Please visit the Web sites of our advertising partners who make it possible for us to bring you this Digital Edition (PDF) of *JDJ*

ADVERTISER INDEX

ADVERTISER	URL	PH	PG
ADRENALINE	WWW.ADRENALINE.COM	877-890-9280	46
ADVANCED DATA, INC.	WWW.JVSEARCH.COM	301-528-0777	76
AIR2WEB	WWW.AIR2WEB.COM	404-815-7707	35
ALLAIRE	WWW.ALLAIRE.COM/DOWNLOAD	888-939-2545	27
BEA	DEVELOPER.BEA.COM	408-570-8000	29
BORLAND	WWW.BORLAND.COM	800-252-5547	37
CALDERA	WWW.CALDERA.COM		25
CAREER OPPORTUNITY ADVERTISERS		800-846-7591	109-117
CEREBELLUM SOFTWARE	WWW.CEREBELLUMSOFT.COM	888-862-9898	103
CIMMETRY SYSTEMS	WWW.CIMMETRY.COM	800-361-1904	38
COMPOZE SOFTWARE	WWW.COMPOZE.COM	866-COMPOZE	31
COMPUTERWORK.COM	WWW.COMPUTERWORK.COM	800-691-8413	66
COMPUWARE	WWW.COMPUWARE.COM/NUMEGA	800-4-NUMEGA	33
CORDA TECHNOLOGIES	WWW.POPCHART.COM	801-802-0800	55
CYSIVE	WWW.CYSIVE.COM/CAREERS	800-996-2582	93
ELIXIR TECHNOLOGY	WWW.ELIXIRTECH.COM/DOWNLOAD	65 532-4300	71
ESMERTEC	WWW.ESMERTEC.COM	877-751-3420	39
FIORANO	WWW.FIORANO.COM	800-663-3621	85
FLASHLINE.COM, INC.	WWW.FLASHLINE.COM	800-259-1961	61
HIT SOFTWARE	WWW.HITSW.COM	408-345-4001	59
IAM CONSULTING	WWW.IAMX.COM	212-580-2700	79
INETSOFT TECHNOLOGY CORP	WWW.INETSOFTCORP.COM	908-755-0200	41
INSTANTIATIONS, INC.	WWW.INSTANTIATIONS.COM	800-808-3737	77
INT	WWW.INT.COM	713-975-7434	52
INTERNET WORLD WIRELESS	WWW.PENTONEVENTS.COM		87
INTUITIVE SYSTEMS, INC.	WWW.OPTIMIZEIT.COM		83
METROWERKS	WWW.METROWERKS.COM	800-377-5416	63
NO MAGIC	WWW.MAGICDRAW.COM	303-914-8074	9
NORTHWOODS SOFTWARE CORPORATION	WWW.NWOODS.COM	800-226-4662	86
NUSPHERE	WWW.NUSPHERE.COM	781-280-4600	73
OOP.COM	WWW.OOP.COM	877-667-6070	51
PACIFIC NORTHWEST NATIONAL LABORATORY	WWW.PNNL-SOFTWARE.COM	509-946-1110	86
PARASOFT	WWW.PARASOFT.COM/JDJ_DEC.HTM	888-305-0041	53
PRAMATI	WWW.PRAMATI.COM	408-965-5513	75
PREEMPTIVE SOLUTIONS	WWW.PREEMPTIVE.COM	800-996-4556	21
PROGRAMIX	WWW.JTHREADKIT.COM		70
PROGRAMMER'S PARADISE	PROGRAMMERSPARADISE.COM	800-445-7899	8
PROGRESS SOFTWARE	WWW.SONICMQ.COM/AD1200.HTM	800-989-3773	2
PROSYST	WWW.PROSYST.COM	678-366-4607	43
PROTOVIEW	WWW.PROTOVIEW.COM	800-231-8588	3,95
QUADBASE	WWW.QUADBASE.COM	800-231-8588	99
QUICKSTREAM SOFTWARE	WWW.QUICKSTREAM.COM	888-769-9898	97
RADVIEW	WWW.RADVIEW.COM	888-RADVIEW	23
RATIONAL SOFTWARE	WWW.RATIONAL.COM/JDJ		17
SEGUE SOFTWARE	WWW.SEGUE.COM	800-287-1329	4
SITRAKA SOFTWARE	WWW.SITRAKA.COM	800-361-3264	91
SITRAKA SOFTWARE	WWW.SITRAKA.COM/DEPLOY	888-361-3264	19
SITRAKA SOFTWARE	WWW.SITRAKA.COM/COMPONENTS	888-361-3264	57
SITRAKA SOFTWARE	WWW.SITRAKA.COM/PERFORMANCE	888-361-3264	120
SLANGSOFT	WWW.SLANGSOFT.COM	972-2-648-2424	119
SOFTWARE AG	WWW.SOFTWAREAG.COM/TAMINO	925-472-4900	67
SOFTWIRED	WWW.SOFTWIRED-INC.COM	41-14452370	81
SYS-CON MEDIA CAREER CENTER	HTTP://CAREERS.SYS-CON.COM		89
THESHORTLIST	WWW.THESHORTLIST.COM	877-482-8827	32,101
THOUGHTWORKS	WWW.THOUGHTWORKS.COM	888-THOUGHT	13
TIDESTONE TECHNOLOGIES	WWW.TIDESTONE.COM	800-884-8665	47
TOGETHERSOFT CORPORATION	WWW.TOGETHERSOFT.COM	919-833-5550	6
UNIFY CORPORATION	WWW.UNIFYEWAVE.COM	800-GO UNIFY	69
VIRTUALSCAPE	WWW.VIRTUALSCAPE.COM	877-VSCAPE4	65
VISUALIZE INC.	WWW.VISUALIZEINC.COM/JDJ	602-861-0999	97
WEBGAIN	WWW.WEBGAIN.COM	408-517-3815	15, 45
WIRELESS BUSINESS & TECHNOLOGY	WWW.WIRELESS-MAGAZINE.COM	201-802-3020	105
XML DEVCON EUROPE	WWW.XMLDEVCONEUROPE2001.COM		107
ZERO G SOFTWARE	WWW.ZEROG.COM	415-512-7771	49

LINUX

Presented by

JAVA DEVELOPER'S JOURNAL

The World's Leading Java Resource

Volume: 5 Issue: 12

FOCUS ISSUE

JAVADEVELOPERSJOURNAL.COM

SYS-CON MEDIA

Cross-platform apps compiled on Linux work right out of the box
JAVA TECHNOLOGY ON THE LINUX PLATFORM
A Guide to Getting Started
by Calvin Austin page 10

SEAN RHODY, EDITOR-IN-CHIEF

Why Linux?

Linux is one of the operating systems I run in my basement in what my wife likes to call the computer lab from hell. That may be because the heat from all the obsolete equipment I collect is reducing my fuel bill and keeping the place toasty and warm. Linux is one of three UNIX variant operating systems I run down there (four if you count Solaris for Intel as a separate beast), and it's probably the one I tinker with the most. Every time I upgrade a machine at home I can't resist the temptation to throw Linux on it to see if it will run a little better, a little faster. I've tried a number of different commercial distributions and I like aspects of each. I like the Enlightenment manager – it's got pizzazz. But it's more than just that.

When I was studying computer science in college I had the opportunity to tinker a little with operating systems. It's one of the most intriguing aspects of computer science – providing a platform where other code can run. In some small way, when people do architectural designs that allow for reuse, they're providing the same kind of functionality. Nothing compares to programming the bare metal.

Still, the joy of programming doesn't really matter when it comes to the business world. Business is about products, solutions, and services. It's about adding value. That's one reason Linux is important – it adds value.

One of the challenges of the industry is getting the most performance and value out of hardware investments. Server-class hardware is expensive. Add to that the licensing costs for some of the most popular operating systems and you have a hefty budget. Throw in the additional costs of running an environment that may have multiple disparate operating systems and you have a real headache on your hands.

Linux can help. As an operating system it runs well on a wide variety of hardware. In my basement lab I've run it on 486s and old Pentiums with good results. Unlike Windows, Linux seems more bound by memory than by processor power. It also runs well on most hardware platforms that run UNIX, such as DEC Alphas or SunSPARC machines. So it's possible to run one operating system in an environment that features a mix of hardware. That alone is an accomplishment.

That it's free and open source makes it even more appealing. The free part is easy to understand. It's the open source part that really drives the popularity and adoption. There's a large community of people developing for Linux, both the internals of the operating system and the software. The platform is being held hostage by a particular vendor who's also forcing hardware vendors to install it as the sole operating system. It's free, and you can change what you don't like in the system, assuming you're good at OS internals.

Linux has also evolved to the point where a large number of industry heavyweights have decided to embrace it. Oracle runs on it and is focusing on it as a key platform for their database. Last time I looked Sybase was providing Adaptive Server for the platform at no cost. IBM has ported it to a number of its hardware platforms and is one of a growing number of companies providing 24/7 support for the product. Inprise JBuilder runs on it. The list keeps growing.

Probably the most interesting aspect from our standpoint is that it makes a good development environment for Java. It runs well on most hardware, there's plenty of support for it, and it doesn't cost anything (in software terms) to get all the tools you need for development. In this issue of ***JDJ*** we've highlighted a number of topics around Linux to try to give you a glimpse of what life is like as an open source developer. Instead of "Why Linux?" it's time to ask "Why not?"

sean@sys-con.com

AUTHOR BIO

Sean Rhody is editor-in-chief of Java Developer's Journal. *He is also a respected industry expert and a consultant with a leading Internet service company.*

EDITOR-IN-CHIEF: SEAN RHODY
EXECUTIVE EDITOR: M'LOU PINKHAM
ART DIRECTOR: ALEX BOTERO
MANAGING EDITOR: CHERYL VAN SISE
EDITOR/COPY CHIEF: NANCY VALENTINE
ASSOCIATE EDITOR: BETTY LETIZIA
ASSOCIATE EDITOR: JAMIE MATUSOW
EDITORIAL INTERN: SUZANNE AUGELLO
EDITORIAL CONSULTANT: SCOTT DAVISON
TECHNICAL EDITOR: BAHADIR KARUV
PRODUCT REVIEW EDITOR: ED ZEBROWSKI
INDUSTRY NEWS EDITOR: ALAN WILLIAMSON
E-COMMERCE EDITOR: AJIT SAGAR

WRITERS IN THIS ISSUE

GABE ALLAN, JOE ANTHONY, CALVIN AUSTIN, BRIAN R. BARBASH, ROBERT J. BRUNNER, MARCEL GAGNÉ, DAVID JOHNSON, ANU MANNAR, MARK MATTHEWS, JOE MITCHKO, CERI MORÁN, MAUREEN O'GARA, BEN OKOPNIK, STÉPHANIE PARKIN, SEAN RHODY, PAUL TINDALL, JASON WESTRA, ALAN WILLIAMSON, MURRAY WILSON, STUART ZIPPER

SUBSCRIPTIONS

FOR SUBSCRIPTIONS AND REQUESTS FOR BULK ORDERS, PLEASE SEND YOUR LETTERS TO SUBSCRIPTION DEPARTMENT

SUBSCRIPTION HOTLINE: 800 513-7111

COVER PRICE: $4.99/ISSUE
DOMESTIC: $49/YR. (12 ISSUES) CANADA/MEXICO: $69/YR.
OVERSEAS: BASIC SUBSCRIPTION PRICE PLUS AIRMAIL POSTAGE (U.S. BANKS OR MONEY ORDERS). BACK ISSUES: $12 EACH

