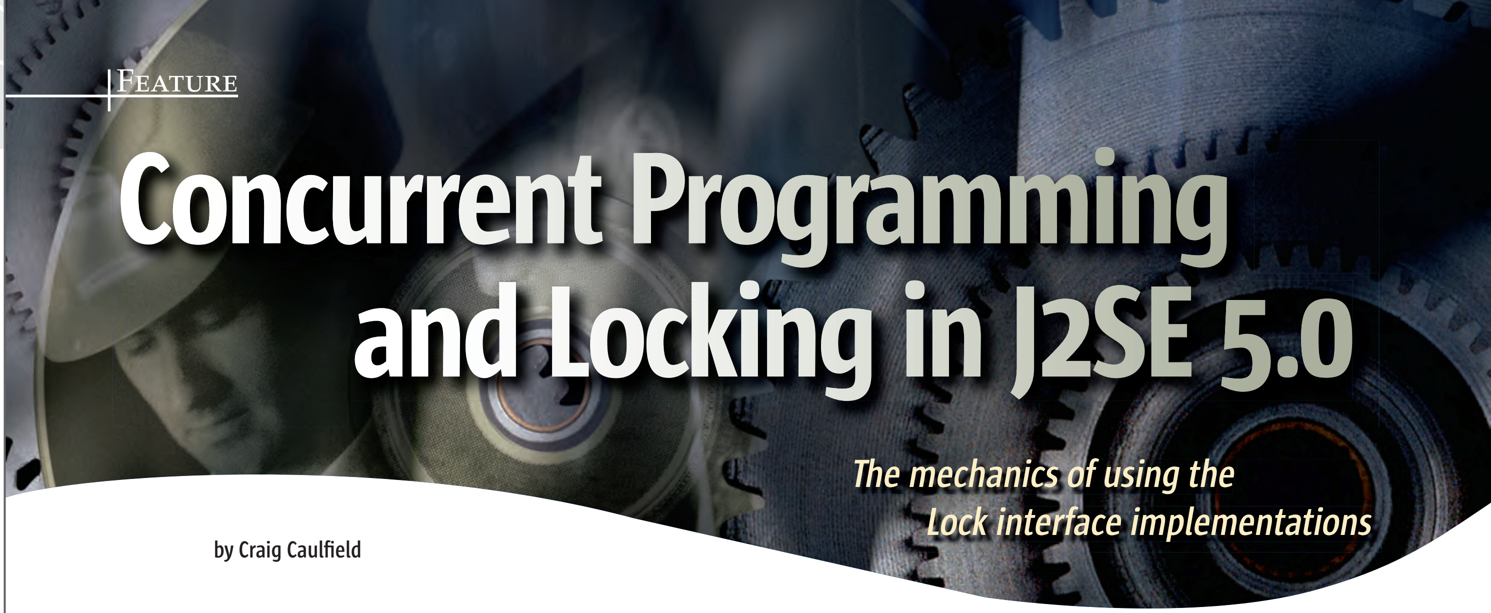
```
  public void encodeEnd(
    FacesContext context,
    UIComponent  component) throws IOException
  {
    String valueString = _getValueAsString(context, component);
    String clientId = component.getClientId(context);

    Map attrs = component.getAttributes();
    String title = (String)attrs.get(TITLE_ATTR);
    String onchange = (String)attrs.get(ONCHANGE_ATTR);
    MethodBinding doSuggest = (MethodBinding)attrs.get(DO_SUGGEST_
ATTR);

    ResponseWriter out = context.getResponseWriter();
    out.startElement("div", component);

    if (title != null)
      out.writeAttribute("title", title, TITLE_ATTR);

    // <input id="[clientId]" name="[clientId]"
    //        value="[converted-value]" onchange="[onchange]" />
    out.startElement("input", component);
    out.writeAttribute("id", clientId, null);
    out.writeAttribute("name", clientId, null);
    if (valueString != null)
      out.writeAttribute("value", valueString, null);
    if (doSuggest != null)
    {
      // disable browser auto-complete when using server-side suggest
      out.writeAttribute("autocomplete", "off", null);

      String expression = doSuggest.getExpressionString();
      // trim #{} from expression
      String bindingRef =
              expression.substring(2, expression.length() - 1);
      ViewHandler handler =
              context.getApplication().getViewHandler();
      String doSuggestURL =
              handler.getResourceURL(context, "mabon:/" + bindingRef);
      out.writeAttribute("onkeypress",
                         "return projsf.jdj.doKeyPress(event);",
null);
      out.writeAttribute("onkeyup",
                         "return projsf.jdj.doKeyUp(event);", null);
      out.writeAttribute("onchange",
                         "projsf.jdj.doChange(event, '" + doSug-
gestURL
                                               + "');", null);
      out.writeAttribute("onblur",
                         "return projsf.jdj.doBlur(event);", null);
    }
    out.endElement("input");
    out.startElement("br", null);
    out.endElement("br");
    out.startElement("div", null);
    out.writeAttribute("id", clientId + "$suggest", null);
    out.writeAttribute("class", "HtmlInputSuggest", null);
    out.endElement("div");

  }
```

Listing 8: A JSP page using the JSF input suggest component

```
<?xml version="1.0" encoding="UTF-8" ?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="1.2"
          xmlns:jdj="http://projsf.apress.com/jdj"
          xmlns:f="http://java.sun.com/jsf/core"
          xmlns:h="http://java.sun.com/jsf/html" >
  <jsp:directive.page contentType="text/html" />
  <f:view>
    <html>
      <head>
        <title>ProJSF : HtmlInputSuggest</title>
      </head>
      <body>
        <h:form id="form" >
          <jdj:inputSuggest id="inputSuggest"
                            title="Input Suggest Component"
                            value="#{backingBean.value}"
                            doSuggest="#{backingBean.doSuggest}" />
        </h:form>
      </body>
    </html>
  </f:view>
</jsp:root>
```

Listing 9: The backing bean value property

```
package com.apress.projsf.jdj.application;

import java.util.ArrayList;
import java.util.List;

/**
 * BackingBean is a backing bean for inputSuggest.jspx document.
 */
public class BackingBean
{
  public void setValue(
    Object value)
  {
    _value = value;
  }

  public Object getValue()
  {
    return _value;
  }

Listing 10: The backing bean doSuggest() method
  public String[] doSuggest(
    String initialValue)
  {
    List<String> suggestions = new ArrayList<String>();

    for (int i=0; i < _MASTER_LIST.length; i++)
    {
      if (_MASTER_LIST[i].startsWith(initialValue))
        suggestions.add(_MASTER_LIST[i]);
    }

    return suggestions.toArray(new String[0]);
  }

  private Object _value;

  static private final String[] _MASTER_LIST = new String[]
                                              {
                                                "Pro JSF and
Ajax",
                                                "Pro Ajax",
                                                "Pro JSP 2",
                                                "Pro Jakarta",
                                                "Pro J2EE 1.4"
                                              };
}
```



Concurrent Programming and Locking in J2SE 5.0

The mechanics of using the Lock interface implementations

by Craig Caulfield

In concurrent programming, exclusion refers to any technique that dynamically locks certain blocks of code so multiple threads can't corrupt their shared resources in ways that can cause integrity problems. In Java, exclusion has meant using the synchronized keyword against a method or block of code to control access to an object's lock.

Even though synchronization is a simple and concise way of controlling access to critical code, there are some limitations:

- While a thread is trying to acquire a lock, it can't be interrupted or timed-out.
- Each lock can only test against a single implicit condition using the wait() and notify() methods, which doesn't give developers much flexibility when trying to react to particular program states.

Starting with J2SE 5.0 there's another way of protecting a code block from concurrent access — the Lock interface and its implementations, the ReentrantLock and ReentrantReadWriteLock classes. These classes overcome the limitations of the synchronized keyword and give developers some useful new features to work with. For example:

- Threads can poll objects for existing locks held by other threads before trying to acquire a lock themselves. Locks can also specify timeouts, during which they can be interrupted.
- Now an object represents a lock. As an object, the lock can be stored, passed around, or discarded, meaning multiple objects can share the same lock, or one object can have multiple locks.
- A lock object can have multiple condition objects so it's possible to target individual threads or groups of threads.

Despite the power of these new features, they complement rather than replace the existing low-level concurrency primitives such as synchronized. So this article walks through the mechanics of using the Lock interface implementations and offers some guidelines for when they can best be used.

Exclusion Pre-Version 5.0

Prior to J2SE 5.0 the usual way of achieving exclusion was to apply the synchronized keyword against a method or block of code.

```
synchronized(someObject) {
   // work with object state
}
```

Any thread that wants to execute an object's synchronized code first has to acquire the object's lock. If the lock is already under the control of another thread, then the seeking thread goes into a blocked state and tries to acquire the lock at a later time. When the lock is eventually acquired, the code in the synchronized method or block is executed, and the lock is automatically released on exit, whether this occurs normally or through an exception.

As well as protecting certain sections of code from concurrent access, Java allows threads to actively cooperate towards a common goal by using the waiting and notification mechanism. For example, in the normal course of execution, a thread may have to wait for some condition to occur before it can continue, such as a variable reaching a certain value. Instead of wasting CPU cycles and retaining its exclusive object lock waiting for the right conditions, a thread can voluntarily step aside by calling wait(), a method of the base class object. The thread gives up its exclusive object lock and enters a waiting state. Only when another thread calls the notify() or notifyAll() methods, indicating conditions have changed, will the original thread try to re-acquire the lock and test the condition again.

```
public synchronized void someMethod() throws InterruptedException
{
   while(!someCondition)
      wait();

   //  work with object state
   notifyAll();
}
```

Using the synchronized keyword and the waiting and notification mechanism are simple ways of performing concurrent programming that are also platform-neutral and cause only a modest performance hit in the case of uncontended locks. (An uncontended lock means no other threads attempt to acquire an object's lock while another thread holds it. If threads have to compete to acquire a popular object's lock, more code is executed at the virtual machine level and performance degrades).

Craig Caulfield is a senior software engineer for a defence and commercial software house in Perth, Western Australia. He has a bachelor's degree in computer science, a master's degree in software engineering, and is certified in Java, XML, DB2, UML, MySQL, and WebSphere.

ccaulfi1@bigpond.net.au

Exclusion in J2SE 5.0

Before diving into the details of the new concurrent utilities in J2SE 5.0, they should be put into some context.
When using threads, developers need to consider some design issues that normally don't figure as prominently in sequential programming:

- ***Safety:*** means periodically locking certain sections of code so contending threads don't corrupt an object's state.
- ***Liveness:*** means ensuring that a program makes gradual progress towards some goal. In concurrent programs this progress can be affected by threads contending for the right to execute synchronized blocks of code, waiting for certain conditions to become true, or getting a slot in the execution schedule. While it's normal for concurrent programs to block occasionally because of these factors, long-term or permanent blockages need to be identified and prevented.
- ***Performance:*** means ensuring that each invoked method executes as soon as practicable.

Effective use of threads means balancing these design considerations, and the concurrent utilities in J2SE 5.0 can help. In contrast to the existing low-level concurrency primitives, these new tools are utilities in the java.util.concurrent package. Just as the Collections framework provides concrete implementations of commonly used data structures, the new concurrency utilities provide concrete implementations of commonly used concurrent tools.

Safety

Whereas the synchronized keyword provides implicit locking, the Lock interface and its implementations make locking explicit. The ReentrantLock and ReentrantReadWriteLock offer the same locking and memory semantics as the existing primitives but provide more features for developers. (The locks are called re-entrant because a thread can repeatedly acquire a lock that it already owns. The lock keeps track of these acquisitions and the tread must call unlock() for each one to release the lock fully.)

Listings 1 and 2 show an example of the ReentrantLock in action. This application models a fundamental double-entry bookkeeping requirement: a valid transaction must consist of a debit and a credit for the same amount. Unless this transaction happens atomically, the financial integrity of any system in which it's used is questionable. The run() method in Listing 2 performs a loop that transfers random amounts of money between a small number of accounts, which are modelled as array elements. After each transfer, the list of accounts is displayed along with the grand total of the money in the system; as long as the grand total remains the same, we can be sure that the transactions are happening atomically. Almost as important as this atomicity, transfers can only be made from accounts that have enough money.

In the transfer() method of Listing 2, the lock() and unlock() methods define the scope of the lock, which can be a single line, a few lines, or may equally span multiple methods and objects. Notice the location of the call to the unlock() method. The implicit locking provided by the synchronized keyword takes care of the acquisition and, most importantly, the release of object locks: when a synchronized method or block exits, either through normal execution or an exception, any locks are automatically released. There's no such protection when using the explicit locking of the Lock implementations. The usual idiom is to immediately follow the call to lock() with a try/finally block, with the lock being released in the finally clause. This guarantees that lock releases won't be forgotten.

ReentrantLock and ReentrantReadWriteLock constructors can take an optional fairness parameter. When this parameter is set to true and there's contention for an object's lock, the lock will be granted to the thread that's been waiting the longest. If the parameter is set to false or omitted the lock is granted to whichever thread tries to acquire it when it's next free, regardless of how many other threads may be waiting. Because of the overhead involved in this kind of positive discrimination, setting the fairness parameter to true will have a noticeable performance impact so that fair locks won't have the same throughput as unfair. For this reason, the fairness parameter should always be set to false or omitted unless there's a requirement that threads be served in a first-created, first-out order in which case there are dedicated data structures that can do this better.

In common with pre-J2SE 5.0 locking, the new locking utilities can conditionally suspend execution at certain times until an application's environment is right to carry on. Previously, lock objects were associated only with single conditions; now, there's no limit. This makes it possible to send wake-up notifications to specific groups of waiting threads rather than a broadcast reveille.

Because object's wait(), notify(), and notifyAll() methods are final and can't be overridden, the new lock utilities use await(), signal(), and signalAll() to do the same things. As a general rule, a call to await() should always be inside a while-loop, as shown in Listing 2, rather than an if-statement since there's no guarantee that the condition will be true when the thread is next notified. It's also a good practice to put the while-loop immediately after the call to lock() with no statements in between because any threads entering the locked code will execute these statements before hitting the while-loop and possibly cause side effects.

Liveness

As mentioned previously, liveness means that a program will do something…eventually. That is, its threads won't become so tangled that they are deadlocked and cause the program to hang. In general, programs may become deadlocked when they use a threading model that contains the following three conditions:

- Mutual exclusion: only one process or thread may use a resource at a time.
- Hold-and-wait: a process or thread may hold allocated resources while waiting to acquire others.
- No pre-emption: no resource can be forcibly removed from a process or thread that holds it.

The Java threading model contains all three conditions. So, it may happen that one thread acquires resource A and then tries to acquire resource B and another thread may already be holding resource B and is, in turn, trying to acquire resource A. This situation is called a circular wait: each thread holds at least one resource needed by another thread. Because each thread has exclusive control of the resources it holds and neither can be forced to give up this control, they are deadlocked.

The only way to avoid deadlock is by preventing the occurrence of one of the three conditions of policy above (which is often beyond the developer's control) or by preventing the occurrence of a circular wait. The usual way of preventing a circular wait is to design a lock hierarchy and make sure that all threads acquire their locks in the same sequence.

As always, developers are solely responsible for preventing deadlock in their applications, but some of the new utilities in J2SE 5.0 can help with this. For example, a thread will block indefinitely when it tries to acquire a lock that's owned by another thread. An alternative is the lockInterruptibly() method from the Lock interface. If the interrupt() method is called on a thread while it's waiting to acquire a lock, an InterruptedException is thrown and the attempt to acquire the lock is abandoned. From here, one possible course of action would be to throw a fresh exception on the basis of which the program can decide what to do next in the absence of a lock. For example:

```
Lock lock = new ReentrantLock();
try {
     lock.lockInterruptibly();
     try {
          // Lock acquired, so work with the object state as normal
     } finally {lock.unlock(); }
     } catch (InterruptedException e) {
       //  Can't acquire the lock. So end, or throw a new excep-
tion to let the caller know.
     }
```

Rather than trying to acquire a lock, possibly waiting, and possibly needing to be interrupted, lock acquisition can also be more tentative:

```
 Lock lock = new ReentrantLock();
if (lock.tryLock())
      // Lock acquired, so work with the object state as normal
      try  { . . .  }
      finally  { lock.unlock(); }
else
     // Can't acquire the lock, so take another path
```

The tryLock() method tries to acquire a lock and will return true if it was successful, otherwise it immediately returns false. If a thread can't acquire a lock, some alternative execution path can be taken. This alternative path may use a physical resource or data structure that isn't quite as fast or efficient as the first choice, but at least the program is progressing rather than spinning its wheels.

The tryLock() method can also be called with a timeout parameter. For example:

```
if (lock.tryLock(100, TimeUnit.MILLISECONDS))…
```

Using tryLock() with a timeout parameter has other advantages. If tryLock() is called with a timeout parameter, an InterruptedException will be thrown if the thread is interrupted during the timeout period. The exception handling in this case might include releasing any resources held and trying some alternative execution path, making the chances of a deadlock occurring less likely.

The same sort of timeouts can also be specified when waiting on conditions.

```
condition.await(100, TimeUnit.MILLISECONDS))
```

In common with tryLock(), the await() method returns if the timeout has elapsed or if the thread is interrupted, and in the normal cases where another thread calls the signalAll() or signal() methods.

Lock testing and timeouts give developers more options for keeping their applications out of deadlock, but this comes at a small cost. Preventing circular waits using a lock hierarchy simply means working out the order in which threads should acquire their locks. Meanwhile, to use lock testing and timeouts an application must be designed to allow alternative execution paths or ways of cleaning up its state so another attempt can be made later. Even so, the power of lock testing and timeouts lies in providing an application with more than one way of getting a task done rather than relying on the anti-deadlock smarts of a single-path algorithm.

Performance

It's most often true that acquiring a contended lock causes a bigger performance hit than acquiring an uncontended lock. One way to avoid this performance hit is to actively manage an application's concurrency by making sure as little work as possible is done under locking. The goal should be to obtain a lock, do whatever is necessary, and then release the lock quickly. If some time-consuming task has to be done, it should be moved out of the locked area wherever possible.

Another way to improve an application's concurrency is to use a ReentrantReadWriteLock. Rather than creating a blanket mutually exclusive lock, as in the case of the ReentrantLock, the ReentrantReadWriteLock defines reading and writing locks. A write lock is still mutually exclusive, but if none of these is active, then a read lock can be held by more than one reader thread at once.

The ReentrantReadWriteLock works best in situations where there will be more reader threads than writer threads as when working with a Lightweight Directory Access Protocol (LDAP)-like data structure. Whether the ReentrantReadWriteLock delivers any noticeable performance benefits really depends on this imbalance between reader and writer threads being maintained. Still, it's another tool that developers can call on.

Which Method to Use, Old or New?

Even though the locking utilities in J2SE 5.0 do the same as the existing concurrency primitives and much more, they

"Always start out with the synchronized keyword until it proves to be inadequate"

LinuxWorld® CONFERENCE & EXPO

Conferences: April 3 – 6, 2006
Expo: April 4 – 6, 2006

Introducing: OpenSolutions WORLD™

Boston Convention & Exposition Center

BOSTON'06

2 Conferences
10 Tracks
100+ Sessions
Over 200 Exhibitors

LinuxWorld Conference tracks

- *Mobile & Embedded Linux*
- *Network Management*
- *Kernel and Driver Development*
- *Linux on the Desktop*

OpenSolutions World Conference tracks

- *Business Case*
- *Security*
- *Open Source Applications*
- *Enterprise Application Development*
- *Dynamic IT: Cluster/Grid/Virtualization*
- *Emerging Trends*

Open source means business.

Find out what it can mean for your business at LinuxWorld Conference & Expo—in ten tracks and more than 100 in-depth sessions, tutorials, and workshops led by the open source movement's top minds.

Go beyond Linux.

Expand your vision at OpenSolutions World, a new conference at LinuxWorld covering the full spectrum of open source solutions and strategies for the enterprise.

Get the full picture.

Explore a comprehensive exhibition of open source products, technologies, and services from more than 200 key Linux and open source vendors—all under one roof.

Put it to work.

Come to LinuxWorld Conference & Expo April 3 – 6, 2006, and harness the power of the entire open source community for your business. The future of enterprise technology is wide open.

Register by 3/3 and save
Enter priority code D0303 and save up to $500
www.linuxworldexpo.com/boston

IDG INTERNATIONAL DATA GROUP   IDG WORLD EXPO

D0303

aren't automatic first-choice replacements. So, at some point the question will arise of which to use. This will naturally depend on the nature of the application being built, but there are some rules of thumb to help decide between the two.

First, avoid using both if at all possible. Unmistakeably, concurrent applications are more complex to design, code, and debug than sequential applications. Keeping an application as simple as necessary could mean avoiding all flavours of handcrafted concurrency. Still, there may be functional alternatives in the existing thread-safe collections, synchronization wrappers, or some of the new concurrent collections. This pushes the responsibility for thread management onto tried –and tested fundamental Java classes.

But if an application's design calls for a choice to be made, using the synchronized keyword produces code that is simpler and more concise. For example, the single synchronized keyword acquires an object's lock and ensures that it's released when the method or block exits, whether by normal means or by exception. Meanwhile, the new utilities rely on the idiom of explicitly acquiring a lock and then releasing it inside a finally block; it's the developer's responsibility to get this right.

Using synchronized can also have advantages when debugging applications. Because lock acquisition and release is handled by the JVM, the JVM is able to provide locking information when generating thread dumps. Meanwhile, the Lock implementations are vanilla Java classes and the same level of debugging detail isn't available.

Still, the features of the J2SE 5.0 locking utilities are alluring. Multiple condition variables, timed lock waits, interruptible lock waits, and broader lock scope offer developers great power. Doug Lea led the JSR 166 (Concurrency Utilities) specification team and the bulk of the new utilities come from his util.concurrent package, which has been around since the late 1990s. This means the utilities have been peer-reviewed and stress-tested over time to deliver the best possible performance and scalability.

So, when choosing between the old and new exclusion techniques, the best advice would be: start out with the synchronized keyword until it proves to be inadequate, then move onto the new utilities when the need for the extra features or performance is justified.

References

- Lea, D. *Concurrent Programming in Java*. Second Edition. Addison-Wesley. (1999).
- Oaks, S. & Wong, H. *Java Threads*. Third Edition. O'Reilly Media. (2004).
- JSR 166: Concurrency Utilities: http://www.jcp.org/en/jsr/detail?id=166
- Sun's Threads Tutorial: http://java.sun.com/docs/books/tutorial/essential/threads/index.html

Listing 1

```
/**
 * Description: a driver program. Creates a thread to run simple dou-
ble-entry bookkeeping
 *               processor.
 */

public class LockingRun {

    private static double[] accounts;

    public static void main(String[] args) {

        // Create some accounts between which funds will be moved.
        accounts = new double[] {109.0, 242.00, 455.89, 231.92,
1200.01};

        Thread processor = new TransactionProcessor(accounts);

        processor.start();

    }

}
```

Listing 2

```
import java.util.Random;
import java.util.concurrent.TimeUnit;
import java.util.concurrent.locks.*;

/**
 * Description: a simple double-entry book-keeping processor. When
amounts are transferred between accounts,
 *               a transaction lock ensures that the transfer (debit
one account, credit another) happens
 *               atomically.
 */

 class TransactionProcessor extends Thread {

     private Lock transactionLock = new ReentrantLock();
     private Condition enoughMoney = transactionLock.newCondition();
     private double [] accounts;
     private volatile boolean stopped = false;

     Random random = new Random();

   /**
    *   Constructs the transaction processor.
    */
    public TransactionProcessor(double[] accounts) {

       this.accounts = accounts;

    }

   /**
    *   Transfers money from one account to another.
    */
    private void transfer(double[] accounts, int fromAccount, int
toAccount, double amount) {

       transactionLock.lock();

       try {

          //  If there isn't enough money in the from-account, wait
and see if a deposit is eventually made.
          while (accounts[fromAccount] < amount)

             enoughMoney.await(1000, TimeUnit.MILLISECONDS);


          System.out.printf("Completing transfer of $%.2f from
account %d to account %s %n",
                                 amount, fromAccount, toAccount);
          accounts[fromAccount] -= amount;
          accounts[toAccount] += amount;

          printAccountPortfolio(accounts);

          //  Maybe there's enough money now. Wake up all threads
waiting on this condition.
          enoughMoney.signalAll();

       } catch (InterruptedException ie) {

           return;
```

```
        } finally {

            transactionLock.unlock();

        }
    }

   /**
    *   Print a list of all acounts and their balances.
    */
    private void printAccountPortfolio(double[] accounts) {

        transactionLock.lock();
        double total = 0.0;
        try {

            for (int index = 0; index < 5; index++) {

                System.out.printf(“\tAccount: %d $%.2f %n”, index,
accounts[index]);
                total = total + accounts[index];

            }

            //  If the transfers between accounts are happening atomi-
cally, this total won’t change.
            System.out.printf(“\tTotal balances: $%.2f %n”, total);

        } finally {

            transactionLock.unlock();
        }

    }
```

```
    public void run() {

        try {

            while (!stopped) {

                // Choose a pair of to- and from-accounts at random.
                int toAccount = random.nextInt(5);
                int fromAccount = random.nextInt(5);

                //  Ignore transfers to and from the same account.
                if (toAccount != fromAccount) {

                    //  Create a random amount under $25 to transfer.
                    double amount = 25 * random.nextDouble();

                    System.out.printf(“Attempting to transfer $%.2f
from account %d to account %d %n”,
                                             amount, fromAccount, toAc-
count);
                    transfer(accounts, fromAccount, toAccount, amount);

                }

                Thread.sleep(random.nextInt(100));

            }

        } catch (InterruptedException ie) {

            return;

        }
    }
}
```

DEVELOPERS,
LOOK NO FURTHER

ODTUG 2006
KALEIDOSCOPE
CLARIFYING THE DEVELOPER'S CHANGING UNIVERSE

Highlights:

- Web Architecture Symposium
  Saturday, June 17
- Hands-on Training
- BI Summit
- More Than 100 Technical Presentations
- Built-in Time to Meet and
  Talk with the Experts

June 17-21, 2006
Wardman Park Marriott Hotel
Washington, D.C.

To obtain the special conference rate of $175/night at the Wardman Park Marriott Hotel, call 1-800-Marriott by May 12, 2006. Attention government employees: Government rates are available, but conference room rates are lower.

TOPICS:

Application Development • Traditional Tools • Business Intelligence • Methodology
Development DBA • Professional Development • Non-Oracle Development Tools

www.ODTUG.com

Comparing Apache Tomcat Performance Across Platforms

by Frank C. Ziglar

Part 1: Performance and distinct error handling under memory load

We have measured performance information to distinguish the differences between the Windows and Linux platforms. Given comparable hardware we found the performance differences almost trivial.

When the server was pressed to capacity, our Windows installation was forced to turn away some traffic with minimal alteration in service performance, while our Linux installation elected to service nearly all the connections at the cost of introducing latency. However, before hitting capacity, our Linux server appeared on average to be capable of servicing connections at a slightly faster rate than our Windows server.

Sun's release of its J2EE specifications has been followed by enthusiastic developers crafting powerful applications that can be seen on the Web in businesses of every size. The result is one very hot topic continually focused on by many developer groups, but mature enough to be well known in all aspects of the industry. Some of the first questions that must be asked with any new project remain: what is the best starting point, and how should this application be deployed? Here, we'd like to provide some guidance as to what trade-offs can be expected before that decision is made.

Earlier editions of this report presented some comparative results between different servlet containers. The response and feedback have

been tremendous, and there's been no shortage of suggestions for comparing function X against Y and Z. However, with J2EE delivering platform-independence, many people (including myself) have gotten interested in determining just how significant a difference there is between the platforms. So we'll look at one of the most popular servlet containers, Apache Tomcat, operating on the two most common platforms in general use, Windows and Linux.

For simplicity's sake, this article will be broken into two parts. Here, we'll discuss a simple scenario that leads the servers quickly into a performance limitation. In Part 2, we'll examine the performance of the platforms after this limitation is lifted.

Testing Goals

Usage Scenario

We're interested in evaluating the servlet performance of a standalone server. It is expected to be applicable to small and medium-sized projects that deploy applications in departmental or small corporate environments. It's not intended to represent server performance in a clustered or distributed environment. Nor is it expected to reflect the performance of a server that's been highly optimized for maximum performance on a particular application.

Difficulties of Benchmarking

Our choice of benchmarks is merely a smoothed-out average of samples that we have observed in the wild. Each application stresses different aspects, or users have different expectations. Some applications will run better on some servers than others. Well aware of the danger of misleading benchmarks, we don't expect to reflect the performance of any particular application or class of applications that might be deployed in the real world. We merely hope to provide the data that helps the reader weigh the performance trade-offs of the platforms.

Remember too that numerical values have a margin of error. When we compare timing measurements that are nearly equal, the faster and slower responses can easily exchange places with the slightest fluctuation. Upsets can derive from practically any cause

"The tests saw the servers behave quite differently when forced beyond their capacity"

Frank C. Ziglar works at Web Performance, Inc.

fziglar@Webperformance.com

imaginable. One source, for instance, might be whether software configurations include or exclude background/idle processes. Minor revision changes in loaded applications/libraries can have the same effect. On the hardware side, the next faster or slower processor can affect the processor-to-IO performance ratio, which can upset subtleties in how the platform manages internal semantics.

Raw Numbers vs Relative Performance

Due to variations in applications and hardware configurations, we can't tell you how many users your particular application will be able to handle in your environment (which is why we sell a product for exactly that purpose). We've got numbers such as hits/sec and average page duration, but raw numbers are of little value by themselves. Only when statistics are compared against other servers do they provide valuable insight on which to make performance judgments.

It may seem obvious to some, but it's important to reinforce this point: If the data indicates that Server X can serve 200 hits/sec with an average duration of under two seconds, it doesn't correlate to the number of hits/sec that your application will be capable of. The data presented here for one server is *only* useful for comparison against another under the identical test conditions of our test lab. Also note that the hardware used for the server in these tests is not a state-of-the-art configuration.

Reproducible Results

No matter how the testing is done, someone will cry "Foul Play!" A primary goal of this effort is a test that's easily duplicated by anyone else who wants to try. No doubt the exact numbers won't be duplicable due to differences in hardware and measurement technique, but the relative differences between the servers should be reproducible.

Configuration

Philosophy

Ideally we'd like to give the community some insight into the relative differences between the platforms so one might be able to infer some information about his own application. So we've intentionally used any default values provided except where noted. We feel that while performance tuning will inevitably become a part of an app that's not meeting expectations, our testing strategy provides an effective baseline guide.

By using default settings, our test results can be interpreted more broadly than had we tuned our application server specifically for our test app. And then too there's practically an infinite number of permutations of options related to server and servlet container's functionality. Evaluating permutations of even a few of these settings can make our results too long to digest. Third, if the same functionality can be obtained by setting twice as many options (hence taking twice as much time to configure), the question of testing fairness could be argued.

Processor	One Intel Xeon 2.8GHz with HT Technology
Memory	512MB
Disk Drive	One 40GB SATA disk drive
Ethernet	One 1GbE Ethernet interface

Table 1 Installed hardware specifications

Server Software

The two operating systems used in this test were Microsoft Windows Server 2003 and CentOS 4.2 x86_64. Windows Update was used to install the most recent service packs as of November 16, 2005. Our copy of Linux was installed via CDs burned from the publicly available ISO images. The YUM Updater was then used to update all installed packages to the latest stable versions as of November 15, 2005.

Scenario	Duration	Number of pages	Page sizes	Number of resources (images, etc)	Resource sizes
Short - represents frequently used pages (e.g., home pages, corporate portals, and bookmarked pages that are checked occasionally).	10 seconds	1	60KB	30	500 B (x15) 2.5KB (x10) 5KB (x4) 10KB
Medium - represents a short operation on a Web site (e.g., a quick product search or shipping status check on an e-commerce site)	1 minute	5	60KB 40KB 20KB (x3)	50	500 B (x27) 2.5KB (x14) 5.0KB (x4) 10KB (x5)
Long - represents long and involved operations on a Web site (e.g., placing an order and entering payment and shipping information)	3 minutes	20	60KB 40KB 20KB (x3)	125	500 B (x72) 2.5KB (x29) 5.0KB (x4) 10KB (x20)

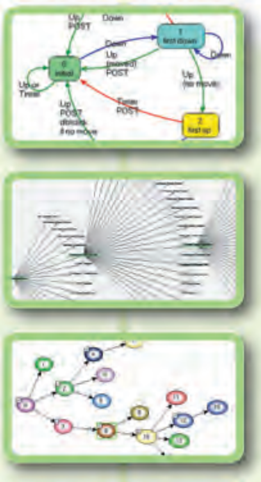
Table 2 Three user scenarios

Server Hardware

Two servers were used, each with identical hardware. Each was a Dell PowerEdge SC1420 (see Table 1).