PUBLISHER, PRESIDENT AND CEO: FUAT A. KIRCAALI
VP, PRODUCTION: JIM MORGAN
SENIOR VP, SALES & MARKETING: CARMEN GONZALEZ
VP, SALES & MARKETING: MILES SILVERMAN
ASSISTANT CONTROLLER: JUDITH CALNAN
CREDIT & COLLECTIONS: CYNTHIA OBIDZINSKI
ACCOUNTS PAYABLE: JOAN LAROSE
ADVERTISING ACCOUNT MANAGERS: ROBYN FORMA
MEGAN RING
ASSOCIATE SALES MANAGERS: CARRIE GEBERT
CHRISTINE RUSSELL
SALES ASSISTANT: ALISA CATILANO
ASSOCIATE ART DIRECTOR: DINA ROMANO
ASSISTANT ART DIRECTORS: CATHRYN BURAK
LOUIS F. CUFFARI
GRAPHIC DESIGNER: ABRAHAM ADDO
GRAPHIC DESIGN INTERN: AARATHI VENKATARAMAN
WEBMASTER: ROBERT DIAMOND
WEB DESIGNERS: GINA ALAYYAN
STEPHEN KILMURRAY
SYS-CON EVENTS MANAGER: ANTHONY D. SPITZER

EDITORIAL OFFICES

SYS-CON MEDIA, INC., 135 CHESTNUT RIDGE RD., MONTVALE, NJ 07645
TELEPHONE: 201 802-3000 FAX: 201 782-9600
SUBSCRIBE@SYS-CON.COM

JAVA DEVELOPER'S JOURNAL (ISSN#1087-6944) is published monthly (12 times a year) for $49.00 by SYS-CON Publications, Inc., 135 Chestnut Ridge Road, Montvale, NJ 07645. Periodicals Postage rates are paid at Montvale, NJ 07645 and additional mailing offices. POSTMASTER: Send address changes to: JAVA DEVELOPER'S JOURNAL, SYS-CON Publications, Inc., 135 Chestnut Ridge Road, Montvale, NJ 07645.

WORLDWIDE DISTRIBUTION BY

CURTIS CIRCULATION COMPANY
730 RIVER ROAD, NEW MILFORD NJ 07646-3048 PHONE: 201 634-7400

Written by Maureen O'Gara

Java and Creeping Open Source

Sun, in case you don't know – and there's no reason you should – has quietly gone and gotten one of its many licenses (Sun has licenses the way Imelda Marcos has shoes) admitted to the tiny pantheon of official open source licenses worshiped by the Linux community.

Taking no chances, it got two ministers to officiate.

It signed up the Open Source Initiative (OSI), the entity presided over by open source theologian Eric Raymond, the guy who got his hands on Microsoft's Linux-fretting Halloween Papers two years ago and published them, and then Sun went and got the backing of the Free Software Foundation, the originator of the famous open source–defining General Public License (GPL), the veritable Petronius Arbiter of this kind of stuff.

The Free Software Foundation evidently needed a bit of persuasion. So Sun whistled up Brian Behlendorf, dean of the Apache Software Foundation and head of Collab.Net, the let's-get-open-source-developers-some-paying-work venture in which Sun is conveniently an investor, to persuade the great god of GNU and GPL, Richard Stallman (basically the guy who invented open source), that a specification is as good as if not better than putting out code. You see, the license that Sun wanted canonized, a thing called the Sun Industry Standards Source License or SISSL, says you don't have to hang your proprietary code out like Monday's wash. You can supply a specification and reference implementation of any deviations instead. Sun claims this is often more understandable than the raw code.

Anyway, SISSL, which is pronounced *Sizzle*, is supposedly good for commercial products because it lets developers keep their actual code, especially kernel-level code, proprietary while using open source code to bootstrap their implementations of standards-based technologies.

Now why, you may ask, has Sun, which is a distinctly proprietary company, gone to all this trouble? It's because Sun, which has previously held Java in a studied death grip, is suddenly toying with the idea of open sourcing it. (Really, talk about manic-depressive.)

If one associates Linux with open source (and it's impossible to separate Linux from its theological apparatus), then the upstart operating system that Sun secretly wishes would fall into the same gaping crevice as Microsoft is having as great an impact on the burgeoning monolith as Sun and Java have had on the popular imagination. Sun is wrestling with the idea of open sourcing Java to appear a thought leader, even though in reality it will be a thought follower, and needs to pull off the caper in such a way that its handsome market cap (now nose to nose with, if not a nose ahead of, the mighty IBM's) isn't damaged. Java may have contributed little to Sun's coffers, but it's certainly made Sun's stockholders rich, and you know how skittish Wall Street is.

Anyway, SISSL is pretty much a GPL, except that under SISSL the source code has to be maintained by a standards body and, since there's no way Sun is going to let any outside body get its hands on it – one ECMA catastrophe is enough, thank you (SISSL gives Sun license, so to speak, to keep Java under its thumb) – SISSL is secretly the model Sun says it's looking for to make Java open source. It'll be the rationale to turn the notorious Java Community Process (JCP), which knowledgeable insiders assure us remains very much a creature of Sun for all its vaunted liberalization recently, into a standards organization, and voilà, control is extended and purified.

Sun says it's been kicking around the idea of open sourcing Java for years, and what with internal resistance to the idea ebbing somewhat (even if selling the idea inside is still its biggest hurdle), George Paolini, vice president of the JCP, went and made a speech at ApacheCon in London in October and used the words *Java* and *open source* in the same sentence, something that's never been done in public before.

Sun says we can take Paolini's trial balloon as a "commitment" even though Sun has no idea when it could do such a thing or exactly what Java we're talking about here: Standard, EE, or the whole magilla. There also needs to be a definitive ruling on IP. Well, IBM did develop an awful lot of it, ya know, and even if it was all supposed to be turned over to Sun or something, lawyers' opinions can vary with the tides and it was supposedly all that code that somebody else developed that stopped Sun in its tracks from making its Solaris operating system open.

It's also unclear how the major Java licensees such as BEA Systems might take to the gambit. These guys are supposed to have a lot of say now about the direction of the Java roadmap and make contributions to it. BEA's whole business depends on Java so its reaction should be interesting. Sun says it's never explicitly discussed the subject with the licensees, and notes that the terms of their licenses are all different so they might react to the IP issue differently. Paolini sent the key licensees a warning note when he got back from London about what he said and Sun says so far they haven't reacted.

As far as *when* goes, Sun's biggest concern is with what it calls *critical mass*, a term it can't exactly define, but it says that, whatever it is, Java has to get to it before it can go open source. That way it can't be forked, and "compatibility," an idea precious to the master of Java, will be retained. You see, if that many people are using it, then compatibility would *have* to be maintained. Sun has repeatedly used the compatibility issue to preserve its sometimes-suffocating stewardship over Java.

Java is currently licensed under the Sun Community Source License (SCSL), which is no friend of Linux developers and has certainly short-circuited clean-room Java-on-Linux projects as well as given open source developers attitudes about Sun that are similar to the esteem in which they hold Microsoft.

Sun sees open sourcing Java as neutralizing some of the flak it's taken from Microsoft and Java partner IBM about how it's exercised its stewardship. Come to think of it, Microsoft has mentioned the words *C#* and *open source* in the same sentence in its discussions with ECMA, the standards body that will be standardizing the new Java-leveling programming language (and would have been standardizing Java instead if Sun hadn't bolted). Sun may ultimately need to open source Java to guarantee ubiquity in the face of onslaughts from the Evil Empire.

ogara@g2news.com

Author Bio

Maureen O'Gara is the editor of Client Server News, *a weekly subscription-based newsletter focused on Microsoft and Windows NT,* The Online Reporter, *a weekly subscription-based newsletter on e-business, and LinuxGram.com, a free Web site dedicated to breaking Linux news.*

WRITTEN BY ALAN WILLIAMSON

Dukey, Meet Tux

Unless you've been stranded on a remote government testing station in outermost Mongolia, you'll know that the new kid on the block is sticking around and making some serious inroads on corporate America. I am, of course, talking of Linux, and with that, welcome to the Linux Focus Issue of ***JDJ***.

We've pulled together a number of articles that offer you a window into the world of Linux and show how you, as a Java developer, can utilize this new tool emerging on the horizon. For a number of years now Linux was not given the respect it deserved. Many large blue chips dismissed it as a hacker's toy and remained loyal to the Oracles and Suns of the world. These same blue chips are now changing their tune somewhat and beginning to explore this new contender for their IT budget. The irony of it is that their budget isn't being as hammered as it was, and I believe this is where the problem lies.

You know the old saying: "You get what you pay for." I believe this is Linux's biggest hurdle. Not that it isn't up for the job or that there isn't any standard hardware platform. It's because it's free. It goes against all known conventions and this is making the CTOs a little nervous.

We'll take you into the world of Linux through the eyes of Java developers. We have some great articles on some of the more frequently used Linux tools. With respect to the task of getting Java onto Linux and how best to configure it, the best person to ask about that sort of thing has to be the lead staff engineer at Sun who's responsible for getting Java to Linux. Calvin Austin works alongside the Blackdown.org team, which in turn was responsible for the original JDK port to Linux, to get Java 2 to Linux. Calvin has written a clear account of installing Java to Linux and the meaning behind many of the terms that Linux newbies are faced with.

As regular ***JDJ*** readers know, I've always advocated MySQL as a great alternative to other costly database servers. I'm honored to announce that we've collared Mark Matthews for a great piece on using MySQL from Java. Mark Matthews is the man responsible for allowing us access to the power of MySQL through his very popular JDBC driver. Linux is usually deployed as a server, and to this end we have an excellent piece from Murray Wilson on using the infamous Apache/JServ combination to deploy Java Servlets. Murray talks us through the installation and deployment woes you may experience with this strong application server.

If you're going to throw down your MS Windows and move to a Linux desktop, then allow Ben Okopnik to talk you through some of the hurdles you may face. Ben runs through some of the procedures you need to do to make your desktop Java development–friendly.

Linux is penetrating many of the large software vendors and we're now seeing the likes of Oracle and even Sun offering Linux alternatives to their main flagship products. Ceri Moran has put together an excellent article detailing many of these offerings. It may surprise you to learn just who is doing what. Linux isn't a fad, nor is it a flash-in-the-pan technology. It's here to stay, and, given that, learning a little more about what it may be able to do for you is never a bad thing. Linux does suffer from the Apple-devotee syndrome. That is, you do get the odd Linux fanatic that can put you off the whole experience. These people are best avoided and kept in the dark server room where they belong. They're happy there.