Build interactive diagrams easier than you ever imagined

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



FREE Download With Full Support!

GoDiagram

Interactive diagram components

www.nwoods.com

Testing Laboratory

Each workstation used in the test was connected directly to a Dell PowerConnect 2324 switch. This switch offers two GbE ports to which the servers were connected as well as 24 100MbE ports.

Scenario	Scenario Distribution	User Distribution
short	40%	5%
medium	35%	30%
long	25%	65%

Table 3 Final user distribution

During each test the server not being tested acted as the test controller. The load controller used five additional workstations for additional load generation to minimize timing discrepancies. Each load-generating workstation was connected directly to the same switch through one of the 100MbE ports.

The Tests

User Scenarios

Despite the fact that any performance test is only an approximation of real-world use, it was important to choose user scenarios that are similar to the kind of use typical of production deployments. The server statistics for our own Web site provided the inspiration. After a detailed analysis of the paths and pages traversed on our Web site, a few common Web sites (Yahoo, Amazon, etc.) were also analyzed (see Table 2).

User Distribution

Based on the distribution observed in our server statistics, the distribution of the scenarios was chosen (below, middle). During the simulation, each virtual (simulated) user will execute a single scenario repeatedly until the end of the test. After compensating for the differences in the length of each scenario, the final user distribution could be calculated (see Table 3).

Bandwidth Distribution

Public Web sites and intranet applications see distinctively different distributions of user bandwidth. For public Web sites 56k to 4Mbit connections are typical. For intranet applications 10Mbit-100Mbit is common (this bandwidth is shared). So a maximum 10Mbit bandwidth per user was selected for simulation purposes. The bandwidth was limited for each virtual user by the testing tool.

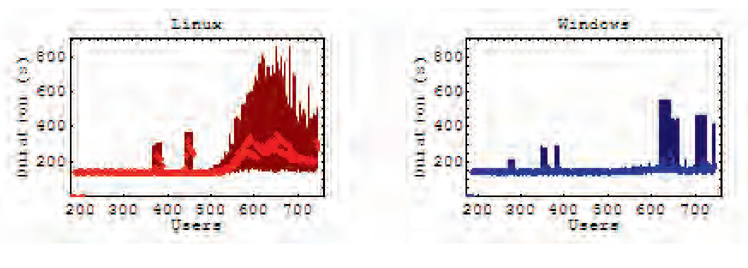


Figure 1 Long scenario

Note that with a bandwidth per user of 10Mbit on a 100Mbit network, no more than 10 users could be supported using their full bandwidth (assuming 100% network efficiency, which Ethernet can't achieve). However, all the scenarios contain significant "think time" (time between pages) that allows more than 1,000 users to use the 100Mbit bandwidth. Each test was stopped before any indications were given that the throughput capacity had been reach to ensure that our measurements were an accurate gauge of the server's performance.

Construction of Test Cases

The test case required a servlet that could return Web pages of a specified length referencing a specified number of external resources (images were used in this simulation). The servlet used provided the required functionality hard-coded in the servlet. Its source code is publicly available in the ContainerBenchmark.java file. Once the servlet and necessary resources were installed (via a WAR file), the test tool was used to record the scenarios interactively using the Web browser. In this case Opera 8.01 was used, but which browser is used to record the scenarios should not affect the test. Each scenario was recorded over the duration listed above. The testing tool was configured to simulate an approximately equal share of "think time" between each page, lingering slightly longer on the last page before restarting the test.

Testing Procedure

Test case recordings, virtual user simulation, and data gathering were all managed by the testing tool in this case Web Performance Trainer 2.8, Build 629.

Servlet Container Preparation

In our simple test environment, all operations were carried out while logged in with full administrative privileges. No further security restrictions were manually applied to the servlet's operating environment.

The first required component to be installed was the JRE. Sun's JRE 1.5.0 - update 5 was used and installed on each platform with the executable installer.

After the JVM was installed, Tomcat 5.5.12 was installed. The Windows download included another executable installer that automated the process. The Linux installation was done by simply unpacking the compressed archive then setting the necessary *CATALINA_HOME* and *JAVA_HOME* variables from the command line before using the shell scripts included with the code to start the container.

Tomcat 5.5.12 supports linkages to the APR library for acceleration, but for simplicity's sake, both servers were installed and configured with this option disabled. The testing was done by connecting directly to Tomcat and not through any additional software connector or proxy.

Once the servlet container was running successfully, the WAR file with our test application was copied to Tomcat's Web apps folder and auto-deployed.

Record the Test

Once the first servlet container was installed, it was possible to record one case of a user using his Web browser to step through each page of the test. Once the recorded, the test was configured as mentioned above.

Run the Test

Each test was preceded by a brief ramp of five to 11 users stepping through each page of the test. This warm-up ensured that the server had the opportunity to fully initialize any basic resources fundamental to the test. The actual test was started shortly thereafter. To focus on meaningful results, the tests quickly ramped to 275 virtual users in the first minute, and then increased by 11 virtual users a minute. Each test was allowed to run for 90 minutes then terminated. At the end of each test, idle time was allotted so the servlet container could restart and let all lingering resources be disposed.

Results

With our servers configured as outlined above, we could proceed to examining the test results. The servers appeared to hit a VM memory restriction quickly. The results after remedying this error should become available in part two of this article. An interesting aspect of this test is that we saw the servers behave quite differently when forced beyond their capacity.

We selected six categories of data to collect and present here:

1. ***Throughput:*** How much response bandwidth the server was able to generate.
2. ***CPU Utilization:*** How taxed the server became.
3. ***Errors per Second:*** How frequently the server was unable to respond.
4. ***Requests per Second:*** At what rate the Web browser made requests from the server.
5. ***Hits per Second:*** At what rate the server responded to requests from users.
6. ***Duration:*** How long a user could expect to wait before the page was complete.

Throughput

We started off with how much throughput the server could sustain under an increasing load. This measurement is taken from the total megabytes of HTTP traffic and doesn't consider lower-level implementation overhead.

The sudden drop in traffic is somewhat surprising. Examining the Tomcat server logs for Windows indicates some OutOfMemoryExceptions as the server is pressed under increasing load. The same logs for Linux, however, revealed no illuminating information.

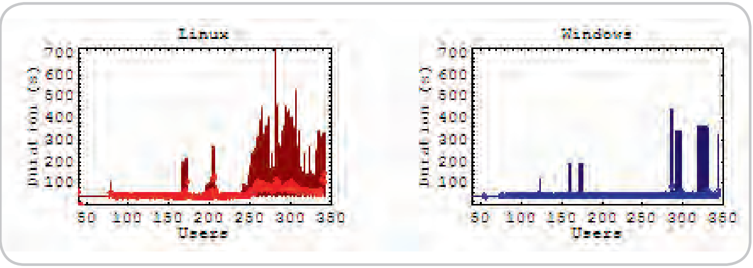


Figure 2 Medium scenario

CPU Utilization

CPU utilization gives some insight as to how well a server copes with increasing load and whether or not it is too computationally overwhelmed to process any further users. Each server in the test was equipped with one CPU with the HT Technology enabled. For simplicity's sake, we'll examine the average processor load on the two reported virtual CPUs.

Evidently both servers continued to consume the CPU linearly with load in roughly equal proportions. The load eased off slightly, but only slightly, during the server's slump in throughput. As the throughput began to increase again, the CPU increased too.

The relatively high level of CPU utilization during the throughput slump confirmed our suspicion that Tomcat and the JVM are still chugging along, evidently running memory management or optimization cycles.

Errors per second

During our tests the only errors that made themselves evident to the end user were the transmission errors that occurred when a user would attempt to open a connection to the server. To users this means their Web browser will display an error message similar to "Connection refused by server." Every page returned during this test was a complete page and not an error page from the server. Note that this plot is scaled logarithmically since the Windows server showed a greater tendency to generate errors.

Server Responsiveness

Now let's examine the number of completed requests per second as measured by the testing tool. This number is allowed to decline with the number of users as the server becomes unresponsive and users are forced to wait before they can make another request. Under normal circumstances, however, the number of requests should equal the number of responses received, and overall should be directly proportional to the server's total throughput.

Interestingly, our Windows server elected to maintain a consistent level of responsiveness to the user, so the Web

"During our tests the only errors that made themselves evident to the end user were the transmission errors that occurred when a user would attempt to open a connection to the server."

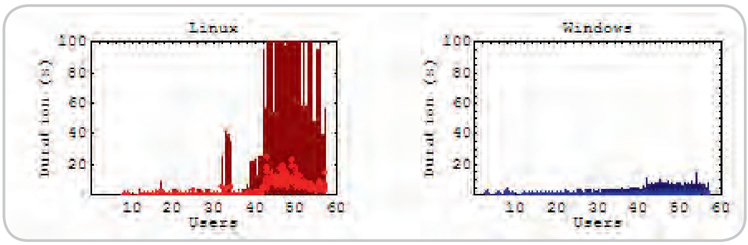


Figure 3 Short scenario

browser didn't have to wait a significant amount of time for a response. During the performance slump, we noted that the server simply refused further connections from the user giving them an immediate error. In contrast, the Linux server appeared to accept the connections and only responded when it was free to do so completely.

Hits per Second

Having seen how quickly requests were issued to the server, let's move to the response rate from the server. Hits-per-Second measures the rate of completed HTTP responses received from the server.

It seems here that even during the slump, despite error-handling techniques, the rate of successful HTTP messages processed remained roughly on a par between the servers, once again very slight favoring the Linux installation.

Duration

For our last performance aspect, we concerned ourselves with how long a user was going to be in completing the task. We measured the full time until the response was received from the server waiting until it arrived before moving onto the next page of the test.

For the duration times for a business case, the baseline for each graph is defined as the amount of time the user spends "interacting" with the page before moving to the next page. The simulated time spent by the user on the last page is therefore omitted.

The duration graphs here show the full range of measured durations during a given interval (dark background), with the averages on top (highlighted points).

In our long case (where the user must navigate through the longest list of pages) the anomalies had a nice chance to average out through one page or another. As expected, we saw the results of our throughput slump by increased wait times from the server. For Linux, as the load approached this turning point the duration increased, but never fully recovered. For our other cases, the general trend remained the same with the average duration declining with shorter test cases.

Static & Dynamic Content

From these cases we saw some relatively consistent increases in Linux duration with each business case, but the question remained were those increases attributed to the server being swamped by multiple requests or were they simply getting hung on the dynamically generated portions of the page? Note that these graphs are scaled logarithmically since the maximum durations were significantly different.

It's interesting to observe that there's a very slight but consistent increase in average durations for both servers, though more pronounced on the Linux server. Under load, the maximum duration for Windows rarely peaks above 10 seconds, whereas Linux steadily maintained maximum durations over 100 seconds.

Analysis

We've noted the behavioral and performance differences of each server up to, during, and after a bottleneck. Different bottlenecks will, of course, have key indicators as to their cause. But from the end user's point-of-view, many bottlenecks end up being dealt with in the same way. Some preliminary testing was done afterwards by tweaking the VM memory settings to remedy our performance slump and verify that it was, in fact, a memory limit. We did see indicators that the VM was forced to block for otherwise background optimizations, since the performance of both servers eventually recovered. In the next part of this article we'll find that tuning the JVM memory parameters resolved the performance slump that we saw in this round of testing confirming that the problem was a memory limitation. Despite the specific cause of our slump, it's not inconceivable that other performance pitfalls will end up being dealt with in the same way.

Before reaching capacity, the Linux server showed it could scale subtly better than the Windows server, with a notably higher completed response rate. This trend was produced again after the servers recovered from their performance slump.

When the servlet found itself hitting the memory limits of the app server, the platforms revealed different error-handling techniques. Linux maintained its lead over its Windows counterpart except when forced to deal with the memory shortage. Users were potentially forced to wait minutes or more for their page to load completely. Potential waits turned into repeated waits for users navigating through a long sequence of pages. Windows users had a different experience. Under the same memory shortage, the operating system turned traffic away, but delivered roughly the same number of successful hits as the Linux server.

"When we compare timing measurements that are nearly equal in value, the faster and slower responses could easily exchange places with just the slightest fluctuation"

REGISTER NOW!
www.iTVcon.com

CALL FOR PAPERS NOW OPEN!

Welcome to the Future of Video on the Web!

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

iTVCON.COM
INTERNET TV CONFERENCE & EXPO 2006

Coming in 2006 to New York City!

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

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

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

Welcome to the Future!

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

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

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

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

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

Jeremy Geelan

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

PRODUCED BY SYS-CON EVENTS

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

Joe Winchester
Desktop Java Editor

All for One and None for All

When someone in a corporate boardroom decides what their IT strategy is going to be, it isn't based on what language or software architecture they will use, but on how a system can provide value to their business. Very few organizations buy their hardware and OS first, and then tool up to write a bespoke solution that meets their business needs. In my first job I worked for a software house that built specialized insurance applications. Companies put out tenders for business that we responded to, and whether our products or a competitors' were chosen was based on the value proposition in the boardroom. The hardware, platform, and application server were dragged into the sale because they were required by the solution, but the app was always the endpoint that drove the purchase. As a software house we provided different configurations of the app that ran on different platforms and middleware. This was done for several reasons: to ensure we didn't have a dependency on a vendor lower down the stack and get maximum leverage by playing them off each other and also because some companies would standardize on a particular platform due to existing applications or an IT infrastructure that needed to be adhered to.

What has occurred over time is that companies have a myriad of applications sold to them by different vendors, perhaps one for HR, another for supply chain, some customer relationship management, accounting apps, and so forth. The stack of middleware, operating systems, and hardware that runs beneath the apps is often a mixed bag whose entropy increases as corporations merge or acquire one another. The IT check that gets written each year gets shared across everyone involved in the pie and the total cost of ownership grows as everyone takes their slice. The mismatch of heterogeneous applications and corporate data makes the overall business picture chaotic and sometimes anarchic.

One solution to the problem of heterogeneous and disparate systems is to migrate and consolidate on a single architecture. The problem with this pangaea is that it's very costly and difficult to achieve and adds very little core value to the business. Whenever I attend an event where customers and IT companies mix together, I'm always puzzled by the dichotomy: on the one hand the latest technology and software releases are being peddled by the vendors, while the IT departments are often several releases behind and are perfectly happy to remain on the existing infrastructure. What usually forces them to migrate is when a particular feature is only available in a newer release; in reality they'd be happy to just get the value the feature gave them in their existing version. A bad migration experience that cost someone a few weekends might be at the root of their reluctance to move,

coupled with the adage "don't fix what ain't broke." A bank I talked to recently admitted to having three 40-year old Dec PDP 11 machines that they need to move to a new data center, but are reluctant to touch because they haven't powered down since 1995. They're not alone in being a highly successful business that, given the choice, would rather allow the status quo of systems to remain. They're happy to add value by growing the communication and sharing each application's individual value. In a nutshell I think this is basically what service-oriented architecture is all about – by recognizing this need and providing an architecture where, instead of having a big central application onto which everyone migrates to, the endpoints publish their functionality and communicate directly with each other.

There was a time when Java tended to position itself as being the answer to the given problem, whatever the question was. Conference presentation foils peddled architectures with a J2EE application server in the middle, while other systems took part by using JCA and JMS. This Java-centric view, however, doesn't necessarily work well in all scenarios such as where a back-end database becomes bottlenecked by unoptimized fine-grained SQL calls thrown at it by the app server's EJBs. When I first heard about Web services, the explanation given was that they were "HTML pages for non carbon-based life forms." The simplicity of usage and large-grained transaction-based nature of the Web could now be enjoyed by programs talking to each other at a functional level. To enjoy this freedom, the existing enterprise applications need only publish their services at a sensible level of abstraction, and no longer have to learn Java protocol to play with the middleware. Java will play in this space, as there is still the need for application middleware to handle the routing, scaling, and management of the service calls. What it means for J2EE is that it will no longer be at the center of a Copernican middleware universe, as data will now flow in all directions and the real value comes from adding new applications, not at the top of the stack, but in the middle to glue, mediate, broker, and analyze. Java must reposition itself to play in this heterogeneous topology, not by asking to run everywhere and have others understand its model of programming, but instead to consume existing technologies and treat them as first class peers. Java can no longer view its being ported to each and every legacy platform as an end-game strategy of engulfing and extinguishing the existing apps. Instead, perhaps JVM architecture needs be extended to natively support other languages so disparate programs can run side-by-side in the same physical application space. Rather than the app server attempting to tackle difficult tasks such as batch or DRDA, perhaps Java needs to be able to speak more tongues, with the end game being to embrace the existing languages, APIs, and protocols already in place.