Don't take anyone's word for it; discover it for yourself. It's free, after all, so the only thing you're really going to lose is time. So, please enjoy what we've pulled together for you here in this special focus issue.

alan@n-ary.com

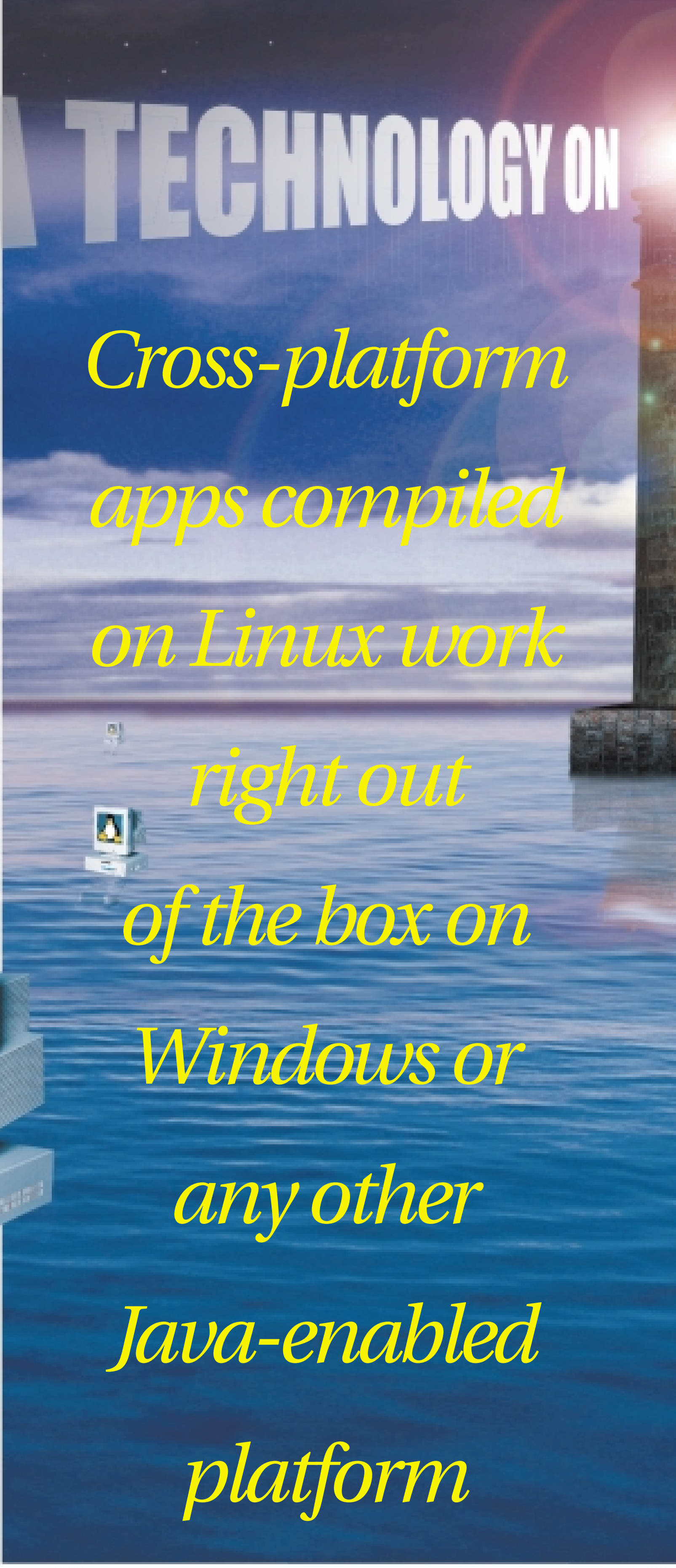
AUTHOR BIO

Alan Williamson is CEO of the first pure Java company in the UK, n-ary (consulting) Ltd (www.n-ary.com), a Java solutions company specializing in delivering real-world applications with real-world Java. Alan has authored two Java servlet books and contributed to the servlet API.

FOCUS

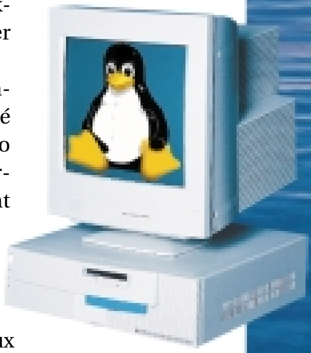
JAVA TECHNOLOGY ON LINUX

WRITTEN BY CALVIN AUSTIN


Cross-platform apps compiled on Linux work right out of the box on Windows or any other Java-enabled platform

The Java 2 Platform, Standard Edition (J2SE technology) v1.3 for Linux means that Linux users and developers can take advantage of thousands of Java technology–based applications, from enterprise e-commerce infrastructure to client-side applications. It also opens up a huge emerging market for companies that already develop Java products.

The Java 2 Platform port was developed with the assistance of the Blackdown.org porting group. Although Linux is a UNIX-based operating system, it's evolved at a different pace and direction than other UNIX platforms, making a port of the Java platform a sizable challenge. Many important bug fixes have gone into the Linux operating system and associated GNU libraries as a result of the work initiated by Blackdown, not just on Intel-based distributions but also Power PC and SPARC platforms. In particular, the LinuxThreads library has improved dramatically over the past year on multiprocessor machines.



In an ideal world, the Java 2 Platform, combined with Java development tools such as Forté Community Edition, should be all you need to develop your projects. However, if the Linux operating system is a new development environment for you and you've previously developed with the Java 2 Platform for Windows or Solaris, you might find Linux a little different. This article provides tips to bring you up to speed as quickly as possible developing on the Linux platform.

SDK and JRE Installation Tips

Installation is straightforward, but be aware that the Java Runtime Environment (JRE) and Java Software Developer Toolkit (SDK) Red Hat Package Management (RPM) files are installed by default to /usr/java. If you want these files in a different location, install with the RPM command:

```
rpm -i  --badreloc --relocate /usr/java/=/usr/local/home j2sdk-
1_3_0-linux.rpm
```

After relocating the files as shown above, start developing by adding the bin directory from the SDK or JRE installation to your $PATH. For example, using the bash shell, the $PATH is updated:

```
export PATH=/usr/java/jdk1.3/bin:$PATH
```

THE LINUX PLATFORM

A GUIDE TO GETTING STARTED

To verify that you've configured your environment, run the Java -version command, and you should see a version string confirming that you've installed the Java 2 Runtime Environment:

```
java -version
java version "1.3.0"
Java(TM) 2 Runtime Environment, Standard Edition (build 1.3.0)
Java HotSpot(TM) Client VM (build 1.3.0, mixed mode)
```

Java Plug-In Installation Tips

The J2SE plug-in is embedded inside the Java Runtime directory. As long as the Netscape browser can find the plug-in library from within the Java Runtime directory, it can also find the additional runtime files it needs.

To let your Netscape browser know where the Java plug-in library is, set the NPX_PLUGIN_PATH environment variable to point to the directory where the javaplugin.so shared library is located:

```
export NPX_PLUGIN_PATH=/usr/java/jdk1.3/jre/plugin/i386
```

To configure the Java plug-in properties, use the ControlPanel program located in the jre/bin directory. In the example above, the program is /usr/java/jdk1.3/jre/bin/ControlPanel.

Java HotSpot Virtual Machine

This is the first release on Linux that includes the Java HotSpot Virtual Machine. Both the client and server Java HotSpot compilers are included by default in the Java Runtime Environment. These two solutions share the unique Java HotSpot runtime environment, but have different compilers optimized for the specific performance requirements of client- and server-side-based applications. The bundling of these virtual machines in one package provides developers with increased flexibility and convenience in deploying their client- or server-based Java technology applications.

By default the client compiler is enabled, but for intense server-side applications, the server compiler can be run using the -server runtime option. The Java HotSpot Virtual Machine normally runs in a mixed mode, as seen in the version output. This means HotSpot will dynamically compile Java bytecodes into native code when a number of criteria have been met, including how many times the method has been run through the interpreter. The mixed runtime mode normally results in the best performance.

About Linux Threads

One major difference between developing on Linux and other UNIX operating systems is the system threads library. In Java 2 releases prior to 1.3, the JVM uses its own threads library, known as GreenThreads, to implement threads in the Java platform. The GreenThreads implementation minimizes the JVM's exposure to differences in the LinuxThreads library and makes the port easier to complete. The downside to GreenThreads is that system threads on Linux are not taken advantage of, so the JVM doesn't scale well when additional CPUs are added.

In J2SE Release 1.3, the HotSpot Virtual Machine uses Linux system threads to implement JavaThreads. Because LinuxThreads are implemented as a cloned process, each JavaThread shows up in the process table if you run the ps command. Although this may look different, it's normal behavior for threaded programs on Linux:

```
java -jar Notepad.jar
ps -eo pid,ppid,command
```

In Table 1, the process ID 11712 is the invoked JVM. The other processes with ID 11712 as the parent process (listed under the PPID column) are JavaThreads implemented by the Linux system threads library. Each LinuxThread is created as a process clone operation that leaves the task of scheduling threads to the process scheduler.

PID	PPID	COMMAND		
11667	28367	/usr/java/jdk1.3/bin/i386/native_threads/java	-jar	Notepad.jar
11712	11667	/usr/java/jdk1.3/bin/i386/native_threads/java	-jar	Notepad.jar
11713	11712	/usr/java/jdk1.3/bin/i386/native_threads/java	-jar	Notepad.jar
11714	11712	/usr/java/jdk1.3/bin/i386/native_threads/java	-jar	Notepad.jar
11715	11712	/usr/java/jdk1.3/bin/i386/native_threads/java	-jar	Notepad.jar
11716	11712	/usr/java/jdk1.3/bin/i386/native_threads/java	-jar	Notepad.jar
11717	11712	/usr/java/jdk1.3/bin/i386/native_threads/java	-jar	Notepad.jar
11718	11712	/usr/java/jdk1.3/bin/i386/native_threads/java	-jar	Notepad.jar
11722	11712	/usr/java/jdk1.3/bin/i386/native_threads/java	-jar	Notepad.jar
11723	11712	/usr/java/jdk1.3/bin/i386/native_threads/java	-jar	Notepad.jar
11724	11712	/usr/java/jdk1.3/bin/i386/native_threads/java	-jar	Notepad.jar
11726	11712	/usr/java/jdk1.3/bin/i386/native_threads/java	-jar	Notepad.jar

TABLE 1 Java threads

On Solaris, however, JavaThreads are mapped onto user threads, which in turn are run on lightweight processes (LWP). On Windows the threads are created inside the process itself. Today, creating a large number of JavaThreads on Solaris and Windows is faster than on Linux. Thus, you might need to adjust programs that rely on platform-specific timing to take a little longer on startup when they run on Linux.

Linux Debugging Tools

As you learned in the section on threads, LinuxThreads are implemented as Linux processes. This makes debugging Java programs running on Linux a little tricky, but not impossible. This section explains how to attach the GDB GNU debugger to the JVM, and how to start the JVM with GDB to run debugging commands.

Using GDB

Going back to the example in the threads section, if the JVM is already running, you can attach the system GDB tool as follows:

```
gdb /usr/java/jdk1.3/bin/i386/native_threads/java 11712
```

The output from using the GDB tool should look similar to Listing 1. At this point the GDB tool has connected to the running Java process and is waiting for user input.

With GDB attached to the JVM, you can run any GDB commands. If you've used GDB before, these commands will look familiar. The info threads command lists the LinuxThreads, the t 12 command selects thread number 12 as the current thread, and the where command lists the stack frames in that thread (thread number 12, in this example). The output from an example session is shown in Listing 2.

Starting a Java Virtual Machine Using GDB

Attaching a GDB trace to an existing program is usually the easiest way to debug a deadlock JVM; however, you might want to start the JVM from within the GDB debug tool from the outset, especially if you suspect the failure happens shortly after your application starts.

To enable the JVM to run from inside GDB on Linux, the following setup needs to be in place:

1. Set the APPHOME and LD_LIBRARY_PATH environment variables before you run the GDB command. This step is required because the Java program is a shell wrapper that configures the environment for the real Java program to run, and GDB needs to run with the real binary Java program, not the shell wrapper version.
2. Instruct GDB to ignore the signals that the HotSpot JVM uses to maintain its own state. An example of such a signal is SIGUSR1.

These next lines start the Java 2D demo inside a GDB session. First, the LD_LIBRARY_PATH and APPHOME environment variables are set. In this example, the setup assumes the SDK is installed in the default /usr/java directory:

```
export APPHOME=/usr/java/jdk1.3
export
LD_LIBRARY_PATH=/usr/java/jdk1.3/jre/lib/i386/native_threads:/usr/
java/jdk1.3/jre/lib/i386:/usr/java/jdk1.3/jre/lib/i386/hotspot
```

3. Start GDB. The following GDB commands will load the JVM, instruct GDB to stop at the main method, and ignore the signals used by the HotSpot Virtual Machine. The commands can be entered either after GDB has started, or put in a .gdbinit file in your home directory. Once the GDB tool has reached the breakpoint, other breakpoints can be added, or just enter the GDB cont command to continue:

```
file /usr/java/jdk1.3/jre/lib/i386/native_threads/java
break main
run -jar Java2Demo.jar
handle SIGUSR1 nostop noprint pass
handle SIGSEGV nostop noprint pass
handle SIGILL  nostop noprint pass
handle SIGQUIT nostop noprint pass
```

Generating a Java Stack Trace on Linux

As LinuxThreads are implemented as Linux processes, this makes debugging Java programs running on Linux a little different. To generate a stack trace from the terminal window that started your application, type the sequence <CTRL>/ to send the QUIT signal to generate a stack trace. However, if you need to generate a stack trace from a Java application already running in the background, send the QUIT or signal number 3 to the launcher process ID. The launcher process in this example is process ID 11667. The stack trace can be generated by sending the signal via the kill command as follows:

```
kill -3 11667
```

The resulting stack trace is a snapshot of the JavaThreads and details the state of each thread. In each thread trace, a value called nid – the hex number of the cloned process it came from – can be used to track down deadlocks or busy sections in your application. For more details on analyzing Java stack traces, refer to the "Debugging Applets, Applications, and Servlets" chapter in *Advanced Programming for the Java 2 Platform* (Addison-Wesley, 2000).

Integrating Native JNI Code on Linux

If you've used native JNI code in your Java application, it probably needs to be recompiled on Linux. Linux distributions come complete with a wealth of GNU tools for compiling C, C++, Fortran, and other languages. Some of the system header files and system calls may be named slightly differently on Linux, and the size of structures or global variables may be smaller than on other UNIX platforms. A quick run-through of the header files can avoid complex issues later on.

To recompile your native C or C++ code and generate the native library file, use the GNU tool gcc for C programs and g++ for C++ programs, and supply the options as shown in the next example. Finally, set the LD_LIBRARY_PATH environment variable to point to the directory where the final native library is located. In the following examples the native library is called libnativelib.so and is loaded from within a Java program by the using System.loadLibrary("nativelib").

GNU C/Linux

```
gcc -o libnativelib.so -D_REENTRANT -shared -Wl,-
soname,libnative.so -I/usr/java/jdk1.3/include -
I/usr/java/jdk1.3/include/linux nativelib.c -static -lc
```

GNU C++/Linux

```
g++ -o libnativelib.so -D_REENTRANT -shared -Wl,-soname,libdbmap.so
-I/usr/java/jdk1.3/include -I/usr/java/jdk1.3/include/linux
nativelib.cc -static -lc
```

Desktop Differences (Copy and Paste)

If you've been using Windows or a keyboard with a copy and paste key, you may be wondering how to copy and paste text between Java programs and other desktop programs and terminals.

Linux uses a mouse-driven copy-and-paste mechanism in which mouse button one selects and copies text, and mouse button two pastes the text. This technique works for Abstract Window Toolkit (AWT) components because they use the primary selection to achieve copy and paste. Project Swing components, however, use the system clipboard for copy and paste, and most tools on the desktop, apart from the Netscape browser, don't use the clipboard.

A workaround to this limitation is to map a key or mouse button to access the system clipboard:

```
*VT100.Translations: #override \
            <Btn3Up>:               select-end(CLIPBOARD) \n\
            <Btn2Up>:               insert-selection(CLIPBOARD) \n
```

The lines above can be passed as a value to an X tool using the -xrm option. Alternatively, the mapping can be made accessible to the entire desktop by including it in the .Xdefaults file in the user's home directory. The command xrdb -merge $HOME/.Xdefaults will reload updates in the .Xdefaults file.

Conclusion

Using Linux to develop and deploy applications written in Java has the same benefits as developing on any Java platform. Thousands of Java applications are available, and because they're cross-platform, those compiled on Linux will work right out of the box on Windows or any other Java-enabled platform.

In addition, the amount of Linux knowledge needed to develop or deploy Java applications is relatively small, making Linux an attractive choice as a development platform.

Resources

Java technologies on linux: http://java.sun.com/linux
Blackdown Java porting team: http://blackdown.org

AUTHOR BIO

Calvin Austin is the lead staff engineer for the Java 2 Standard Edition Linux project at Sun Microsystems, Inc., and works with the Blackdown.org Java porting group. He's cofounder of the Java Developer Connection and coauthor of Advanced Programming for the Java 2 Platform *(Addison-Wesley, 2000).*

calvin.austin@eng.sun.com

Blackdown Develops Java for Linux

WRITTEN BY JUERGEN KREILEDER, LEAD ENGINEER

Blackdown is a small international community of volunteer developers dedicated to the professional development of the Java platform for Linux. Blackdown started developing Java technology for Linux at the time of the Java Software Development Kit 1.0.2 release, years before commercial vendors even thought about supporting Linux. The project began because we felt the combination of the Java programming language and Linux would make a powerful platform for development and deployment. This feeling and the lack of commercial support provided the right ingredients to get the project started.

The noncommercial nature of Blackdown reduces the impact of commercial constraints that other Java technology development groups face, freeing Blackdown to focus only on the technology and bug-fixes. In contrast to commercial vendors like Sun or IBM, Blackdown doesn't limit its Java support to Linux on the Intel architecture. It also supports other popular architectures that run Linux, like Power PC, SPARC, S/390, ARM, and Motorola 68k. Our primary goal is to provide the best Java solution for Linux regardless of the architecture. The fact that Blackdown isn't a commercial entity doesn't mean that we don't provide support for our products. On the contrary, we treat every bug report or problem seriously, regardless of whether it was reported by a company, a single developer, or an academic user.

Our work has been made possible by Sun, which has granted Blackdown a special noncommercial license. While our relationship with Sun hasn't always been easy, it's getting better and better since early 2000. We're now working more closely with Sun than ever before.

Since Blackdown isn't formally funded, all work is done by volunteers, so we're always looking for qualified new team members.

By codeveloping Java for Linux, Blackdown's activities should be viewed as a complement to Sun's. Blackdown offers an experienced developer group working on enhancing and tuning Java for Linux while remaining vendor-independent.

In addition to the Java Platform Standard Edition, Blackdown provides Linux versions of essential optional packages for the Java 2 platform.

- Java 3D 1.2 gives developers high-level constructs for creating and manipulating 3D geometry, and tools for constructing the structures to render that geometry. This enables you to write stand-alone 3D graphics applications or Web-based 3D applets. It's currently available for Linux on Intel, PowerPC, and SPARC.
- The Java Advanced Imaging API 1.0.2 is available for Linux on Intel and SPARC. The JAI API extends the Java 2 platform by allowing sophisticated, high-performance image processing to be incorporated into Java applets and applications.
- The Java Media Framework 2.1 was tuned for use on Linux in a cooperative effort with Sun. The Java Media Framework is an API for incorporating audio, video, and other time-based media into Java applications and applets.

Blackdown will continue to collaborate closely with Sun to provide further versions of the Java 2 platform and optional packages for Linux. We're also working with Caldera, which joined the Java-Linux community recently and represents the Linux/Open Source world in the Java Community Process program.

Listing 1

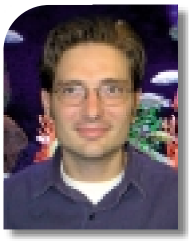
```
[root@ayli bin]# gdb
/usr/java/jdk1.3/bin/i386/native_threads/java 11712
GNU gdb 4.18
Copyright 1998 Free Software Foundation, Inc.
 GDB is free software, covered by the GNU General Public
License, and you're welcome to change and/or distribute copies
of it under certain conditions. Type "show copying" to see the
conditions. There's absolutely no warranty for GDB. Type "show
warranty" for details. This GDB was configured as "i386-red-
hat-linux"...

/usr/X11R6/bin/11712: No such file or directory.
Attaching to program:
/usr/java/jdk1.3/bin/i386/native_threads/java, Pid 11712
Reading symbols from /lib/libpthread.so.0...done.
Reading symbols from
/usr/java/jdk1.3/jre/lib/i386/native_threads/libhpi.so...
done.
Reading symbols from /usr/java/jdk1.3/jre/lib/i386/client/lib-
jvm.so...done.
Reading symbols from /lib/libdl.so.2...done.
Reading symbols from /lib/libc.so.6...done.
Reading symbols from /usr/X11R6/lib/libX11.so.6...done.
Reading symbols from /lib/libnsl.so.1...done.
Reading symbols from /lib/libm.so.6...done.
Reading symbols from /usr/lib/libstdc++-libc6.1-1.so.2...done.
Reading symbols from /lib/ld-linux.so.2...done.
Reading symbols from
/usr/java/jdk1.3/jre/lib/i386/libverify.so...done.
Reading symbols from
/usr/java/jdk1.3/jre/lib/i386/libjava.so...done.
Reading symbols from
/usr/java/jdk1.3/jre/lib/i386/libzip.so...done.
Reading symbols from /lib/libnss_files.so.2...done.
Reading symbols from
/usr/java/jdk1.3/jre/lib/i386/libawt.so...done.
Reading symbols from
/usr/java/jdk1.3/jre/lib/i386/libmlib_image.so...done.
Reading symbols from /usr/X11R6/lib/libXp.so.6...done.
Reading symbols from /usr/X11R6/lib/libXt.so.6...done.
Reading symbols from /usr/X11R6/lib/libXext.so.6...done.
Reading symbols from /usr/X11R6/lib/libXtst.so.6...done.
Reading symbols from /usr/X11R6/lib/libSM.so.6...done.
Reading symbols from /usr/X11R6/lib/libICE.so.6...done.
Reading symbols from /usr/java/jdk1.3/jre/lib/i386/libfontman-
ager.so...done.
Reading symbols from /usr/lib/gconv/ISO8859-1.so...done.
0x404fa320 in __poll (fds=0x805ac50, nfds=1, timeout=2000)
    at ../sysdeps/unix/sysv/linux/poll.c:45
45      ../sysdeps/unix/sysv/linux/poll.c: No such file or
directory
```