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

joewinchester@sys-con.com

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

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

SYS-CON MEDIA

ONLY

$69.99

ONE YEAR
12 ISSUES

Subscription Price Includes FREE JDJ Digital Edition!

www.JDJ.SYS-CON.com

or **1-888-303-5282**

OUT OF MEMORY! PAGE 36
Magazine in the World
JDJ
IN THIS ISSUE...
Leveraging Open Source Solutions to Improve Productivity While Using EJBs PAGE 8
Spring Web Flow PAGE 14
Intelligent Web Applications with AJAX PAGE 28
Navigating the Global Enterprise
ARE PORTALS THE MAGIC BULLET OF WEB APPLICATION DEVELOPMENT?
The many advantages to utilizing portal software
PLUS...
The Performance of EJB 3.0
Dealing with Open Source Software
A Blueprint for Developing Language Tools
THE WORLD'S LEADING i-TECHNOLOGY MAGAZINE JDJ.SYS-CON.COM

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

BY NOW THERE ISN'T A SOFTWARE DEVELOPER ON EARTH WHO ISN'T AWARE OF THE COLLECTION OF PROGRAMMING TECHNOLOGIES KNOWN AS AJAX!

REAL-WORLD AJAX SEMINAR

Hilton	**April 24, 2006** San Jose, CA
The Roosevelt Hotel	**June 5-6, 2006** New York, NY
AjaxWorld Conference & Expo Rich Internet Applications	**Oct 3-4, 2006** Santa Clara, CA

How, in concrete terms, can you take advantage in your own projects of this newly popular way of delivering online content to users without reloading an entire page?

How soon can you be monetizing AJAX?

This "Real-World AJAX" one-day seminar aims to answer these exact questions...

HURRY! LIMITED SEATING THESE SEMINARS WILL SELL-OUT CALL 201-802-3022 TO REGISTER!

Led by "The Father of AJAX" himself, the charismatic Jesse James Garrett of Adaptive Path, "Real-World AJAX" has one overriding organizing principle: its aim is to make sure that delegates fortunate enough to secure themselves a place – before all seats are sold out – will leave the seminar with a sufficient grasp of Asynchronous JavaScript and XML to enable them to build their first AJAX application on their own when they get back to their offices.

Jeremy Geelan
Conference Chair, Real-World AJAX
jeremy@sys-con.com

Register at...
www.AJAXSeminar.com

April 24, 2006 (San Jose, CA)	$1,195
See website or call for group discounts	
June 5-6, 2006* (New York, NY)	$1,295
See website or call for group discounts	
October 3-4 (Santa Clara, CA)	$1,495
See website or call for group discounts	
On-Deman Online Access (Any Event)	$695
Purchase both New York and San Jose seminars for on-demand access	

*Includes SOA Web Services Edge plus Enterprise Open Source Conference & Expo

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

SPONSORED BY Adobe telerik BACKBASE TIBCO Perficient Microsoft

AND XML ASYNCHRONOUS JA-
VASCRIPT AND XML ASYNCHRO-
NOUS JAVASCRIPT AND XML
ASYNCHRONOUS JAVASCRIPT
AND XML ASYNCHRONOUS JA-
VASCRIPT AND XML ASYNCHRO-
NOUS JAVASCRIPT AND XML
ASYNCHRONOUS JAVASCRIPT
AND XML ASYNCHRONOUS JA-
VASCRIPT AND XML ASYNCHRO-
NOUS JAVASCRIPT AND XML
ASYNCHRONOUS JAVASCRIPT
AND XML ASYNCHRONOUS JA-

What you'll walk away with...

A New understanding of AJAX and how to make it work for you, plus:

A) **Conference Proceedings Binder**
Full-conference proceedings in a self-contained commemorative binder.

B) ***Ajax in Action* Book**
A copy of the Best-Selling AJAX book by Dave Crane.

C) ***Real-World AJAX "Secrets of the Masters"* Book + DVD**
New AJAX book edited by Dion Hinchcliffe (Release Date: Summer 2006)

D) **Notebook**
Executive style keeps you looking sharp for note-taking wherever your business might take you!

E) **Computer Back-Pack**
A hip new way to conceal your laptop! The Urban Wonder Compu-Pack is an ingenious creation - a 600-denier polycanvas and nylon knapsack!

F) **T-Shirt**
100% preshrunk 6.1 oz. heavyweight cotton tagless shirt features shoulder-to-shoulder tape, and double-needle sleeves and bottom.

G) **Porcelain Mug**
Lightweight and durable, you can use it as a teacup, coffee mug or keep it as a pencil holder.

H) **Satin Silver Contemporary Pen**
Satin-silver plastic barrel with the look and feel of brass. Rubber comfort grip improves writing control and comfort.

I) **Real-World AJAX Seminar on DVD**
Watch complete seminar video and slide presentations (Release Date: Summer 2006)

Featuring... Real-World AJAX Rock Stars!

Jesse James Garrett (San Jose April 24)
Father of "AJAX" Who Coined the Term in 2005

Jesse James Garrett is the Director of User Experience Strategy and a founding partner of Adaptive Path, the world's premier user experience consulting company. He is author of The Elements of User Experience (New Riders), and is recognized as a pioneer in the field of information architecture. Jesse's clients include AT&T, Intel, Crayola, Hewlett-Packard, Motorola, and National Public Radio. Since starting in the Internet industry in 1995, Jesse has had a hands-on role in almost every aspect of Web development, from interface design and programming to content development and high-level strategy. Today, information architects around the world depend on the tools and concepts he has developed, including the widely acclaimed "Elements of User Experience" model. He is co-founder of the Information Architecture Institute, the only professional organization dedicated to information architecture. He is also a frequent speaker and writer whose work has appeared in numerous publications, including New Architect, Digital Web, and Boxes and Arrows.

Adam Bosworth (San Jose April 24)
Vice President of Engineering, Google
One of the Fathers of XML & the Creator of MS Access

Adam Bosworth is Vice President of Engineering, Google. He joined Google in 2005 from BEA Systems, where he was Chief Architect & Senior VP of Advanced Development. Prior to joining BEA, Bosworth co-founded Crossgain, a software development firm acquired by BEA. Known as one of the pioneers of XML, he previously held various senior management positions at Microsoft, including General Manager of the WebData group, a team focused on defining and driving XML strategy. While at Microsoft he was also responsible for designing and delivering the Microsoft Access PC Database product and assembling and driving the team that developed the HTML engine of Internet Explorer 4.0.

Dion Hinchcliffe (San Jose April 24)
Cofounder & CTO, Sphere of Influence Inc.
Editor-in-Chief, Web 2.0 Journal

Dion Hinchcliffe, newly appointed editor-in-chief of SYS-CON's pioneering Web 2.0 Journal, is cofounder and chief technology officer for the enterprise architecture firm Sphere of Influence Inc., in McLean, Virginia. A veteran of software development, Dion works with leading-edge technologies to accelerate project schedules and raise the bar for software quality. He is highly experienced with enterprise technologies and he designs, consults, and writes prolifically. Dion actively consults with enterprise IT clients in the federal government and Fortune 1000. He is a frequent speaker on AJAX, Web 2.0 and SOA and is currently the top-read SYS-CON.com blogger.

Christophe Coenraets (San Jose April 24)
Senior Technical Evangelist, Adobe
AJAX/Flex Integration Guru

Christophe Coenraets currently works as a Senior Technical Evangelist at Adobe. Before joining Adobe, Christophe was an evangelist at Macromedia, focusing on Rich Internet Applications and Enterprise integration. Prior to Macromedia, Christophe was the head of Java and J2EE Technical Evangelism at Sybase, where he started working on Java Enterprise projects in 1996. Before joining Sybase in the US, Christophe held different positions at Powersoft in Belgium, including Principal Consultant for PowerBuilder, and Manager of the Professional Services organization. Before joining Powersoft, Christophe worked as a developer and architect on several retail and BPM projects. Christophe has been a regular speaker at conferences worldwide for the last 10 years.

Paul Rademacher (San Jose April 24)
Google, Creator of HousingMaps.com

Paul Rademacher is the creator of HousingMaps.com, which combined Craigslist and Google Maps for the first web mashup. Paul holds a Ph.D. in Computer Science from UNC-Chapel Hill, and worked as an R&D Engineer at Dreamworks Animation on such movies as Shrek 2 and Madagascar. Since creating HousingMaps, Paul is now at Google.

Jouk Pleiter (San Jose April 24)
Co-Founder & CEO of Backbase

Jouk Pleiter is the CEO of Backbase, a leader in the field of Rich Internet Applications and AJAX development software. Backbase's clients include ING, ABN AMRO, TNT, KPN, Comsys and Heineken. Backbase operates globally with offices in San Mateo (North America) and Amsterdam (Europe). Since 1995, Jouk has been an entrepreneur: he founded three successful software companies. Prior to Backbase, Jouk was part of the founding team at the web content management company Tridion, where he led the product management operations, and was driving the company's efforts to become a leader in the European WCM software market. Jouk previously was part of the founding team at the Interactive Agency Twinspark where he grew the company to a leading market position in Europe and was instrumental in the sale of Twinspark to Agency.com. He has an MBA from the University of Groningen.

Kevin Hakman (San Jose April 24)
Director of Product Marketing for TIBCO
General Interface TIBCO Software

Kevin Hakman is the director of product marketing for TIBCO General Interface, the award winning AJAX and Rich Internet Application framework and toolkit. Kevin Hakman pioneered AJAX in the enterprise co-founding General Interface in 2001. Since that time General Interface (aka 'GI') has been powering Web applications that look, feel and perform like desktop applications, but run in the browser at Fortune 500 and U.S. Government organizations. General Interface was also the first to use its own toolkit to provide full visual tooling for AJAX when it released it's 2.0 Version in 2003. TIBCO acquired General Interface in 2004 to extend its vision for service oriented applications to the end user. Kevin is a contributor to the SOA Web Services Journal and the AJAX Developer's Journal.

Shanku Niyogi (San Jose April 24)
Product Unit Manager of the UI Framework and Services Team Microsoft Corporation

Shanku is Product Unit Manager of the UI Framework and Services (UiFX) team, which is responsible for delivering high-productivity UI framework technologies for the .NET platform, including ASPNET, Atlas, Windows Forms, and frameworks for smart clients. Prior to his current role, Shanku was Group Program Manager of the Web Platform and Tools team on the Whidbey release of ASPNET and Visual Web Developer. Shanku joined Microsoft in 1998 as a developer, having spent several years shipping products in the Windows ISV industry. Shanku holds a Bachelor of Mathematics degree in Computer Science from the University of Waterloo.

Coach Wei (New York June 5-6)
Chairman, Founder and CTO, Nexaweb
The Creator of First Commercial AJAX Applications

Coach Wei combines in-depth IT industry expertise with extensive education and research experience at MIT to drive technology innovation and business direction for Nexaweb. He founded Nexaweb in 2000 and served as CEO until summer 2003. Before founding Nexaweb, Coach architected and designed enterprise software for managing storage networks at EMC Corporation. As a graduate researcher at MIT, Coach developed software and hardware systems for non-destructive evaluation as well as signal/image processing algorithms. Coach was a finalist in the 1999 MIT $50K entrepreneurship competition and holds several U.S. patents. An accomplished writer and speaker, Coach has published numerous articles on topics including: AJAX, J2EE and .NET, RIA development, XML, signal/image processing, composite materials and ultrasonic imaging. He has spoken at top industry events, such as JavaOne and Web Services Edge. Coach holds an MS in information technology from MIT.

Ajit Jaokar (New York June 5-6)
CEO, futuretext
Author, "Mobile Web 2.0"

Ajit Jaokar, based in London (England), is the CEO of a publishing company, futuretext (www.futuretext.com). He is currently writing a book about Mobile Web 2.0 (Mobile Web 2.0: The Innovator's Guide to Developing and Marketing Next Generation Wireless / Mobile Applications). Ajit also chairs Oxford University's Next-Generation Mobile Applications Panel and, since January 2006, has been a member of the Web 2.0 Workgroup. In his "Real-World AJAX" conference session, Ajit will discuss the "AJAX Use in Mobile Applications" as part of the wider impact of Web 2.0, sometimes referred to as the "Global SOA."

Jonas Jakobi (New York June 5-6)
AJAX Evangelist and Co-Author, "Ajax and JSF: Friend or Foe?"
Jonas will autograph a copy of his book for all delegates!

Jonas Jacobi is a principal product manager and evangelist for Oracle's Java/J2EE tool offering, JDeveloper, and over the past three years has been responsible for JavaServer Faces, Oracle ADF Faces, and Oracle ADF Faces Rich Client development features within Oracle JDeveloper. Jonas has been in the software business for 15 years. Prior to joining Oracle, he worked at several software companies in Europe, covering many roles including support, consulting, development, and project team leadership. Jonas' new book "Ajax and JSF: Friend or Foe?" released by Apress on February 25, 2006.

John Fallows (New York June 5-6)
AJAX Evangelist and Co-Author, "Ajax and JSF: Friend or Foe?"
Jonas will autograph a copy of his book for all delegates!

John Fallows, former lead developer for Oracle ADF Faces Rich Client, has been working in distributed systems for over a decade. After five years spent focused on designing, developing the JavaServer Faces standard to provide AJAX functionality, playing a leading role in the Oracle ADF Faces team, he recently joined an AJAX start-up. Originally from Northern Ireland, John graduated from Cambridge University in the United Kingdom and has worked in the software industry for more than ten years. Prior to joining Oracle, he worked as a research scientist for British Telecommunications Plc.

Steve Benfield (New York June 5-6)
Well-known AJAX Evangelist and CTO of Agentis Software
Steve's first talk on "Aspect-Oriented Programming & AJAX"

Steve Benfield is CTO of Agentis Software and one of the pioneers of AJAX technology, a gifted writer and a technical visionary. A technology marketeer and strategist with 20 years of software entrepreneurism experience, a combination of qualities that made him the perfect choice of editor-in-chief for SYS-CON Media's inaugural publication 12 years ago. Steve's proven ability to determine marketing and technology strategies that align with market needs led to successful stints at SilverStream, where he started as technology evangelist and ended as CTO, and at ClearNova, an open source AJAX company, where he was CTO and AJAX evangelist.

Jeremy Geelan
Conference Chair
Group Publisher & Editorial Director, SYS-CON

Jeremy Geelan is group publisher and editorial director of SYS-CON Media, and is responsible for all print titles and online i-technology portals for the firm. He regularly hosts SYS-CON.TV, is executive producer of the "Power Panels with Jeremy Geelan" iTV series, and represents SYS-CON at conferences and trade shows, speaking to technology audiences both in North America and overseas. His i-Technology Blog is at jeremy.linuxworld.com and he is conference chair of the upcoming iTVcon - "Internet TV Conference & Expo 2006".

For more information...

Call 201-802-3022 or email events@sys-con.com

SYS-CON EVENTS **For more great events visit www.EVENTS.SYS-CON.com**

NOTE: SPEAKER LINE-UP SUBJECT TO CHANGE WITHOUT NOTICE
VISIT WWW.AJAXSEMINAR.COM FOR THE MOST COMPLETE UP-TO-DATE INFORMATION

Debugging JDBC with a Logging Driver

A new way to debug and solve DB problems

by Ryan Bloom

A couple of years ago I began developing in Java, and my first Java project required that I also learn SQL. Our project team was using mostly EJBs for database access, although for some performance-critical sections of the application we wrote the JDBC logic directly. A problem that we faced regularly was tracking the bind parameters to our PreparedStatements. Over the course of the project, all of the team members tried different techniques to determine what our JDBC statements were actually doing.

This article presents a solution that one of the developers created to help solve this problem. I took his original idea and extended it after I left that team to help my next company solve the same problem. The solution is a JDBC driver that uses log4j to print all SQL statements and any bound parameters to the application's log file.

Most JDBC developers have faced the problem of having to debug SQL statements in their applications. While this sounds like a relatively easy task, it's often error-prone, and results in code that is hard to maintain and that clutters otherwise clean application logic. An example of this type of logic is:

```
PreparedStatement ps = conn.prepareStatement("Select * from users
" + "where id = ?");
log.debug("Select * from users where id = ?");
ps.setInt(1, x);
log.debug("1: " + x);
```

This solution has some obvious problems. The first problem is that you have duplicated all of the bind parameters and the SQL statement in the code. If either the SQL statement or the parameter that you want to bind to the statement changes, you must remember to update the log statement as well. If you forget to update either log statement, your log will be useless when you try to debug your next problem. Also, because of the performance of using log.debug, most people put those statements in an if block, making the code that much harder to read.

The next problem with this model is that you don't have a centralized location to enable debug logging. When a QA developer reports a bug that you have never seen in your dev testing, the best place to start is to enable more detailed logging to determine what is going wrong. With the above code, you must either enable all debug logging, which is often data overload, or you need to debug the problem enough to determine which class is causing the problem.

The final problem with this system is most important when using a system like Hibernate. Hibernate is an object-relational mapping tool that also defines its own SQL-like language, HQL. The problem with this is that in your Java code, you don't have access to the actual SQL statement that is executed; you only have the HQL statement. In a previous job, we had a problem like this. The SQL statement translated to:

```
Select * from Sessions where id = ? and subId = ? and user = ?
and list = ?
```

The developer using this statement spent days poring over this query trying to determine why it wasn't returning any data when we expected a list with 10 elements in it. The final answer turned out to be that the final bind parameter was being passed in as NULL. Since the details of the SQL statement had been abstracted out by HQL, this problem wasn't clear. We discovered this by accident after reading the log file as a group. The rest of this article will focus on a better answer that would have shown us the problem immediately.

The Logging JDBC Driver

The idea behind this project has some concepts in common with aspect-oriented programming (AOP). The basic thought process is that there are some problems that are found throughout a project, and the solution to those problems shouldn't require programmers to remember to add special logic every time one of these problems comes up. Instead, the programming framework that you are using should create solutions to these problems for you. In this case, our problem is determining exactly what SQL queries are being run and with which parameters.

Working to our advantage with SQL queries, they all must go through a JDBC driver, so if we can find a way to log the correct information in the driver, the rest of the application won't have to change. That is the goal of the logdriver, available at http://www.rkbloom.net/logdriver under the Apache 2.0 License.

Ryan Bloom is a development manager at Peopleclick, an HR software company. He has a history of open source development with the Apache Software Foundation, including working on Apache 2.0.

rbb@rkbloom.net

The driver currently logs comprehensive debug information for Statements, PreparedStatements, and CallableStatements and some important actions on the Connection. The primary data being logged is the SQL query and any bind parameters. There are two formats that can be used. The first separates the SQL from the bind parameters:

```
executing PreparedStatement: 'insert into ECAL_USER_APPT (appt_id,
user_id, accepted, scheduler, id) values (?, ?, ?, ?, null)'
    with bind parameters: {1=25, 2=49, 3=1, 4=1}
```

The second format is enabled by setting the system property replace.bindParams to either 1or true. This format tries its best to insert the bind parameters into the SQL query:

```
    executing PreparedStatement: insert into ECAL_USER_APPT (appt_id,
user_id, accepted, scheduler, id) values (25, 49, 1, 1, null)
```

Both formats have their advantages, but most of the time it's better to use the default format. The biggest argument against using the second format is that for string parameters the logging driver doesn't attempt to escape the string. While this isn't a security concern at all, it does make it harder to read the SQL and understand it correctly. The format is similar for CallableStatements, but the driver includes information about the type of the parameter.

Advantages of the Driver

I have used this driver beyond just debugging my SQL queries when applications don't behave as I expect. When using Hibernate, I have found that many developers don't understand how many queries a simple query can invoke. By enabling the logDriver, I have been able to isolate performance problems in my Hibernate-based applications and use that information to modify my mapping files to reduce the number of SQL queries used.

Because the logDriver intercepts all SQL queries regardless of how they are invoked, I have found this solution to work better than the options that Hibernate provides. In most of my DB applications, I use a combination of either EJBs or Hibernate with direct JDBC calls. By moving the logging into a single location, I am able to track all SQL interaction with a single configuration change. This can be invaluable when QA reports a bug that can't be explained quickly.

Limitations with the System

For all of the benefits of the logDriver, there are a few limitations. First, you can't enable logging of SQL statements on a per-class basis. Ideally, you would be able to configure the logDriver to log queries for the Foo class, but not the Bar class. If the log statements were embedded in the classes, this would be possible with any of the logging systems; however, since the logging statements are in a common class, it can't be done natively. One possible solution to this is to include a configuration system for the logDriver that would allow the driver to inspect the call stack to determine if the logging statements should be executed. Because this system is intended for developers, it's easier to ask developers to isolate the methods called.

The second limitation is that this solution has a big impact on performance. The more DB operations you perform, the slower this driver performs. This makes sense, because logging is always a potential performance problem for Java applications, and the point of this solution is to log a lot of data. While logging can slow down the application, again this solution is intended for development, not production, so performance should not be a big consideration.

The final limitation is that we cannot log the results of the queries. The problem is that not all ResultSets are rewindable, so if we try to log the ResultSet, it is possible that we will consume the data, making it unusable for the application. While logging that data would be useful, the implications of doing so make it impractical.

"There are some problems that are found throughout a project, and the solution to those problems shouldn't require programmers to remember to add special logic every time one of these problems comes up"

Configuring the Logging Driver

Configuration of this driver is relatively straightforward. Instead of using your standard JDBC driver, configure your application to use net.rkbloom.logdriver.LogDriver, and your connection URL should be changed to use the following format:

```
    jdbc:log:real_driver_class:real_jdbc_connection_string
```

If your original connection URL was "jdbc:oracle:thin:@local-host:1521:FOOBAR", your new URL would be: "jdbc:log:com.oracle.jdbc.OracleDriver:oracle:thin:@localhost:1521:FOOBAR". The logdriver.jar file must be on your classpath, but so must the JAR file with the original JDBC driver. Under the covers, the logDriver will create an instance of your desired JDBC driver and call it for all operations. With just that change your application will be using the logDriver, but you won't see any log messages by default.

To enable logging, you must configure the logging system. The logDriver uses log4j for all logging, so you will need to configure your log4j.properties file to log all net.rkbloom.logdriver classes to log in debug mode. The logDriver uses debug mode for all logging because of the performance implications of doing so much logging. It would be possible to modify the driver to use either commons-logging or the java.util.logging, but I only use log4j, so I went for the easiest solution.

Conclusion

I hope that I have shown you a new way to debug and solve DB problems in your Java applications. I would like to thank Jonathan Cobb for creating the original version of this driver, which inspired me to re-create it when I began to encounter similar problems.

Meet the *JDJ* Editors

We thought it was time that the readers of *JDJ* had a chance to meet the editors, those individuals behind the scenes who work tirelessly to bring you the best articles about Java in particular and *i*-Technology in general.

Over the next few issues, the editors will provide a brief glimpse into their daily lives, their likes and dislikes, why they like to write, and more.

Joe Winchester
Desktop Java Editor

Q *What is your primary job?*

JW: It's the development of software tools for IBM. My specific projects at the moment are the Eclipse Visual Editor project and Rational Application Developer for Java.

Q *What is your typical day like?*

JW: I deal with the customers and users. I probably spend one hour dealing with e-mails two to three times a day as well as newsgroup traffic and customer questions. The rest is spent coding for some of the IBM tools I work on, usually knee-deep in the debugger and screen sharing and on the phone with colleagues in the U.S. (I work in the UK) trying to solve problems together.

Q *Why do you write for* JDJ?

JW: Before working for IBM I was a user of its tools and hardware. Each time I met someone from IBM I'd beat them up on why there weren't more articles and things written down, and why information was hard to find. I really enjoy reading to find out information for my job and I like to give back by doing the same – writing what I know. I think everyone who has something to say should write it down, especially in the situation where you can't find the information you need and so you do the research and fact-finding to discover it; write and publish it so the next person doesn't suffer the same pain you did.

Q *Do you blog?*

JW: Not as much as I should, but I do on http://www.joewinchester.javadevelopersjournal.com/. I don't read blogs that I think tend to be too "me, me, me" and not that informative. I much prefer sites written by people like Paul Graham (www.paulgraham.com) or Joel Spolsky (www.joelonsoftware.com). If I had the time I'd like to try to create sites like theirs.

Q *What do you like about Java?*

JW: The community of people around it, especially in the tools space where I am fortunate enough to work with Eclipse. I also enjoy hearing about and seeing what NetBeans is up to, and I think Java is a very fast-moving and dynamic community. I enjoy the language, although I wish it was as good as Smalltalk, which is my first and only real programming language love.

Q *What don't you like about Java?*

JW: Strong typing as this leads to lots of deprecated methods, and also constructs like ((MyClass) getFoo()).doSomething(). In Smalltalk there was soft typing, which didn't have this and it had the error "doesNotUnderstand", but this has just been replaced with ClassCast-Exception. The OO patterns you can do in Java are less flexible than in Smalltalk and I don't like interfaces. I'd prefer dynamic typing and a better way of describing behavior.

Q *What would be a perfect job for you?*

JW: Teaching sailing in a place where the water is warm. Actually I'm pretty fortunate where I am now – I live five minutes from work and five minutes from where my two sons go to school, so I consider myself lucky.

"What I like about Java is the community of people around it, especially in the tools space where I am fortunate enough to work with Eclipse"

– Joe Winchester

"I write in JDJ to encourage other Java programmers to think about what they are doing and to also look at the big picture where business is concerned"

–Jason Bell

Q *What's the most exciting project you are working on now?*

JW: The Eclipse Visual Editor – I get to work with smart colleagues, companies that extend and leverage it, and when I visit customers and help them I get a big kick out of seeing something that started very small at IBM being used successfully and helping people solve their day-to-day problems. It's fun.

Q *What's your hobby?*

JW: Sailing.

Q *Do you have children? If yes, would you like them to be computer programmers?*

JW: Two sons: Ben who is eight and Jared who is five. Ben wants to be a scientist who mixes potions as in Harry Potter and Jared wants to be a train driver. As long as they're happy I don't mind what they do. I think programming is fun if you enjoy math and they are both good at math, so maybe they'll hack code one day.

Q *What are the three most important things you've learned during your life in IT?*

JW: (1) Don't become a technology bigot and be prepared to adapt and change if customers and the market changes.
(2) Don't engage in flaming and always be level-headed and talk to people about the facts and content and disconnect from emotional arguments.
(3) Keep an open mind about technologies you don't know and don't feel threatened by new ideas and technologies; instead learn and understand them.

Jason Bell
Contributing Editor

Q *What's your primary job?*

JB: I'm a freelance Web applications consultant, mainly concentrating on Java systems. I'm also the founder of Aerleasing, which auctions aircraft and aircraft engines.

Q *What's your typical day like?*

JB: Every day is different; as a consultant working from home there's no such thing as a typical day. Some days client communication tends to outstrip the programming. Other days you're coding and coding and coding to get to the deadline. Speaking of which…

Q *Why do you write for* JDJ*?*

JB: To encourage other Java programmers to either think about what they are doing and also to look at the big picture where the business is concerned. We IT types sometimes have tunnel vision that doesn't leave the cubicle.

Q *Do you blog?*

JB: I did. I don't now for a number of reasons. Time is the first. Second, I never struggled to be a "personality" in the Java world. Third, I'm getting really bored with the medium.

Q *What do you like about Java?*

JB: The community support and the wealth of useful libraries out there. From 3D to business intelligence, Java has good coverage of most things.

Q *What don't you like about Java?*

JB: There was a time when I so yearned for regular expressions to be in the core JDK; that finally happened in 1.4. There are some Java bigots out there, but that's not a complaint about Java, it's a complaint about the people.

Q *What would be a perfect job for you?*

JB: A fine art photographer.

Q *What's the most exciting project you are working on now?*

JB: My photography – I have an exhibition coming up in June.

Q *What's your hobby?*

JB: Programming :)

Q *Do you have children? If yes, would you like them to be computer programmers?*

JB: No, never. If they came to me when they were 16 and really, really, really wanted to do it though, I wouldn't stop them.

Q *What are the three most important things you've learned during your life in IT?*

JB: (1) Most clients don't give a hoot about methodologies.
(2) Never believe all the hype on the Internet.
(3) You are only as good as your last project.

Q *What question, not asked here, are you burning to answer?*

JB: Yes, I'll take the money in a brown envelope as you promised. Now which wash basin did you say you would leave it under.

Oracle SOA Suite

An integrated and standards-based platform

by Feroze Mohammed and Lawrence Pravin

Service-oriented architecture is an architectural approach to building software applications as a collection of reusable business services. Interest in SOA is growing within the business community. To address the need for an infrastructure to enable the building of service-oriented applications, a new breed of SOA suites is emerging. Vendors such as Oracle, BEA Systems, and IBM now provide platforms specifically focused on SOA applications. In established, complex IT environments with diverse applications – including legacy applications that are tied together with custom adapters, and business scenarios that demand extensive partner interaction – the move to SOA may appear challenging. However, new SOA suites make it easy. We at Sierra Atlantic decided to review one of them – Oracle SOA Suite.

Sierra Atlantic has more than a decade of experience implementing application integration solutions. We've worked with a variety of vendors' products and have seen the integration landscape mature over the past six years.

Building an SOA Solution

Building a typical SOA application involves the following activities:

1. ***Building services:*** This may include service-enabling existing business logic using application or technology adapters, or writing new business logic, in J2EE or another language.
2. ***Enabling service communication:*** Services must be able to reliably communicate with each other and with back-end applications. SOA encourages loose coupling. Capabilities such as messaging, data transformation, and message routing, which are often captured in an Enterprise Service Bus, are key.
3. ***Wiring services together into business flows:*** Implementing business processes as orchestrations of services enables easier change. High-level representations of business processes, such as Business Process Execution Language (BPEL), an XML-based language for expressing service orchestration, provides the basis for agile business processes.
4. ***Securing services:*** Within an SOA environment, you must be able to effectively and consistently secure and manage services, and apply different policies (security, auditing, logging) depending on who or what is interacting with a particular service. You should be able to do this without changes to the services.

5. ***Optimizing services:*** A very important capability within an SOA environment is the ability to effectively monitor services and events, which is captured in a Business Activity Monitoring (BAM) solution. BAM provides users with an event aggregation and correlation platform that allows for building a state model defining relationships between various events that impact the operations business KPIs.

Oracle says it has the most comprehensive and mature SOA platform in the market. At first glance, we were impressed to see that its suite addresses all the requirements we mentioned above:

- Oracle JDeveloper to build services, to service-enable existing assets
- Oracle ESB for messaging and routing
- Oracle BPEL Process Manager to wire services together into business flows
- Oracle Web Services Manager for security and policy management for services
- Oracle BAM for business-level monitoring and optimization of business services and processes
- Oracle Application Server 10g R3/ Oracle Fusion Middleware – the J2EE 1.4 runtime on which all this infrastructure may be deployed

In this review, we focused on three components of the Oracle SOA Suite – Oracle BPEL Process Manager (Oracle BPEL PM), Oracle Web Services Manager and Oracle Business Activity Monitoring (Oracle BAM). We give an overview of each of these components, and show each of their capabilities in light of an order-processing scenario – a problem well suited to the application of SOA. We didn't review the full capabilities of Oracle JDeveloper, Oracle Application Development Framework (Oracle ADF), Oracle ESB and the core J2EE-compliant runtime. This is because you can develop apps for the Oracle SOA Suite with third-party tools such as Eclipse; the Suite's components can be deployed to any J2EE application server; and Oracle BPEL Process Manager, Oracle BAM, and Oracle Web Services Manager can integrate with any standards-based messaging solution.

Oracle BPEL Process Manager

Business processes are at the heart of any business. They touch different applications, persons, and business partners during their execution and evolve according to changing business requirements. Some of them are automated with technology solutions. SOA can make a key contribution to enabling easy-to-develop, maintain, and easy-to-adapt business processes.