Listing 2

```
(gdb) info threads
  12 Thread 11726  0x4046e58b in __sigsuspend (set=0xbe3ff808)
    at ../sysdeps/unix/sysv/linux/sigsuspend.c:48
  11 Thread 11724  0x404fa320 in __poll (fds=0x48cf8d60,
nfds=2, timeout=250)
    at ../sysdeps/unix/sysv/linux/poll.c:45
  10 Thread 11723  0x4046e58b in __sigsuspend (set=0xbe9ff82c)
    at ../sysdeps/unix/sysv/linux/sigsuspend.c:48
  9 Thread 11722  0x4046e58b in __sigsuspend (set=0xbebff7bc)
    at ../sysdeps/unix/sysv/linux/sigsuspend.c:48
  8 Thread 11718  0x4046e58b in __sigsuspend (set=0xbedffae0)
    at ../sysdeps/unix/sysv/linux/sigsuspend.c:48
  7 Thread 11717  0x4046e58b in __sigsuspend (set=0xbefffaf8)
    at ../sysdeps/unix/sysv/linux/sigsuspend.c:48
  6 Thread 11716  0x404e07f1 in __libc_nanosleep () from
/lib/libc.so.6
  5 Thread 11715  0x4046e58b in __sigsuspend (set=0xbf3ff7f8)
    at ../sysdeps/unix/sysv/linux/sigsuspend.c:48
  4 Thread 11714  0x4046e58b in __sigsuspend (set=0xbf5ff830)
    at ../sysdeps/unix/sysv/linux/sigsuspend.c:48
  3 Thread 11713  0x404e07f1 in __libc_nanosleep () from
/lib/libc.so.6
  2 Thread 11667 (initial thread)  0x4046e58b in __sigsuspend
(set=0xbfffd9e0)
    at ../sysdeps/unix/sysv/linux/sigsuspend.c:48
* 1 Thread 11712 (manager thread)  0x404fa320 in __poll
(fds=0x805ac50,
    nfds=1, timeout=2000) at
../sysdeps/unix/sysv/linux/poll.c:45

(gdb) t 12
[Switching to thread 12 (Thread 11726)]
#0  0x4046e58b in __sigsuspend (set=0xbe3ff808)
    at ../sysdeps/unix/sysv/linux/sigsuspend.c:48
48      ../sysdeps/unix/sysv/linux/sigsuspend.c: No such file
or directory.

(gdb) where
#0  0x4046e58b in __sigsuspend (set=0xbe3ff808)
    at ../sysdeps/unix/sysv/linux/sigsuspend.c:48
#1  0x4001d1db in pthread_cond_wait (cond=0x81b8efc,
mutex=0x81b8ee4)
    at restart.h:49
#2  0x401cc7bd in ObjectMonitor::wait ()
   from /usr/java/jdk1.3/jre/lib/i386/client/libjvm.so
#3  0x401ea68f in ObjectSynchronizer::wait ()
   from /usr/java/jdk1.3/jre/lib/i386/client/libjvm.so
#4  0x4017c462 in JVM_MonitorWait ()
   from /usr/java/jdk1.3/jre/lib/i386/client/libjvm.so
#5  0x8069c95 in ?? ()
#6  0x8067579 in ?? ()
#7  0x8067579 in ?? ()
#8  0x8067525 in ?? ()
#9  0x404439b0 in StubRoutines::_code1 ()
   from /usr/java/jdk1.3/jre/lib/i386/client/libjvm.so
#10 0x40154342 in JavaCalls::call_helper ()
   from /usr/java/jdk1.3/jre/lib/i386/client/libjvm.so
#11 0x401d1801 in os::os_exception_wrapper ()
   from /usr/java/jdk1.3/jre/lib/i386/client/libjvm.so
#12 0x40154690 in JavaCalls::call ()
   from /usr/java/jdk1.3/jre/lib/i386/client/libjvm.so
#13 0x40153e8b in JavaCalls::call_virtual ()
   from /usr/java/jdk1.3/jre/lib/i386/client/libjvm.so
#14 0x40154b1b in JavaCalls::call_virtual ()
   from /usr/java/jdk1.3/jre/lib/i386/client/libjvm.so
#15 0x4018880f in thread_entry ()
   from /usr/java/jdk1.3/jre/lib/i386/client/libjvm.so
#16 0x401fef1f in JavaThread::thread_main_inner ()
   from /usr/java/jdk1.3/jre/lib/i386/client/libjvm.so
#17 0x40202a87 in JavaThread::run ()
   from /usr/java/jdk1.3/jre/lib/i386/client/libjvm.so
#18 0x401d05c3 in _start () from
/usr/java/jdk1.3/jre/lib/i386/client/libjvm.so
#19 0x4001dea5 in pthread_start_thread (arg=0xbe3ffe60) at
manager.c:213

(gdb) quit
The program is running.  Quit anyway (and detach it)? (y or
n) y
Detaching from program:
/usr/java/jdk1.3/bin/i386/native_threads/java, Thread 11712
```

Making the Case For a J2EE Application Framework

A business-oriented overview asks: Jump in, go slowly, or "wait and see"? Each response has a cost, but the benefits are many Part 1



WRITTEN BY
Paul Tindall

By the time you read this, the J2EE revolution will be gaining significant momentum. Most large companies will have some type of new development around it; many more existing and new e-businesses that leverage this technology will have emerged. Many platform providers will have delivered core J2EE functionality, committed resources to transition to this technology, or made plans to build J2EE-based interfaces to their systems. All systems go…. "J2EE is ready for the masses" will be the theme for the near term.

To give this notion of a J2EE revolution a little more perspective, we should look at it in terms of the technology adoption curve well articulated by Geoffrey Moore in his book *Crossing the Chasm: Marketing and Selling High-Tech Products to Mainstream Customers* (HarperBusiness; 1999). As with any new technology, there will be an industry adoption curve that starts with the innovators and early adopters, individuals and companies that will accept a new technology at any price and level of pain. They do this for a variety of reasons, but it's typically driven by the innovative functionality offered by the technology or a passion to help reach the long-term vision.

If all goes well in this early stage and the technology gets enough momentum behind it, it'll jump into the majority stage of adoption. Many individuals would argue, me included, that J2EE has already made this jump and is headed up this portion of the adoption curve. The individuals and companies at this stage will have seen enough success delivered by the early adopters that they're ready to engage the technology as well. At this point the technology has probably evolved to where it's more stable and has a certain level of required functionality to appeal to the broader market. We've seen evidence of this transition in the level of J2EE activity and support being generated in the industry. From this early-adopter stage the technology transitions to the late majority where it's mature and stable. Finally, in Moore's model, there will also be a group of companies and individuals, called *laggards* or *skeptics*, who will never adopt the technology.

Why have I spent a paragraph reviewing the technology adoption curve in an article about framework development? Because any talk about framework development, regardless of the underlying technology used, must encompass some sacrifice of short-term deliverables to meet long-term goals. This becomes an investment decision, which entails an understanding of both the risk and reward.

Understanding where an underlying technology is relative to its adoption in the market is required for such a critical decision-making process. This jump of the J2EE technology is forcing individuals and companies to decide whether to move full stride, take small steps, or adopt a "wait and see" approach. Regardless of the decision, there's a cost associated with the level of transition taken. Those who enter with full stride may invest more heavily, take on a bit more risk, but ultimately reap greater rewards. Those who take smaller steps may be trying to minimize cost and risk but still be ready to take advantage of the technology shift. Those following the "wait and see" approach may still be able to capture some of the residual value if the technology succeeds, but will probably have a significant opportunity cost associated with not making the transition and being left behind. As such, any individual or company must determine how best to progress into the new technology in terms of assessing hype versus reality, hiring and training the appropriate individuals to make the transition, understanding what base infrastructure is available, and determining what the next steps should be.

Application Frameworks and Components

Assuming that there's more reality than hype and that J2EE has crossed over into the early majority stage, one of the best ways to jump directly in and capture the momentum is for individuals and companies to figure out how to rapidly implement high-quality, repeatable J2EE solutions. Those that do will be successful to a point in this early majority phase, but also in the late majority. One mechanism to reach this goal is to start new developments with the mindset toward building enterprise-level components and the application frameworks to host them.

The adoption of J2EE has already started to fundamentally shift the application market, as individuals and companies are now seeking prebuilt or jump-start components rather than end-to-end platform solutions to meet their requirements. The traditional system development model has been to pick a full-featured platform and try to configure it to meet the needs of the project. In many cases the platform may meet only 20–30% of the functionality requirements, and extra time and money must be spent to fill in the gaps. The cost of this extra work is often many times greater than the initial investment in the platform.

If this model is flawed, why has it been in place for so long? It grew over time out of necessity because of the lack of lower-level standards to drive baseline infrastructure. In essence, a customer buying a platform product was buying the underlying horizontal infrastructure as well as the higher-level vertical functionality. This horizontal infrastructure, with the advent of J2EE, can now be built using a robust, standardized, distributed object model. With a baseline infrastructure in place, individuals and companies can shift time and energy from the nonvalue-added parts of the application to the parts that do provide value, such as the niche functionality that may be their core line of business or competitive advantage. With component-based development, this can be accomplished in a manner that allows implementation of only the needed functionality in an à la carte fashion.

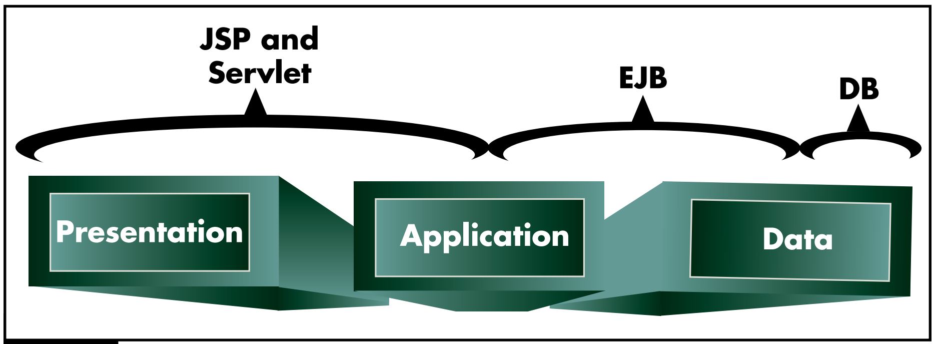


FIGURE 1 How J2EE maps to the traditional distributed architecture

What Is a Framework?

If J2EE is good, what exactly do I mean by framework? You could argue that J2EE is already a framework and that nothing new is needed. While this is a defensible argument, I tend to think of J2EE as more of the foundation and raw materials necessary to build something much bigger. J2EE is a means to an end, not the end itself. This model is analogous to going to your favorite do-it-yourself home center, buying the necessary raw materials, tools, and instructions, and building something for your home. You may choose to go to one home center over another, perhaps because of convenience, quality of service, or the variety of products offered. In the end, though, the basic materials, tools, and instructions remain the same regardless of where you shop.

As any do-it-yourselfer knows, it takes a long time to build something from scratch, especially if you're just learning how to drive a nail with a hammer and have just read the how-to books and pamphlets for the first time. If we then start to think about building a house using prefabricated components, such as roof trusses and walls, we begin to see some advantages. These trusses and walls are built by specialists using proven techniques and processes. You could argue that in this approach we give up quality and flexibility, but we can point to prefabricated homes in Sweden that rival some of the best custom homes in the United States when judged along these same quality and flexibility attributes.

Where Do Frameworks Come From?