BPEL makes it easier for organizations to adapt to changing business needs on the fly. Oracle BPEL Process Manager, which is in its fourth release, has been on the market since 2001 (when it was acquired from Collaxa). It includes a BPEL engine, a management console, and a graphical interface for wiring services, as well as robust messaging and routing capabilities. We like that Oracle BPEL Process Manager is engineered to run on top of all major J2EE-compatible application servers. This helps organizations preserve their existing IT infrastructure and in-house resources.

Feroze Mohammed is the senior vice president, product engineering, custom development and integrations, Sierra Atlantic, Inc. He is also responsible for Sierra Atlantic's own R&D and product development initiatives. Feroze holds an master's degree in computer applications from the University of Hyderabad.

feroze.mohammed@sierraatlantic.com

One of the most important features in any BPM tool is the ability to model business processes graphically. Oracle BPEL Process Designer offers a graphical modeling environment for this task, and its drag-and-drop features make it easy to use. It is targeted at the developer and what Oracle calls *Functional Developer/ Business Savvy.* To create a more full-featured modeling environment for business analysts, Oracle partners with pure-play BPM modeling vendors including IDS Scheer and Proforma.

Oracle BPEL Process Manager provides native support for standards such as BPEL, XML, XSLT, XPATH, JMS, JCA, and Web services, making it a good solution for enabling SOAs.

Processes interact with heterogeneous applications and, in many cases, require human intervention. These applications speak different languages. In order to connect to various systems, Oracle BPEL Process Manager ships with a set of technology and application adapters to let you connect with endpoints that are not exposed as services. These include file, file transfer protocol (FTP), advance queues (AQs), database (DBs), Oracle applications, and other ERP application adapters.

Graphical tooling is also included within JDeveloper for performing data transformations to reconcile different data formats within XML documents being transmitted between services.

Many industry experts are concerned about BPEL's lack of workflow support. Oracle has simply built an application on top of their product that implements workflow capabilities – in theory, workflow built on top of Oracle BPEL Process Manager should be portable across BPEL engines. These built-in, standards-based workflow services are linked to a BPEL process through a WSDL contract. A BPEL process assigns a task to a user or role by making a service call, and waits for a response.

Designing business processes is only one piece of the puzzle. How do you secure these services to prevent unauthorized access? How do you apply security, audit and logging policies to services? First, you need a cohesive security policy infrastructure that governs access to these services. Oracle Web Services Manager, part of Oracle's Identity Management solution, is Oracle's solution.

Oracle Web Services Manager

Security is important for any kind of distributed computing environment. Oracle Web Services Manager, which Oracle obtained last year through its acquisition of Oblix, Inc., lets you define and implement security and operational policies for services. As a developer, you don't want to worry about applying security policies - authentication, encryption and digital signatures. Web Services Manager enables these policies to be enforced from outside the service.

As with Oracle BPEL Process Manager, we like that Oracle Web Services Manager supports multiple Web services platforms and providers including BEA Systems, IBM, Microsoft, Netegrity, TIBCO, and Verisign. It also provides out-of-the-box support for multiple transports such as HTTP, HTTPS, JMS, and IBM Websphere MQ, and multiple messaging models including synchronous and asynchronous messaging.

Oracle Web Services Manager includes two enforcement components that ensure maximum deployment flexibility: *policy gateways,* which are deployed before a group of applications or services and intercept inbound requests to services, and *policy agents,* which run in-process with the service that is being secured.

Oracle Web Services Manager policies are a set of operational tasks that are performed when service requests are processed, and the responses between a service client and a service provider at specified policy-enforcement points. Each task is implemented as a policy step that addresses a specific operation (such as authentication, authorization, encryption, decryption, security signature, token or credential verification, transformation, auditing, logging) that's performed on either a Web services request or a response message.

We like the management console, which also monitors each Web services' performance. The graphical dashboard shows overall statistics, including security metrics such as unauthorized access attempts, and service figures including the average service failure rate and average registered service latency. You can drill down into the dashboard by service to see statistics on individual operations. You can also define and monitor individual service levels.

Oracle BPEL Process Manager and Oracle Web Services Manager let you design, orchestrate, and secure services. But how do you gain real-time visibility into business entities and their interactions?

Oracle Business Activity Monitoring

The promise of BAM is to push information to users via visual dashboards and alerts, helping them improve operational effectiveness and make informed decisions. As it relates to SOA, services and events, which provide real-time visibility into business processes, people, and systems need to be monitored. The ability to aggregate service metrics and deliver actionable information on critical business service parameters to business users is also key.

The Oracle BAM architecture uses functions including messaging, data integration, data caching, analytics monitoring, alerting, and reporting to collect, analyze and deliver critical information. We like that these functions support integration with different industry-standard business applications such as JMS, JCA, Web services, file system, MQs (IBM, MS, Sonic), TIBCO, Webmethods, and BEA WebLogic. This means that Oracle BAM can collect real-time information from heterogeneous environments.

In fact, Oracle BAM is generally targeted at business users. However, before they can start creating and viewing reports, someone (the developer/ architect) must create the underlying data model and populate it. Oracle BAM segregates the two tasks. While developers can use Oracle BAM Architect to create the data objects and rules, business users can use Oracle BAM Active Studio's simple interface to create reports. This means that you as the developer just plug-in the data sources – the business users can build their own dashboards.

After reviewing the capabilities of the product suite, we wanted to model a real-world scenario to more completely assess its capabilities. For this, we built a use case that captures a typical order-processing business process.

The Business Scenario: Order Processing

An order-processing application is a classic example of an SOA application. Imagine a fictional original equipment manufacturer that sells widgets to its customers.

Lawrence Pravin is the product manager, Process Integration Packs, Sierra Atlantic, Inc. He has over 10 years of experience in packaged applications, and has deep integration expertise with Oracle, PeopleSoft, Siebel and SAP applications. Lawrence holds a computer science and engineering degree from the University of Madras.

lawrence.pravin@sierraatlantic.com

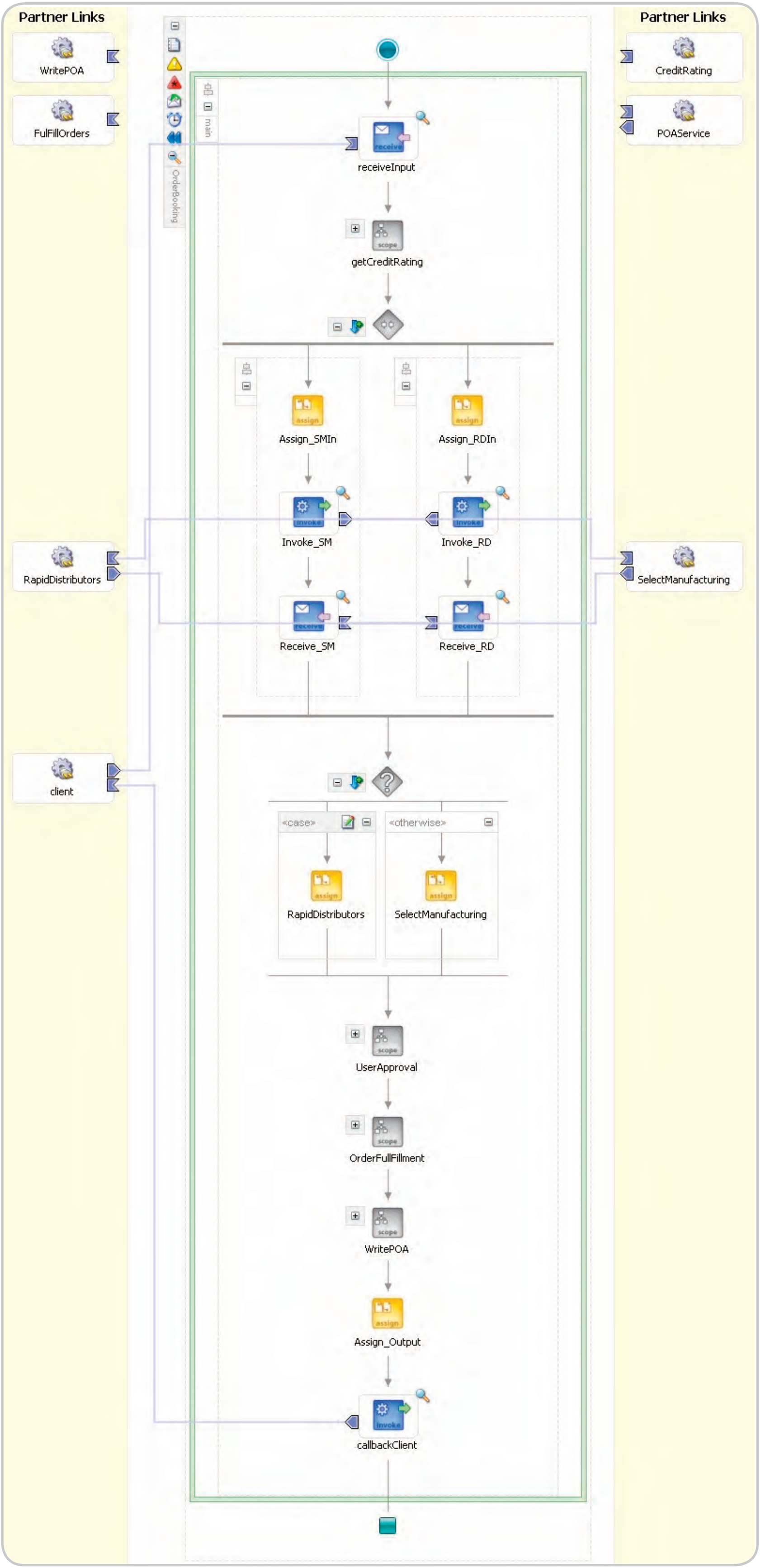


Figure 1 Order Booking Process in Oracle BPEL Designer

The company uses a virtual inventory business model. Let's say the company deals with two primary suppliers to fulfill orders – Select Manufacturing and Rapid Distributors. The company wants to build an order processing application using SOA principles, secure it, and provide operational dashboards to managers. In this section we show how we can do this using Oracle SOA Suite. Oracle BPEL Process Manager orchestrates all the existing services in the enterprise for order fulfillment with the right supplier; Oracle BAM is used to monitor the order status and other issues; and Oracle Web Services Manager is used to secure and manage the Web services.

In the first step of the process, the company receives sales orders from different channels. Customers are identified from the orders and checked against a credit rating service. The orders' line items are then passed to the two suppliers for price quotes. The process collects the quotes and selects the supplier with the lowest quoted price. This information is passed to the business managers for approval along with other details, including the sales price and the supplier-quoted price, for margin reviews. Once approved, the order is sent to fulfillment. An acknowledgment is sent to the order-placing channel, confirming the acceptance/rejection of the order.

Let's examine the different activities and features of Oracle BPEL Process Manager and see how a developer designs the order processing process. Figure 1 shows the order process in Oracle BPEL Designer:

The order-processing scenario uses various Web services from different suppliers for quotes. For this evaluation, we used a database to insert approved and rejected orders. We liked being able to configure these adapters with graphical wizards. This, in conjunction with many out-of-the-box adapters, will make developers happy.

Since more than one supplier can provide a quote, these services should be called in parallel. We were able to quickly do this by dragging and dropping the BPEL parallel flow activity into the process and adding the necessary logic to it.

To process the orders, you must send them to different applications.

This data must be transformed into an application-specific format. Using the XSLT mapper for this transformation was easy, thanks to its drag-and-drop nature. The transform activity takes source and target objects and transforms them through one-to-one mapping, which creates XSL. The XSLT mapper supports various built-in functions.

We implemented an order approval workflow by dragging and dropping the User Task activity into the BPEL flow.

During order processing, users must be notified about their order status. Oracle BPEL Process Manager offers a cool notification services feature: you can notify users by e-mail, SMS, or VoIP by dragging and dropping the notification service into the workflow and configuring a mail server.

In an SOA world, where multiple applications from various company departments take part in a process, business exceptions become critical. With just a few mouse clicks in the Scope Activity, we were able to add fault-handling to our process. Oracle BPEL Process Manager provides easy-to-use constructs such as compensation handlers and catch activities to manage business exceptions.

Moving on to administration, the Oracle BPEL Process Manager console is a great tool for administrators to gather information at the process level. We were able to deploy and trace each and every one of our processes, and it was helpful to be able to visually inspect any process and drill down into it. You can even see the sequence of interactions and XML payloads for each in-flight process in the BPEL Console.

In our scenario, different users from different channels access the process, and you must secure and manage these services. Let's see how Oracle Web Services Manager does this.

Once the order is submitted, you must authenticate the customer information before performing the credit check. The credit rating service should return the results within 30 seconds. The process should monitor the number of authentication failures; if failures exceed a 20% threshold, it should notify the customer and manager. Interactions should also be logged for auditing purposes.

It was fairly easy to configure the gateway using the Web-based console included with Oracle Web Services Manager. The gateway virtualizes the underlying Web service, hiding the address details of the service. We were able to successfully configure the gateway to route customer orders based on the size of the order (content-based routing), but we would have also liked to see the routing based on the source of the message, such as the IP address (context-based routing).

Next, using the console, we registered the service with the gateway by providing the WSDL URL of the service. This generated a new URL to access the Web service. At runtime, when the gateway receives a request, it routes the request to the actual Web service.

We wanted to see how easily we could define policies on Web services. Using the Policy Manager of the Web Services Manager, we defined the steps as:

- Decrypt the XML payload containing the order and customer information
- Perform customer authentication against an LDAP directory
- Log failures
- Monitor a SLA indicating that the credit check service should complete its execution in 30 seconds

This was easy to define using the console. It's a great idea to group all these steps into a policy pipeline that can be reused. This lets developers build a policy template and apply it to other Web services.

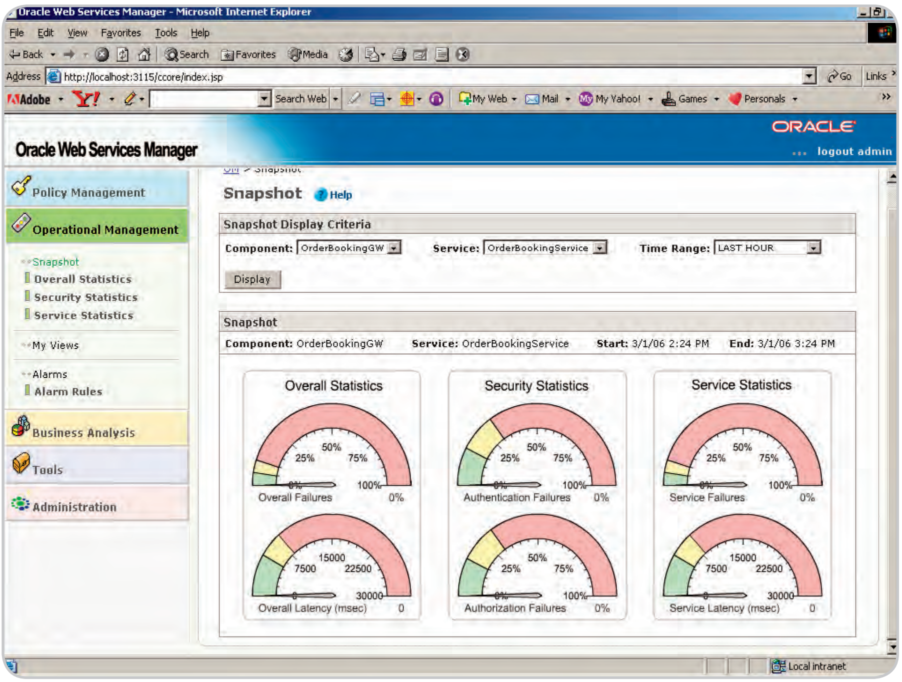
In addition to the pre-built policies included with the product, Oracle Web Services Manager lets you define custom policies. This is handy when you have fine-grained security requirements within your organization.

We also defined some service levels to see how easy it is to define SLAs in Web Services Manager. The service level we monitored had a number of authentication failures on a particular service and set an alert when it reached a threshold. This was also easy to define since Oracle Web Services Manager logs all interactions, whether they're failures or successes. Figure 2 shows the monitoring dashboard for our order booking service.

Now let's see how Oracle BAM can monitor the business and create operational dashboards for business users.