Turning attention back to J2EE, as more projects are developed and more code is written, patterns start to emerge, indicating an opportunity for componentization, or prefabrication, using the terminology from the home building example. This componentization can simply be in the form of the development of helper classes, but it can also take the form of reusable, configurable components such as EJBs and JavaBeans. These components can be used to support functionality such as database interaction and transaction management, command and response brokering, system authentication, mail processing, queue working, system reporting, application integration, and so forth. A significant part of their behavior can be driven by configuration information stored in property or XML files rather than in lines of code.

We can then begin to think of these various components as configurable engines or brokers within a much larger system. Once our engines and brokers become cognizant of one another and begin working together within the larger system, we have the start of a framework. Over time, these engines can be assembled in a loosely coupled fashion to form more commonly used vertical functionality, such as user, catalog, sales processing, content, or project management. Either by plan or by chance, a framework is born.

Why Do I Care About a Framework on J2EE?

While frameworks can be built on top of just about any distributed component technology, such as CORBA, COM+, or J2EE, the case for the latter is quite compelling. Perhaps the strongest argument mentioned in the introduction to this article is that J2EE is based on Java, and the technology has jumped into the early majority phase, while the other technologies are well into their late majority of adoption. What does that mean to you, the person responsible for implementing a new application on J2EE?

Java and the J2EE specification have matured to the point where the goal of writing once and running anywhere is a reality. Certain "100% Java" J2EE containers have made it possible to have a single code base run across just about any Java-compliant operating system and hardware platform. For example, you can perform J2EE-based development on a laptop computer running NT. You can check the code into a shared Linux environment at the office for testing, then push to a Solaris production environment at the data center. All this can be done using the exact same code base.

Another advantage afforded by this flexibility is that you can start your production environment small and scale it to meet your growth. For example, you may be a start-up with grand aspirations but need to minimize your cash outflow while you hone your business model. In this scenario, for example, you can start your production environment on nonclustered Linux running on dual processor Intel-based servers. Once your business starts to grow, you can easily progress to clustered Linux running on quad processor Intel-based servers. If your business model really takes off, you can progress into eight-way Intel-based servers, or make the decision to switch to various sizes of Sun servers running Solaris. Again, we can do this progression with the same code base. If you're a company with an existing infrastructure of Windows NT or Sun or Alpha, you can utilize this hardware to take advantage of the J2EE technology. There's a high degree of value in this flexibility.

Another advantage of J2EE is that it brings about competition, which is good for you, the consumer of this J2EE infrastructure. At last count there were something like 14 EJB containers available on the market with as many servlet containers. Likewise, there's JDBC support for every popular database server vendor. This means that if properly constructed, a single code base can be shared across both open-source and commercial infrastructure providers. In the open-source camp are the Resin and Tomcat servlet engines, the Enhydra and Sun Reference EJB containers, and Postgress and InterBase database servers. In the commercial camps you can use products such as iPlanet, BEA WebLogic, ATG Dynamo, Allaire JRun, IBM WebSphere, GemStone, and BlueStone as your J2EE containers. Similarly, you can choose among Oracle, Sybase, Informix, and others for your database server. In short, there are ample choices to meet a particular project's requirements.

You may think that a J2EE framework isn't for you. Surprisingly, just about any software development effort can benefit from a framework approach. If you're a systems integrator, J2EE allows you to support a much larger customer base that may already have established hardware and operating system preferences. Adding a framework philosophy to J2EE allows you to build reusable core components that can be used across projects. Rather than building from scratch at the outset, you can begin leveraging work performed in the past. This continual reuse allows an ongoing improvement (or "hardening") of the core code base while letting you focus on the more complex functionality that the client really needs you to be working on. Additionally, as the company brings on more developers and a consistency in components and frameworks is fostered, a certain level of operating efficiency and leverage is created.

In a start-up business there are two absolute truths:

- ***Cash is king:*** We've already talked about how J2EE allows better control over cash flow without building a code base that will have to be ported or thrown away at some later stage.
- ***Get there fast:*** And be flexible enough to change once you're there. In many cases the work to get there fast will make it difficult to be flexible when you arrive. Again, the value of flexibility is high. A good framework is often worth the investment in time and money.

For a large corporation that may have a heterogeneous hardware and operating system environment, J2EE can help protect those investments. Many large corporations have departmental pockets of development activity that are most likely building applications using similar components – they're just hooking them together a little differently and adding a couple of niche components that define their department. Likewise, there may be multiple departments writing similar interfaces to get similar data from the same ERP system. Considering even the broader scope, many companies have had to deal with the issues of platform or ERP lock-in over the years. This can force a company to adhere to preconceived business processes enabled by these applications, leaving little room for the company to engage in business-process improvement activities that could generate competitive advantage. A component-based J2EE framework mentality can help even large companies leverage current hardware investments and work across multiple departments, reducing its dependence on lock-in.

How Do I Get One?

By now you may be asking yourself how to bring a J2EE-based application framework into your development efforts. Fortunately, you have several options:

- Take a look at something available in the open-source community, such as the iCommerce platform by GemStone. You may choose to use such an existing framework to get an idea of what's involved, and perhaps as a starting point to one you might build on your own.
- Look at purchasing a jump-start platform, such as the Commerce and Personalization Server from BEA.
- Start converting your existing development efforts to more of a component-based, framework-oriented process.

A Quick Intro to J2EE

Regardless of the route taken, the horizontal aspects of a J2EE framework should be fairly similar. Most distributed implementations will start with what's known as a multilayered architecture that, at a minimum, partitions out presentation, application, and data functionality into discrete logical and physical components. Some frameworks further refine the application layer to break out domain and interaction services.

Figure 1 shows this basic layering and how it maps to the J2EE technologies. In general terms, the presentation layer provides the presentational and navigational aspects of the application. The application layer is often considered the heart of the system because it defines not only the problem domain in terms of simple entities tracked by the system, but also the behavioral aspects of those entities and how they interact with one another. The data layer is responsible for managing the long-term state within the system, typically in the form of RDBMS storage.

As an example, a user may interact with an HTML page (presentation layer) that's submitted back to the Web server. The Web server may respond immediately based on navigational programming, or it may need to interact with the application server (application layer) to gain access to data or behavioral aspects of the system. If the application server needs to manage the state of some domain object to complete its task (say, a database retrieval), it will have to contact the database server (data layer) to do so. Once the application server has satisfied the data and behavior requests, the results are passed back to the Web server for final navigational and presentational work before sending a response back to the user.

Layering is important because it allows the distribution of processing across multiple logical processes, whether they're physically on one machine or spread across multiple machines. This layering approach also sets the initial foundation and boundaries for the components that will make up the framework. Such modularity provided by a component-based framework has several benefits:

- Various development tasks can be split between multiple developers according to the components and partitions.
- Long-term maintenance of the application becomes more manageable as well-designed components can be easily replaced over time as improvements are made.
- Test harnesses can be built rather easily to exercise the functionality by one or more components within the framework.

Conclusion

With a business-oriented overview of a J2EE-based application framework complete, I'd like to turn technical and focus on a real-world example. Part 2 of this article will provide a more detailed dive into such a J2EE framework used on one of our recent projects (www.bid4real.com). Not only will we cover the framework aspects of the system, but we'll go into detail on the hardware and operating system selection process, the clustering solution implemented, design decisions made along the way, and some of the components that were leveraged.

ptindall@emerging.com

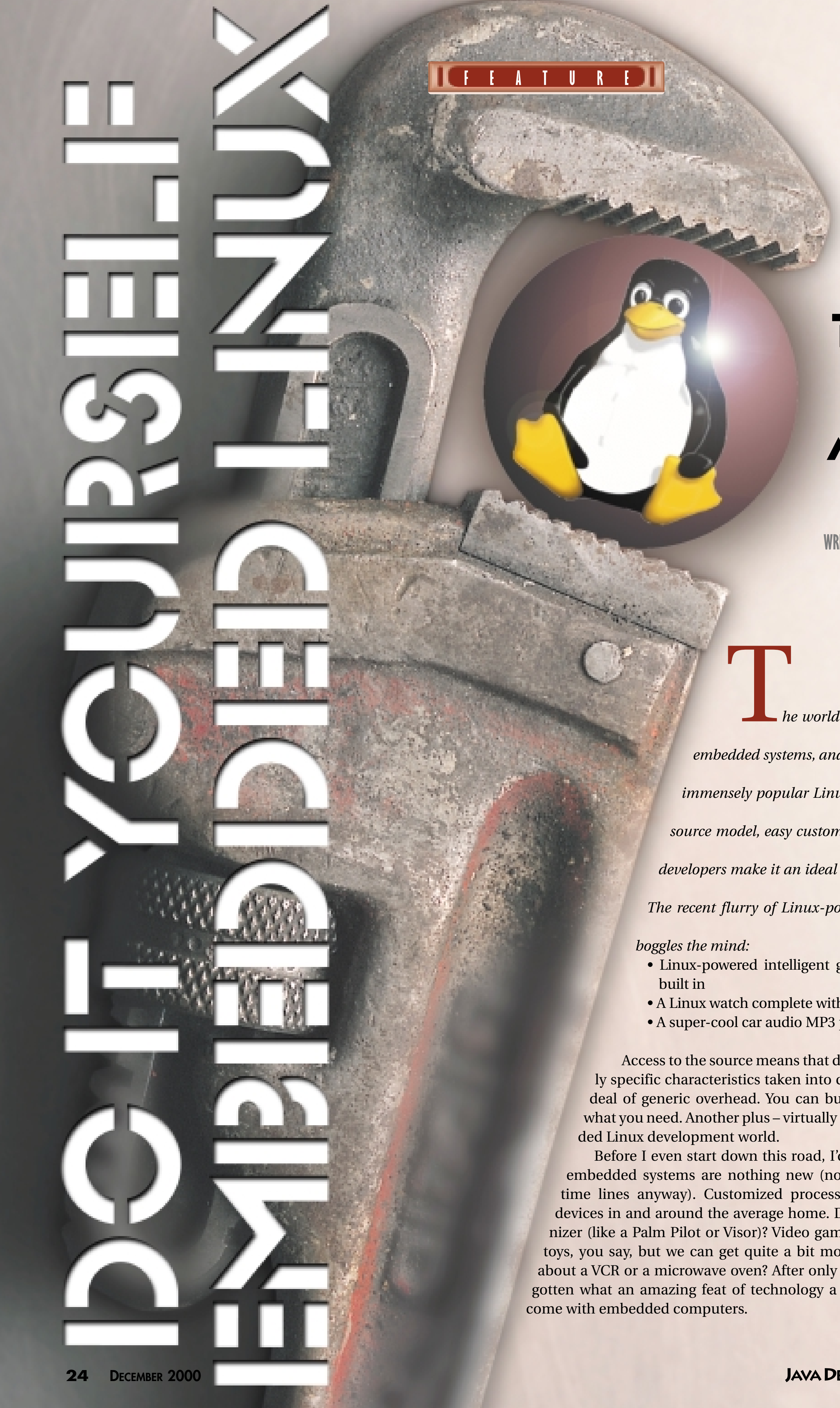
AUTHOR BIO

Paul Tindall, technology director at Emerging, has 12 years of experience in software and application development across many architectures and technologies. He's the author of the book Developing Enterprise Applications – An Impurist's View *(Que; 2000).*

FEATURE

DO-IT-YOURSELF EMBEDDED LINUX

GO AHEAD TRY THIS AT HOME



WRITTEN BY MARCEL GAGNÉ

The world is abuzz with the promise of embedded systems, and hopes are riding high on the immensely popular Linux operating system. Its open-source model, easy customization, and popularity with developers make it an ideal choice for embedded systems. The recent flurry of Linux-powered devices on the market boggles the mind:

- Linux-powered intelligent golf carts with a GPS system built in
- A Linux watch complete with X Window display
- A super-cool car audio MP3 player

Access to the source means that devices can be built with highly specific characteristics taken into consideration without a great deal of generic overhead. You can build what you need and only what you need. Another plus – virtually anyone can enter the embedded Linux development world.

Before I even start down this road, I'd like to begin by saying that embedded systems are nothing new (not in information technology time lines anyway). Customized processors exist in a vast array of devices in and around the average home. Do you have a personal organizer (like a Palm Pilot or Visor)? Video game unit? Digital camera? Tech toys, you say, but we can get quite a bit more mundane than that. How about a VCR or a microwave oven? After only a few years we've all but forgotten what an amazing feat of technology a cell phone is. Even our cars come with embedded computers.

Embedded systems were considered one of the great black holes of the (at the time) upcoming (and now long past) Y2K crisis. While many companies could refer to the source code for their financial applications, manufacturing systems, and so on, it became a bit more foggy when it came to embedded systems. In the end the most optimistic industry watchers were declaring that the real Y2K threat lay in embedded systems. While we all survived more or less intact, I do have a VCR that can only be programmed within the time frame of a single day because I can no longer set the date. It's a minor beef since I generally don't tape too many shows, but it highlights one of the problems with embedded systems to date and why embedded Linux is so exciting.

In a nutshell, the problem is too many different embedded system technologies are out there, too many proprietary chips and languages, and not enough standardization. Each chip is developed for separate applications. This makes extended development and support extremely expensive on many counts, such as the manufacturing process. When we buy a proprietary embedded system, we bank on one and only one vendor to support us throughout the life of that device. We also bank on never having to upgrade, maintain, or modify anything on that system. It's literally carved in stone.

Linux changes all this and the change is a powerful one that's being watched at all industry levels. Witness the creation of the Embedded Linux Consortium, a nonprofit vendor-neutral organization whose sole purpose is supporting embedded Linux technologies. Should you still have any doubts about the industry's interest in these developments, witness some of the names on ELC's roster: IBM, HP, Lineo, RedHat, Motorola, Palm – an impressive list.

Should you feel the need to get involved in this Linux microrevolution, rest assured that you don't need the backing of a major corporation and millions in venture capital funding. The tools are out there, free for the taking. Saying that creating embedded systems necessitates some miniaturization of the code may sound like a case of the bleeding obvious, but there's an art to building what you need and nothing more. It requires a desire to get your "hands" dirty with kernel builds, and more than a little restraint.

For developers building tiny Linux applications BusyBox is a great place to start. BusyBox, originally created by Bruce Perens (now maintained by Erik Andersen and sponsored by Lineo), bills itself as "the Swiss Army Knife of Embedded Linux." As soon as you start exploring it you'll understand why. It's a single small executable that packs over a hundred common Linux commands into a single package. This approach helps fix one of the great problems with creating a tiny embedded Linux, namely the relatively large executables that are created when linking against the GNU C libraries. To be fair, I should point out that large is a matter of perspective. A 100K program is no big deal when you have a 4 Gig drive; however, in the world of an 8MB chip it won't take long to use up that precious memory if every program is that "small."

Let's try this, shall we? Head on over to the BusyBox Web site at http://busybox.lineo.com and pick up the latest source. Contrary to many warnings, you can try this at home.

Once you've downloaded BusyBox, extract the source and build the executable on your system. Here are the steps based on release 0.47:

```
tar -xzvf busybox-0.47.tar.gz
cd busybox-0.47
make
```

This builds the binary with all tools activated. I mention this because BusyBox is quite modular in its design and allows you to build in only the functions you're looking for. This is done by editing the Config.h file you'll find in the source directory. Here's a sample from that file:

```
//
// BusyBox Applications
#define BB_AR
#define BB_BASENAME
#define BB_CAT
#define BB_CHMOD_CHOWN_CHGRP
#define BB_CHROOT
#define BB_CHVT
#define BB_CLEAR
#define BB_CP_MV
#define BB_CUT
#define BB_DATE
// #define BB_DC
```

Notice the last line where C++-style comment characters have been added (//). I've undefined the build flag for the "dc" command (a Reverse-Polish desk calculator). In all, 107 defines can be undefined to create as small a binary as your needs require.

What do you do with this executable once you've built it? How do you use it? To test your build simply type the command "./busybox" followed by the command you wish to execute. For instance, BusyBox comes with a simple telnet client. If I wanted to use it to connect to my remote development system, I'd type the following:

```
./busybox telnet devsys
```

Of course, you don't want your users typing something this pedantic if all they want is to do a file listing with "ls". What you can do, however, is create symbolic (or hard) links to the BusyBox executable. Let's work with my telnet command above:

```
ln busybox telnet
```

To use the telnet command and connect to my development system I need to type only "./telnet devsys".

Sometimes the best way to start is by looking at the work of someone else in the same environment. The BusyBox site contains links to tools that projects are currently using. One of those sites is a personal favorite of mine, tomsrtbt, which stands for "Tom's floppy which has a root filesystem and is also bootable." At least that's what the man claims. Tom is Thomas Oehser, by the way, and his little Linux provides a bevy of tools on a single floppy diskette. Granted, tomsrtbt uses a formatting trick to squeeze 1.7MB from a floppy diskette space rather than the normal 1.4. Even so, the results are a great demonstration of what can be done with a small-footprint Linux. To get your own copy of tomsrtbt, visit the Web site at www.toms.net/rb/.

The site contains both a development version and a stable release. As I write this, the latest stable distribution is tomsrtbt-1.7.185.tar.gz. Once you've downloaded the file, extract it to a temporary directory. From there you can use the install script to create your diskette. Start by putting a blank diskette in your drive, then:

```
tar -xzvf tomsrtbt-1.7.185.tar.gz
cd tomsrtbt-1.7.185
./install.s
```

That's all there is to it. Almost. You'll want to change a few things, particularly if you're on a network. For instance, the default installation boots to a preconfigured set of network addresses, including DNS entries that aren't likely to match your environment. To customize tomsrtbt for your own network, edit the settings.s file in the tomsrtbt distribution directory. The following lines represent my changes to the network parameters so that tomsrtbt can exist on my network:

```
DOMAIN=mycompany.com
IF_PORT=10baseT
DNS_1=192.168.22.10
IPADDR=192.168.22.4
NETWORK=192.168.22.0
NETMASK=255.255.255.0
```

```
PASSWD=xxxx
FD=/dev/fd0u1722
FN="b 2 60"
```

The original network numbers are in the network "192.168.1.0" and the domain is rb.com.

Once you've booted, log in as root (you can even remove the diskette at this point) and start exploring. If you wander over to the "/usr/bin" directory and do an "ls -l", you'll notice something interesting. Several of the files have the same size, date, and time stamp. You'll also notice that one of these identical files is called *busybox*.

This little diskette is a great tool in itself. It supports SCSI and PCMCIA, and contains a host of handy tools. Whenever I go off-site to do some work, I carry a tomsrtbt with me for emergencies (not to mention a known quantity on an otherwise unknown network). One of the other great things about it is that it understands your hard drives and lets you mount them. For instance, if I boot from the floppy on a Windows workstation, I can mount the C: drive and navigate it from my little Linux. From the root prompt I'd do this:

```
mkdir /mnt/cdrive
mount /dev/hda1 /mnt/cdrive
```

From there I can navigate the hard drive and make backups with tar, cpio, or ftp files to another server. I also carry other tiny distributions that have been put together for very specific purposes. At any given time you'll find LOAF, Trinux, and tomsrtbt in my arsenal.

If you'd like to create your own specialized boot diskette, consider visiting BYLD (originally developed by Erich Roncarolo) at its SourceForge site, http://byld.sourceforge.net. You'll find a package that's designed to allow you to create customized single floppy Linux distributions. You decide your own applications and build to your needs whether it's a router, firewall, or rescue disk. To try your hand at a personal tiny Linux, download the source and extract it to a work directory. I decided to try out the latest beta release:

```
tar -xzvf byld-1_0beta3.tar.gz
cd byld-1.0beta3
```

If you haven't already done so, load your "loop" module:

```
insmod loop
```

There's a lot to cover here and I have no room to go into it, but if you're feeling impatient, start building your floppy right now:

```
./BuildRoot SingleFloppy
```

Among other things, you'll see BYLD putting together and compiling BusyBox to stand in for a number of basic Linux commands. Once this process is over, do the following:

```
./MakeImage
```

At this point the program creates a temporary loop device filesystem (in case you wondered why we were loading the driver), writes out the image, compresses it, and prepares the final product:

```
./WriteImage
```

As you may have expected, this writes the whole thing out to your floppy. I haven't done any customization to the source in any of this. This is where the experimenting comes into play. When I tried to build my first diskette I decided to go easy on myself and simply use the existing kernel from my RedHat 6.2 system. Nice enough idea but I went over the 1722KB limit for a boot diskette. Because I really wanted to have my very own single floppy Linux, I decided to comment out some of the packages from BusyBox in the BYLD directory. After several other attempts (I broke down and deleted a handful of programs from the root/sbin directory), the "WriteDisk" process told me that my image was now smaller than the maximum of 1722K. I hit Enter and waited for my diskette to be created.

Note: You may or may not already have a 1722K diskette device defined on your system. If it doesn't exist, you can create one like this:

```
mknod /dev/fd0u1722 b 2 60
```

My new diskette chugged away for a while and eventually the "WriteDisk" process proudly proclaimed that my BYLD disk was complete. I could simply reboot the system with the diskette in place. Since I was busy using that machine (and writing this article on it), I walked over to my development machine, popped the diskette in, rebooted, and waited. I had trimmed a bit too much. Finally, I broke down and decided that I'd have to build my own kernel. At 700K, this is where I'd have to do my real work. Out came the kernel sources followed by those magic keystrokes, "make xconfig," and I was off to trim away until I had a small "needs-only" kernel.

I copied my new kernel (ln -s linux/arch/i386/boot/bzImage vmlinuz) to the BYLD directory and tried my luck again.

Success! I now had the beginnings of my very own, custom embedded Linux system, a cornucopia of tiny Linux apps in my own "unique" distribution.

While I found my own entry into embedded Linux development exciting, it hardly constituted a well-thought-out approach. You'll want to build you own kernel optimized for space. You'll also want to decide precisely what you want on this complete system. Furthermore, you need an idea of the environment you're building for; in other words, what do you want this thing to do when you're through?

I've given you just a few examples of embedded development. Other projects that combine the miniaturization of Linux include Coyote Linux, the Linux Router Project, and the Trinux security toolkit. This barely touches the surface. For those of you raring to get started, the "Resources" section below provides links to everything I've mentioned.

Just as Linux has changed the face of the computing world in the last few years, it will also change the way we interact with the devices in and around us. Intelligent televisions, microwave ovens, refrigerators, security systems, and just about anything else you care to put your mind to will start appearing in the near future. The open-source development model should make it possible to drastically reduce development costs, thus making the arrival of new technology that much closer. Not only that but Linux has been ported to numerous processor types. We may find that old hardware suddenly has a new lease on life as we reprogram it to perform new functions.