In our scenario, we wanted to monitor some KPIs, including the number of:

- Pending orders (those waiting for suppliers to respond)
- Rejected orders (due to bad customer credit)
- Rejected orders (due to customers with orders greater than accounts payable)



Snapshot Display Criteria

Component: OrderBookingGW Service: OrderBookingService Time Range: LAST HOUR

Display

Snapshot

Component: OrderBookingGW Service: OrderBookingService Start: 3/1/06 2:24 PM End: 3/1/06 3:24 PM

Figure 2 Dashboard to monitor security policies in OWSM

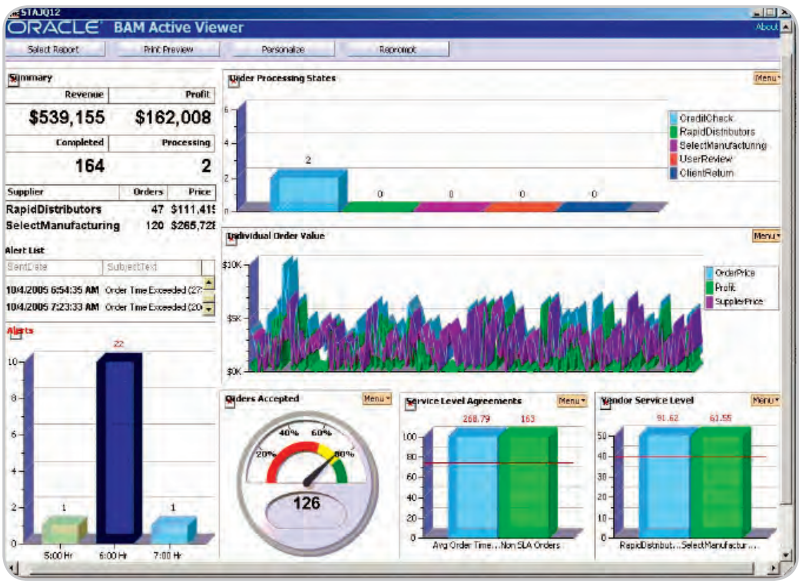


Figure 3 Oracle BAM monitoring KPIs

To continuously monitor business performance, there must be direct communication with business processes. Oracle BAM can collect business events from any source. However, in our particular case, we could pre-instrument the BPEL order processing process with *sensors* at various steps in the flow, and route the data to Oracle BAM, JMS or database. Applying sensors to steps within a BPEL Process proved to be very easy – you just configure the step to publish into Oracle BAM. Whenever the order booking process is started or completed, the data is sent from Oracle BPEL PM into Oracle BAM and the dashboard – in this way we could monitor average completion times.

We used the Oracle BAM Architect tool to create data objects from the data coming in on the JMS queue from BPEL PM. Once we defined them, we used Oracle BAM Active Studio to design our reports without worrying about the underlying meta-data or rules. This demarcation of responsibility (developer vs. business user) not only facilitates cleaner development but also lets the business community define its own dashboard – as long as the data sources have been registered with BAM, a business user can build a real-time dashboard with them.

Creating reports was intuitive, thanks to Oracle BAM Active Studio's drag-and-drop interface. Once deployed, the dashboard offered a rich collection of reports such as multiple filters, drill into, drill across, moving time-slice window, alerts, and navigation across objects.

We configured a dashboard that showed us incomplete orders. We were able to drill down from the dashboard to the order level to find out exactly which orders had the problem and analyze them individually. It would have been nice to drill down from the order to the actual BPEL process instance, which we managed to do by rolling up our sleeves. Figure 3 shows the BAM dashboard that we built.

Conclusion

Oracle SOA Suite offers a comprehensive and standards-based functionality. All of its components are designed with developers in mind. Whether it's ease of use or a high degree of customization, Oracle SOA Suite helps developers' productivity. Oracle BPEL Process Manager is a good tool for orchestrating services. However, it doesn't offer extensive business process analysis and analysts might find it challenging to model business processes with Oracle BPEL Designer unless they had help from a developer. Oracle Web Services Manager is ideal for securing the entire SOA infrastructure including services and business processes developed using Oracle BPEL Process Manager. Oracle BAM provides an intuitive framework for defining and monitoring KPIs.

One of the greatest advantages of leveraging tools from a single vendor is a high level of product integration. Oracle has done a good job of integrating its development and run-time environment across all components of Oracle SOA Suite. However, it falls a little short in offering a 100% integrated administration console; we hope this is going to be addressed in future releases. Finally, we were impressed with the interoperability Oracle has built into its products. Oracle BPEL Process Manager can work with all J2EE application servers; Oracle Web Service Manager can secure Web services running on BEA or IBM; and Oracle BAM can monitor data from any data source. This will help developers build solutions without having to rip apart existing applications.

Whether you're embarking on an SOA initiative or already in the middle of an SOA project, we recommend you review Oracle SOA Suite. Its combination of Oracle JDeveloper, Oracle BPEL Process Manager, Oracle BAM, and Oracle Web Services Manager offers an integrated and standards-based platform to build, automate, monitor, and secure your SOA.

Rating

We rated Oracle SOA Suite on a scale of 1 to 10, using the following criteria. (These scores are based on the collective assessment of the product evaluation team. See table below.)

Criteria	Description	Average Score
Capability	How satisfied we were with the functions and features.	8.5
Usability	How satisfied we were with the suite's ease of use.	9.0
Performance	How satisfied we were with the response time or speed.	9.0
Installation initialization.	How satisfied we were with the ease of installation and	8.0
Administration	How satisfied we were with the administration of the suite.	7.5
Overall	Considering all the above aspects of the experience, how we rate our overall level of satisfaction.	**8.4**

JAVA DEVELOPER'S JOURNAL

Advertiser Index

Advertiser	URL	Phone	Page
AJAX Seminar	www.ajaxseminar.com	201-802-3022	48-49
Altova	www.altova.com	978-816-1600	4
Extentech	www.extentech.com/jdj	415-759-5292	27
Fiorano	www.fiorano.com/downloads	800-663-3621	17
Instantiations	www.instantiations.com/rcpdeveloper	800-808-3737	9
InterSystems	www.intersystems.com/ensemble11p	617-621-0600	7
IT Solutions Guide	www.itsolutions.sys-con.com	888-303-5282	59
iTVcon.com Conference & Expo	www.itvcon.com	201-802-3023	45
JadeLiquid Software	www.webrenderer.com	+61-3-6226-6274	13
Java Developer's Journal	www.jdj.sys-con.com	888-303-5282	47
JavaOne Conference 2006	java.sun.com/javaone/sf		31
LinuxWorld Conference & Expo	www.linuxworldexpo.com/boston		37
Microsoft Corp.	www.learn2asp.net/webcasts		37
Northwoods Software Corp.	www.nwoods.com	800-434-9820	41
ODTUG	www.odtug.com		39
OpenAccess Software, Inc.	www.openaccesssoftware.com/jdj	888-805-ODBC	15
Parasoft Corporation	www.parasoft.com/jdjmagazine	888-305-0041	Cover IV
Quest Software	www.quest.com/911	949-754-8000	Cover II
SOA/EOS Conference	www.soaeosconference.sys-con.com	201-802-3022	60-61
Software FX	www.softwarefx.com	800-392-4278	Cover III
Synaptris	www.intelliview.com/jdj	1-866-99IVIEW	19
TIBCO Software Inc.	www.tibco.com/mk/gi	800-420-8450	11
WebAppCabaret	www.webappcabaret.com/jdj.jsp	866-256-7973	25

BONUS TRACK ADDED! (FREE $1,295 VALUE!)
"Real-World AJAX" 2-Day Seminar
SOA
WebServices
& Enterprise
OpenSource
Conferences

The #1 i-Technology Educational and

Register at... www.SOAEOSConference.sys-con.com

The **10th International SOA Web Services Edge Conference**, colocated with the First Annual **Enterprise Open Source Conference**, will take place on **June 5-6, 2006** at the historical **Roosevelt Hotel in New York City**. SOA Web Services Edge and Enterprise Open Source Conference together deliver the #1 i-technology educational and networking opportunity of the year.

This year, more than 100 distinguished conference faculty members will present **96 cutting-edge SOA, Web Services and Enterprise Open Source topics** in educational classes, panels, and keynotes in six simultaneous tracks for two information-packed days, plus a bonus track, a **"Real-World AJAX"** two-day seminar, a $1,295 value!

The conference program does not merely present a comprehensive view of all the development and management aspects of integrating a **SOA strategy** and an **Open Source philosophy** into your enterprise, its organizing principle is that delegates will go away from the intense two-day program replete with why-to and how-to knowledge delivered first-hand by industry experts.

According to analyst firm Gartner Group, **by 2008 more than 60 percent of enterprises will use SOA** as the guiding principle when creating mission-critical applications and processes. **"Businesses that ignore the potential of SOA will find themselves outpaced by rivals who improve their agility and transform themselves into new kinds of enterprises,"** says Gartner analyst Yafim Natis.

Meanwhile, the **Open Source revolution** has spread into traditional areas of enterprise IT, with all major vendors and a global group of **Open Source entrepreneurs** developing the technology and delivering the solutions that are changing the way the world does business and the way governments serve their citizens.

Receive FREE WebCast Archives of all Entire Conference!

The best news for this year's conference delegates is that your "Golden Pass" registration now gives you full access to all 96 conference sessions. We will mail you the complete content from all the conference sessions in seven convenient DVDs after the live event takes place.

SOA WebServices • OpenSource • AJAX
On-Demand Webcast Archives
DVD

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

Receive 8 Days Worth of Education F

FACULTY

Jeffrey Barr
Amazon
As Web Services Evangelist for Amazon.com, Jeff Barr focuses on creating developer awareness for the Amazon software platform. He has a longstanding interest in Web services and programmatic information interchange. Jeff has held development and management positions at KnowNow, eByz, Akopia, and Microsoft, and was a co-founder of Visix Software. Jeff's interests include collecting and organizing news feeds using his site, www.syndic8.com . He holds a Bachelor's Degree in Computer Science from the American University and has done graduate work in Computer Science at the George Washington University.

Israel Hilerio
Microsoft
Israel Hilerio is a program manager at Microsoft in the Windows Workflow Foundation team. He has 15+ years of development experience doing business applications and has a PhD in Computer Science.

Adam Kolawa
Parasoft
Adam Kolawa, Parasoft co-founder and CEO, is considered to be a visionary in his field. In 1983, he came to the United States from Poland to pursue his Ph.D. In 1987, he and a group of fellow graduate students founded Parasoft to create value-added products that could significantly improve the software development process. Kolawa's years of experience with various software development processes has resulted in his unique insight into the high-tech industry and the uncanny ability to successfully identify technology trends. As a result, he has orchestrated the development of numerous successful commercial software products to meet growing industry needs to improve software quality.

Jason Levitt
Yahoo!
Jason Levitt, Technical Evangelist on creating Flash-based Yahoo! Maps applications.

Duane Nickull
Adobe
As senior standards strategist for Adobe Systems, Duane Nickull is responsible for managing Adobe's participation in OASIS and UN/CEFACT, as well as ensuring that Adobe's enterprise solutions support emerging XML standards. Previously Mr. Nickull co-founded Yellow Dragon Software Corporation, a privately held developer of XML messaging and metadata management software, recently acquired by Adobe. Mr. Nickull currently serves as a vice chair of the United Nations Centre for Facilitation of Commerce and Trade (UN/CEFACT) where he oversees the United Nations Electronic Business strategy and architecture.

Bob Pasker
Azul
Bob Pasker is deputy CTO with Azul Systems. He has been designing and developing networking, communications, transaction processing, and database products for 25 years. As one of the founders of WebLogic, the first independent Java company (acquired by BEA Systems in 1998), he was the chief architect of the WebLogic Application Server. Bob has provided technical leadership and management for numerous award-winning technologies, including the TribeLink series of routers and remote access devices, and the TMX transaction processing system. Bob graduated magna cum laude and Phi Beta Kappa from San Francisco State University and holds a Masters degree from Brown University.

Brian Behlendorf
CollabNet
Brian Behlendorf founded CollabNet, with O'Reilly & Associates, in July 1999. The company provides tools and services based on open source methods. Before launching CollabNet, Behlendorf was co-founder and CTO of Organic Online, a Web design and engineering consultancy located in San Francisco. During his five years at Organic, Behlendorf helped create Internet strategies for dozens of Fortune 500 companies. During that time, he co-founded and contributed heavily to the Apache Web Server Project, co-founded and supported the VRML (Virtual Reality Modeling Language) effort, and assisted several IETF working groups, particularly the HTTP standardization effort.

Marc Fleury
JBoss
Born in Paris in 1968, Marc Fleury got his Ph.D in physics from the Ecole Polytechnique in Paris. He started in Sales at Sun Microsystems France and then moved to the US where he worked on early java enablement of SAP at SAPLabs. Marc started the JBoss project in 1999. An ex-Lieutenant in the paratroopers, Marc holds a degree in Mathematics from the Ecole Polytechnique, a master in Theoretical Physics from the Ecole Normale ULM and was a visiting scientist at MIT during his thesis. Marc's research interest focuses on aspect oriented middleware.

Andy Astor
EnterpriseDB
Andy is President and CEO EnterpriseDB, the world's leading enterprise-class, open source database company. Previously, Andy was vice president webMethods, leading the company's open source, standards, and Web services agendas. Andy was elected twice to the Board of Directors of the Web Services Interoperability Organization (WS-I), and led WS-I's marketing efforts. Prior to joining webMethods, Andy was vice president at D&B, where he led worldwide development of all on-line products. His work at D&B included the development and launch of one of the earliest commercial Web services.

Mike Milinkovich
Eclipse.org
Mike Milinkovich has held key management positions at Oracle, WebGain, The Object People, and Object Technology International Inc. (which subsequently became a wholly-owned subsidiary of IBM), assuming responsibility for development, product management, marketing, strategic planning, finance and business development. Mike earned his MS degree in information and systems sciences and a bachelor of commerce degree from Carleton University in Ottawa, Canada.

Peter Yared
ActiveGrid
Peter Yared is the founder and CEO of ActiveGrid. Most recently, he was CTO of Sun Microsystems's Liberty Network Identity initiative. Mr. Yared was also CTO of Sun Microsystems Application Server Division. Before its acquisition by Sun, Mr. Yared served as CTO of NetDynamics, which pioneered the then-leading J2EE application server. Earlier, Mr. Yared was founder and CEO of JRad Technologies, an enterprise Java company acquired by NetDynamics. Additionally, Mr. Yared was Chief Architect of client/server products at object-oriented tool maker Prograph International and the architect of several mission-critical systems deployed by U.S. government agencies and the GED Testing Service.

David Temkin
Laszlo
David Temkin is Chief Technology Officer of Laszlo Systems, Inc. In this role, he has positioned the company to become the next technology standard for rich Internet applications. Under his direction, Laszlo developed its patent-pending open-source product suite and extended operations to both coasts of the United States. Before founding Laszlo, Temkin was senior director of engineering at Excite@Home where he led a team of 55 engineers, designers and technical writers responsible for developing the company's consumer software. Prior to Excite@Home, Temkin was an engineering manager in the Newton division at Apple Computer and developed enterprise software at EDS.

Kevin Hakman
TIBCO
Kevin Hakman is Co-founder, TIBCO General Interface, TIBCO Software Inc. Prior to TIBCO General Interface, he was the co-founder of Versalent Inc. a leading provider of enterprise client technology. Prior to Versalent, he founded a series of successful emerging Internet technology and ecommerce ventures. He has also written for eBusiness Journal and HotWired.

Coach Wei
Nexaweb
Coach Wei currently serves as CTO for Nexaweb, which develops the leading XML-based rich client technology platform for building and deploying Enterprise Internet Applications. Previously, he played a key role at EMC Corporation in the development of a new generation of storage network management software. Coach is a graduate from MIT, holds several patents, and is an industry advocate for the proliferation of open standards.

Luis Derechin
JackBe
Luis Derechin is CEO and Co-Founder of JackBe. Mr. Derechin has over 12 years of entrepreneurial and management experience. He has been part of the founding team of successful startups, including a catalogue retail company that achieved $15M in sales.

Jouk Pleiter
Backbase
Jouk Pleiter is the CEO of Backbase, a leader in the field of Rich Internet Applications and AJAX development software. Backbase's clients include ING, ABN AMRO, TNT, KPN, Comsys and Heineken. Backbase operates globally with offices in San Mateo (North America) and Amsterdam (Europe). Since 1995, Jouk has been an entrepreneur: he founded three successful Software companies. Prior to Backbase, Jouk was part of the founding team at the web content management company Tridion, where he led the product management operations, and was driving the company's efforts to become a leader in the European WCM software market.

Networking Opportunity of the Year!

SOA WebServices Edge 10th International conference

2006 ENTERPRISE> OPENSOURCE CONFERENCE

Early Bird Price (Before March 31, 2006)
Incl. Golden Pass to Both Conferences* **$1,295**

Discounted Price (After March 31, 2006)
Incl. Golden Pass to Both Conferences* **$1,495**

Conference Price (On June 5-6, 2006)
Incl. Golden Pass to Both Conferences* **$1,695**

*BONUS: "Real-World AJAX" Two-Day Seminar!

Attention Sponsors:
A Forum will be available to display leading Web services, OpenSource, and AJAX products, services, and solutions.

The Roosevelt Hotel

JUNE 5-6, 2006
Roosevelt Hotel / New York, NY

For Sponsorship Information ▸ Call 201 802-3022

TOPICS INCLUDE...

SOA Web Services Edge:
- *> Transitioning Successfully to SOA*
- *> ebXML*
- *> Orchestration*
- *> The Business Case for SOA*
- *> Interop & Standards*
- *> Web Services Management*
- *> Messaging Buses and SOA*
- *> SOBAs (Service-Oriented Business Apps)*
- *> Delivering ROI with SOA*
- *> Java & XML Web Services*
- *> Security*
- *> Systems Integration*
- *> Sarbanes-Oxley*
- *> Business Process Management*

Enterprise Open Source:
- *> Open Source Licenses*
- *> Open Source & E-Mail*
- *> Databases*
- *> ROI Case Studies*
- *> Open Source ERP & CRM*
- *> Open-Source SIP*
- *> Testing*
- *> Grid Computing*
- *> Open Source Middleware*
- *> LAMP Technologies*
- *> Professional Open Source*
- *> Open Source on the Desktop*
- *> Open Source & Sarbanes-Oxley*
- *> IP Management*

AJAX User Experience:
- *> Rich Internet Applications*
- *> Flash-AJAX Integration*
- *> Developing Interfaces*
- *> AJAX on Rails*
- *> Best Practices*
- *> Enterprise AJAX*
- *> Technology Demos*
- *> Lessons from the Frontline*
- *> Bringing Desktop Apps to the Web*
- *> AJAX Power Panel*
- *> and more...*

SYS-CON EVENTS www.EVENTS.sys-con.com

Packed into 2 Information-Filled Days!

Women Spec Leads Make History for Java Technology

Onno Kluyt

Every March here in the U.S. we mark Women's History Month in recognition of women's contribution to the progress of our world. Technology and the Java platform are benefiting as well from women's talent and dedication and an instantiation of that is women engineers' contribution to the development of Java standards through the Java Community Process Program. Several of them won the distinction of Star Spec Leads for their leadership in driving Java specifications from concept, submission, standard development, Technology Compatibility Kit (TCK), and Reference Implementation (RI) delivery.

Ekaterina Chtcherbina is one of them. She started with various projects in the area of innovative software architectures for mobile devices and mobile infrastructure at Siemens. Ekaterina focuses on developing standards related to the Java Micro Edition and until recently was co-Spec Lead of JSR 253 Mobile Telephony API (MTA) along with Eric Overtoom of Motorola; together they won the 2005 JCP Program Best Spec Lead Award for Java Micro Edition.

Ekaterina speaks passionately about Java and the community: "Java technology for me is not just a programming language, rather it is a new style of technology innovation. Java technology is not created somewhere and given to everyone as a final technology. Instead, the evolution of Java technology relies highly on community input. This makes the community feel involved in the creation of the technology that they are using. At the same time this process ensures the usefulness of Java technology for the community. These facts attracted me a lot."

Another Star Spec Lead is *Jaana Majakangas*. A senior design engineer at Nokia Corporation, Jaana has pursued a diversity of telephony software projects and made her debut in the JCP program in September 2003, quickly getting involved with Expert Groups. She attributes her success as Spec Lead of JSR 257 Contactless Communication API to excellent mentors, her colleagues Kimmo Loytana and Jere Kapyaho, also Star Spec Leads from Nokia.

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

onno@jcp.org

Even before achieving Star status, Jaana found the spec lead work deeply satisfying. "I like my current role where we create a standard in some area, and it is interesting to see how it is then implemented in actual products. This gives us feedback on how we have succeeded," she says.

Currently co-specification lead of JSR 220 Enterprise JavaBeans 3.0, *Linda DeMichiel*, senior engineer at Sun Microsystems, has also gained the distinction of Star Spec Lead. Since she became involved with the JCP, she has participated in the development of several JSRs including leading EJBs 2.0 and EJBs 2.1. Holder of a PhD in computer science from Stanford University and with over 15 years of industry experience in the areas of databases, distributed computing, and object-oriented programming, Linda provides strong leadership for the JSR 220 Expert Group and lends her expertise to the effort of making EJB components easier to use and the EJB programming model more flexible and powerful.

Pia Niemela, another spec lead from Nokia Corporation, worked hard in March with her Expert Group to take JSR 256 Mobile Sensor API to its final development stage. The specification received the final approval from the EC last month and the final release posting will follow. This JSR targeted *Java ME* applications for which it defines generic sensor functionality optimized for resource-constrained devices like mobile devices.

Congratulations to all women spec leads and experts who dedicate their time and exceptional talents to the development of Java standards and inspire other engineers to get involved with the Java community.

During the month of March a number of JSRs got closer to the finish line or crossed it. *Java Data Objects 2.0 (JSR 243)* was approved by the JCP EC in a final approval ballot. Led by Sun, this JSR's high-level objectives are to make JDO easier to use, closely align JDO with J2EE, standardize JDO's database support, and broaden the scope of JDO.

JSR 292, Supporting Dynamically Typed Languages on the Java Platform, recently submitted by Sun, was approved by the JCP EC to be developed through the JCP. This JSR initiative was prompted by the growing interest in running a variety of programming languages on the Java platform in particular scripting languages.

JSR 239 Java Bindings for OpenGL ES published its proposed final draft in March. This specification is an optional package and is inspired by the need to have access to hardware-accelerated 3D graphics from a low-level 3D graphics library.

In the final development stage, *JSR 115 Java Authorization Contract for Containers* filed its Maintenance Review draft. Developed by Sun, this JSR seeks to define a contract between containers and authorization service providers that will result in the implementation of providers for use by containers.

JSR 291 Dynamic Component Support for Java SE recently proposed by IBM was voted on and approved by the JCP EC as a new JSR to be developed under the JCP. This JSR sets out to establish an API for a dynamic component framework supporting existing Java SE environments based on the OSGi dynamic component model specifications.

JSR 253 Mobile Telephony API (MTA) co-spec led by Motorola and BenQ Corporation published its final release in March. This JSR creates a mobile telephony API and platform definition that utilizes common telephony features and is small and simple enough to be implemented on terminals with limited resources.

For details on any of these JSRs and the updates I highlighted, visit http://jcp.org and the respective JSR pages.

Don't miss the opportunity to meet in May at the JavaOne Conference with the JCP PMO, EC representatives, spec leads, and expert group members at the activities we are organizing that include JCP training sessions, BOFs, technology sessions, and the popular Java technology community event. Check http://jcp.org for the dates, times, and locations and mark your preferred event on your JavaOne Conference Calendar.

Software FX
Bringing Your Data, Business & Strategy Into Focus.

New! version 6.2
Now Includes Maps!

All New Flavors... Same Great Taste of Java!

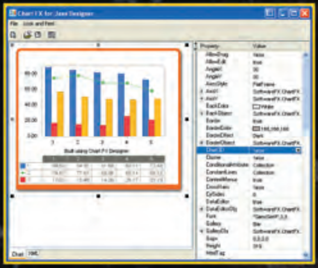


Chart FX for Java Designer

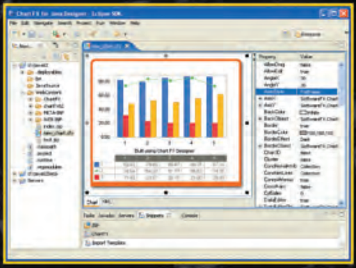


Chart FX for Java - Eclipse

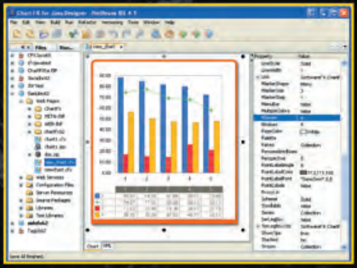


Chart FX for Java - NetBeans

Chart FX for Java 6.2

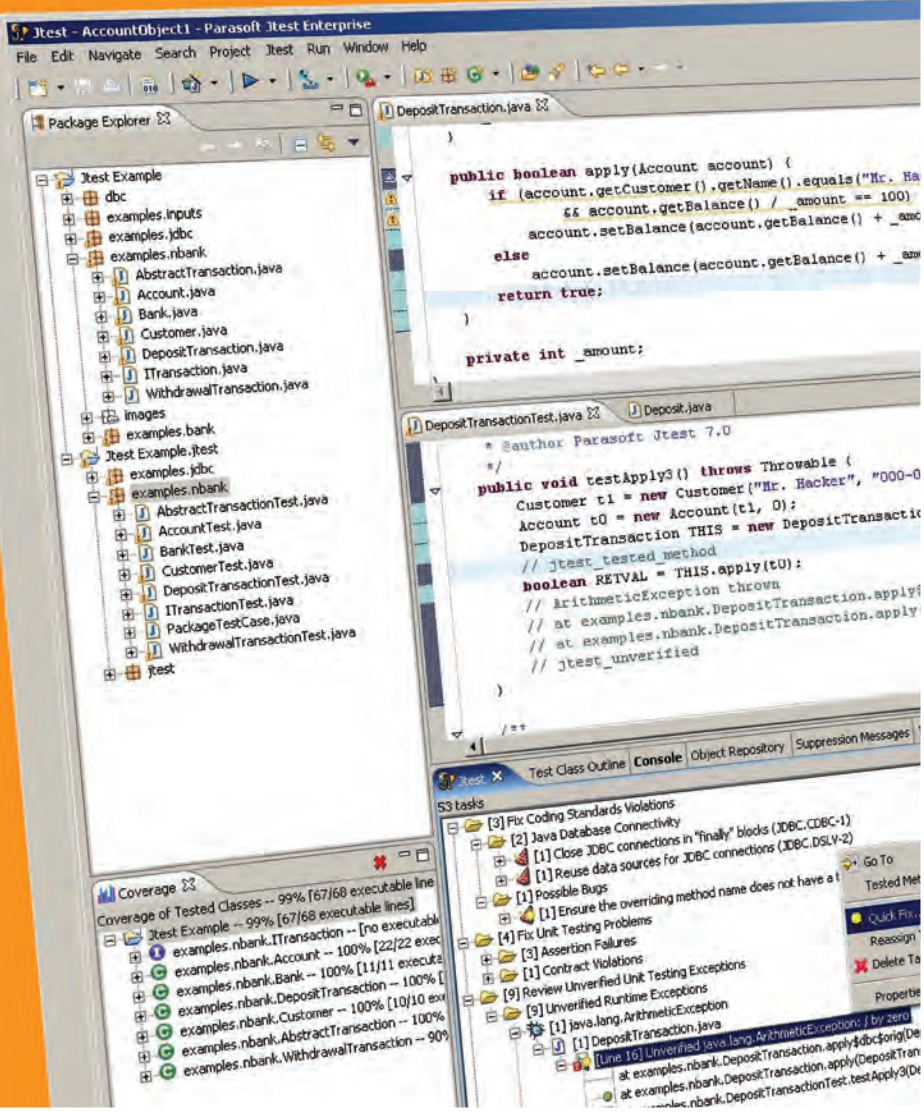
Now offers seamless integration within Eclipse and NetBeans, Chart FX for Java 6.2 brings New features include Smart & dynamic axis labeling, conditional marker and axis attributes, extended URL linking, multiple chart panes and robust annotation objects. One of the most dynamic new features is the inclusion of the Chart FX Maps Extension as a permanent fixture within Chart FX for Java. ➤

Chart FX

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

Code in quality. Test out bugs.

Take control of your code through Automated Unit Testing and Code Analysis...



Jtest - AccountObject1 - Parasoft Jtest Enterprise

Package Explorer

```
Jtest Example
  dbc
  examples.inputs
  examples.jdbc
  examples.nbank
    AbstractTransaction.java
    Account.java
    Bank.java
    Customer.java
    DepositTransaction.java
    ITransaction.java
    WithdrawalTransaction.java
  images
  examples.bank
Jtest Example.jtest
  examples.jdbc
  examples.nbank
    AbstractTransactionTest.java
    AccountTest.java
    BankTest.java
    CustomerTest.java
    DepositTransactionTest.java
    ITransactionTest.java
    PackageTestCase.java
    WithdrawalTransactionTest.java
  jtest
```

DepositTransaction.java

```java
    }

    public boolean apply(Account account) {
        if (account.getCustomer().getName().equals("Mr. Ha
                && account.getBalance() / _amount == 100)
            account.setBalance(account.getBalance() + _amo
        else
            account.setBalance(account.getBalance() + _amo
        return true;
    }

    private int _amount;
```

DepositTransactionTest.java | Deposit.java

```java
     * @author Parasoft Jtest 7.0
     */
    public void testApply3() throws Throwable {
        Customer t1 = new Customer("Mr. Hacker", "000-0
        Account t0 = new Account(t1, 0);
        DepositTransaction THIS = new DepositTransactio
        // jtest_tested_method
        boolean RETVAL = THIS.apply(t0);
        // ArithmeticException thrown
        // at examples.nbank.DepositTransaction.apply$
        // at examples.nbank.DepositTransaction.apply(
        // jtest_unverified
    }

    /**
```

Coverage

```
Coverage of Tested Classes -- 99% [67/68 executable lines]
Jtest Example -- 99% [67/68 executable lines]
  examples.nbank.ITransaction -- [no executabl
  examples.nbank.Account -- 100% [22/22 exec
  examples.nbank.Bank -- 100% [11/11 executa
  examples.nbank.DepositTransaction -- 100% [
  examples.nbank.Customer -- 100% [10/10 exe
  examples.nbank.AbstractTransaction -- 100%
  examples.nbank.WithdrawalTransaction -- 90%
```

Jtest | Test Class Outline | Console | Object Repository | Suppression Messages

```
53 tasks
[3] Fix Coding Standards Violations
  [2] Java Database Connectivity
    [1] Close JDBC connections in "finally" blocks (JDBC.CDBC-1)
    [1] Reuse data sources for JDBC connections (JDBC.DSLV-2)
  [1] Possible Bugs
    [1] Ensure the overriding method name does not have a t
[4] Fix Unit Testing Problems
  [3] Assertion Failures
  [1] Contract Violations
[9] Review Unverified Unit Testing Exceptions
  [9] Unverified Runtime Exceptions
    [1] java.lang.ArithmeticException
      [1] DepositTransaction.java
        [Line 16] Unverified java.lang.ArithmeticException: / by zero
          at examples.nbank.DepositTransaction.apply$dbc$orig(De
          at examples.nbank.DepositTransaction.apply(DepositTran
          at examples.nbank.DepositTransactionTest.testApply3(De
```

PARASOFT®

Jtest®

- Automatically analyzes your code, applying 500+ Java coding best practices that surface poor code constructs affecting reliability, security and performance.
- Automatically unit tests your Java code evaluating code behavior and exposing unexpected exceptions, boundary condition errors and memory leaks.
- Generates extendable, high coverage test cases in JUnit format that can be quickly turned into libraries of reusable test assets.
- 100's of Quick Fix options for fast, accurate resolution of errors.
- Integrated code coverage analysis ensures that your code is thoroughly tested.
- Team collaboration support allows you to define, distribute and enforce code quality standards across entire development teams.

Stop bugs before they start:

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

The result? Cleaner, safer, more reliable and consistent code. Reduced code rework. Faster, more predictable release cycles. Reliable applications and happier, more productive end-users. **To try Jtest today, call 888-305-0041 (x-3501) or go to www.parasoft.com/JDJmagazine.**

Parasoft Jtest clients include: Lehman Bros., Cisco, Disney, Raytheon, ADP, Sabre Holdings, GE and more.

PARASOFT®

We make software work.™

Automated Error Prevention™