Go ahead. You can try this at home.

Resources

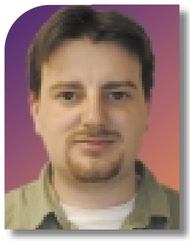
- *BYLD (Build Your Own Linux Diskette):* http://byld.sourceforge.net
- *The BusyBox site at Lineo.Com:* http://busybox.lineo.com
- *Coyote Linux:* www.coyotelinux.com
- *Embedded Linux Consortium:* www.embedded-linux.org
- *Linux Router Project:* www.linuxrouter.org
- *tomsrtbt:* www.toms.net/rb/
- *Trinux Security Toolkit:* www.trinux.org

Author Bio

Marcel Gagné is president of Salmar Consulting Inc., a systems integration and network consulting firm. He's currently working on Linux System Administration: A User's Guide from Addison Wesley Longman due out in 2001.

mggagne@salmar.com

Source Code – It's Not Pretty



WRITTEN BY
ALAN WILLIAMSON

I've had an interesting month, I can tell you. This was the month we had our annual n-ary Halloween party. The time of year when we go a little wild, dress up stupidly, and generally make a right tit of ourselves. This year I decided to don a genie outfit as opposed to the eighteenth century ladies garb I borrowed last year. It was a good do, couldn't complain. We all got horribly drunk, sang awful '70s songs in true karaoke style, and watched fireworks light up the sky. Thankfully it only comes once a year, and even then it doesn't seem 52 weeks since the last time. Amazing how fast it's all going and how little Murray's singing has improved.

Want to Buy an Operating System?

I'm sure you didn't miss the news and media buzz around Microsoft and their wee boo-boo with respect to leaving the lid off the cookie jar only to have someone come along and pinch the source code to Windows. It would appear that someone in the Ukraine is now poring over reams of their source code trying to make some sort of sense of it. Apparently the hacker snuck in using an official Microsoft home worker's remote dial-in account and then, once in, set up the necessary access rights to get into other areas of the Microsoft Empire. Apparently this occurred over a period of weeks, and apparently Microsoft was well aware of the hacker's movements.

I've used the word *apparently* an awful lot, haven't I? Don't panic – this is by design. The problem is that there're so many conflicting reports about it all that getting to the truth is something I doubt we mere mortals are ever going to do. But if all this is true, one has to question a number of obvious blunders.

One, if Microsoft was aware of the hacker's movements right from the start, why did it feel the need to shut down the remote-access servers for its some 35,000 employees? Second, if it knew from the start, why was the FBI called in so late? I don't know, but it makes wonderful material for us writers to play with.

I was giving it some thought: the theft of source code. Is it really that serious? Let's look at it from several points of view. First of all, let's assume that Microsoft's developers are not the elite-of-the-elite as we're all told in software folklore. Let's assume they're just an ordinary crew of software developers. With this comes the chance that the code isn't that well documented. You know what I mean.

Of course, you're supposed to document your code as you go along, but the module you're working on is late, the project manager is breathing down your neck, and your partner is continually vibrating your phone in a desperate bid for your attention. Ah, what the hell. I'll document it in the morning. Sadly, nine out of 10 times the morning never comes and only after the code review does the need for some documentation arise. You can't really remember what it did, let alone how it did it. You whisper to yourself that it works, and no one will ever need to go near it again. You then decide to throw enough comments that will get it through code review and hope it slips through with minimum fuss. Sound familiar? I'm sure you're sitting there convincing yourself that's not how you work. Sure it is...who are you kidding? You're not a real developer if this scenario hasn't happened at least once (if an n-ary developer is reading this, I'm on to you!).

Assuming that some of the core dudes have suffered from this fate at some point, that even they can't figure out what the code is doing (which if you count how often Windows 2000 crashes, then I can believe that), what are the chances of an outside body being able to decipher it? Slim, methinks.

Maybe that's a bit harsh. Assume that the code is all nicely documented, every line clearly marked with the necessary annotations that clearly pave the flow of execution. That said, I'd bet money that somewhere in the Windows' code base is a comment that says something like:

```
/*
Not too sure what this line does, but
don't take it out as it seems to be
fairly important.
*/
```

It has to. Even Microsoft developers are human. Aren't they?

So a hacker has got his or her hands on the crown jewels. Well, the hacker can't do any worse. The worse thing that could happen is for it to be e-mailed back to Microsoft with various improvements. Remember back when Netscape open sourced its crown jewels? Did you ever download it and have a look? It was a mess. How they thought anyone would take it on and do a "Linux" on it I'll never know. Based on the state of the code for Netscape, I don't think Microsoft has too much to worry about. Only good can come of it.

I think the media has to report the facts a little more clearly in the future. I had someone come up to me and ask me what Microsoft was going to do now that it no longer had the source code to Windows. I tried my hardest to stifle a small titter as I explained that "stealing" the source code didn't mean there was now an empty cupboard somewhere where the code used to be. Bless such innocence, I say.

Oh, Go on Then...You Can Play

In early November Sun announced a fairly major milestone in the history of Java – almost a year since they nearly fell out with IBM about not allowing them enough say in the evolution of the language. As part of their Community Process program, they've set up two governing bodies and, more important, given those bodies control over the future of Java. One body will concentrate on Java Micro Edition, while the other deals with the Standard and Enterprise Editions.

Sun, wonderfully democratic, has appointed itself with a seat in each body, while everyone else has to be elected. I guess it's their party so why not, eh? It would appear that IBM is the only company that's in the running to have representation on both bodies.

These bodies have to take Java and move it forward, developing the interfaces that will enable the next generation of applications. This on the whole is very exciting. For one, it should silence the critics of Sun's near monopoly on Java. I for one didn't have a major issue with Sun taking the mantle of maturing this new language. Of course, I had my reservations and what have you, but never enough to stop using Java.

Java is like a small child. It's now nearly 6 years old and the time has come to venture out into the world of primary school. Up to now it has had to listen and take its lead from its parents (Sun). It's now being introduced to other teachers who will take its evolution to the next level. After primary school comes high school. Here it has to be tough enough to play in the schoolyard with the other children. George Paolini was even reported saying that an open-source Java wasn't that many years away after all. Well, they do say that once a child gets to high school, that's when the problems generally come as the parents no longer have the same level of control.

Whether Java will survive the schoolyard with the other open-source kids no one can say for sure. It seems to have worked for Apache and to a certain degree Linux, but can a language that pertains to be write once, run anywhere really survive the pushing and pulling that will transpire once it gets into the hands of the world at large? Who knows?

From the Dead It Arises

I'm seriously getting into "Buffy the Vampire Slayer" these days. I had avoided it for so long, writing it off as a silly teenagers' series. Boy, was I mistaken. It's extremely funny and well worth checking out if you, like me, have not really given it a fair chance. Buffy, our heroine, slays vampires as her destiny. A vampire results when human and vampire exchange blood; chicken and egg situation there, if you see what I mean. Generally, if you come back as a vampire, you're not in as good shape as you were when you were fully alive and kicking.

With that, let me I introduce you to the latest vampire that's been created. It would appear that boo.com, the ill-fated fashion e-tailer that went bankrupt in early spring with 150 employees, has risen from the grave with just 10 support crew. A brave move by its new owner who assures us that it's here to stay and people shouldn't have any concerns about placing orders on its Web site.

Maybe some things are best left dead, or do we have to rely on an e-Buffy to come and save us from the charms of a dot.vampire?

Speaking of Creations

It's a rather exciting month. My first wee one is about to enter the world and I'm frantically trying to prepare myself for this new addition to my life. It's exciting and bloody damn scary at the same time. By the time I write the next column I could be a "dad." Gosh what a strange feeling that is. Me? A father? Now there's a thought.

With that I had better go. I'm hearing contractions in the background and need to start counting. Well, what else can I do? Precious little, let me tell you!

Author Bio

Alan Williamson is CEO of the first pure Java company in the UK, n-ary (consulting) Ltd (www.n-ary.com), a Java solutions company specializing in delivering real-world applications with real-world Java. Alan has authored two Java servlet books and contributed to the servlet API.

alan@n-ary.com

Transcoding Technology Extends Wireless Capabilities to Linux

As e-business evolves, the need for transcoding solutions grows

WRITTEN BY
JOE ANTHONY AND
ANU MANNAR

A forecast by International Data Corporation (IDC) predicts that by the end of 2002, there will be more wireless subscribers capable of Internet access than wired users. In fact, IDC predicts that almost 1.3 billion people worldwide will be plugged into Web-capable phones by 2004, compared to just 700 million people with ordinary Net connections. Another 100–200 million or so will use a variety of new kinds of devices like Internet appliances, gaming machines, set-top boxes, and everyday household appliances.

The chief obstacle to ubiquitous computing comes from the technology itself. Multiple formats, markup languages, device capabilities, and network constraints threaten to limit the promise of pervasive computing. The potential will be realized only when there's a way to bridge disparate data seamlessly, transcending multiple data protocols, devices, and users.

Businesses and developers can reauthor content and applications – an expensive undertaking – or they can rely on transcoding software. Transcoding dynamically adapts source content to match particular devices.

Leading software vendors are embracing transcoding technology to connect their workforce and customers to business applications and Web content. IBM's server-side software provides Linux developers with a solution for the wireless environment. Transcoding Publisher Version 3.5 is supported on Red Hat Linux Version 6.2, SuSe Linux 6.4, Caldera eServer 2.3, and Turbo Linux 6.0.

The Need for Transcoding

Computers have long been used to get information and perform business functions. When the Internet was adopted by the everyday computer user, the opportunity to communicate and interact for business and personal use exploded to almost unbelievable levels.

Now that opportunity has gone a step further with the advent of pervasive devices such as cell phones and personal digital assistants. The potential to do business and share ideas is tremendous.

Consider the possibilities:

- Customers on mobile devices trade stocks and transfer funds.
- Employees access the company intranet and enter data from the field.
- Travelers check real-time itinerary updates en route.

Transcoding Publisher manages the complexity of new devices and markup languages, allowing businesses to focus on their core functions. This flexible solution provides an easy-to-use, simple way for handheld devices, traditional personal computers, and back-end systems to communicate and readily exchange data.

Effective transcoding software will:

- Extend existing Web content to new devices.
- Streamline delivery so that content is provided efficiently.
- Customize content presentation for end users.

Extend the Reach of Content

Today data is available from many sources, including host and Web applications, but the lack of a universal data exchange format limits companies and consumers from exploiting the true value of the information. Transcoding extends the reach of content by dynamically transforming data so that it's tailored for new environments, including the wireless Internet.

Enterprise Content

There is substantial opportunity to extend the use of enterprise data by transcoding the formats and bringing the data out from behind host protocols. As enterprises expand to new e-business markets and workforces become more mobile and widespread, easy access to enterprise data becomes more critical.

With up to 80% of all business data residing on mainframes and other host systems, growing e-businesses need quick, easy, and efficient ways to extend data to Web users and to users on the new breed of pervasive devices. One

AUTHOR BIOS
Joe Anthony and Anu Mannar are responsible for the brand and market management of IBM's WebSphere Transcoding Publisher.

answer is Transcoding Publisher, coupled with IBM WebSphere Host Publisher. Host Publisher lets you implement Web applications, such as Web self-service, from existing 3270, 5250, virtual terminal, Java, and JDBC host applications without modifying the host applications. It then takes that data and extends the reach to handheld devices.

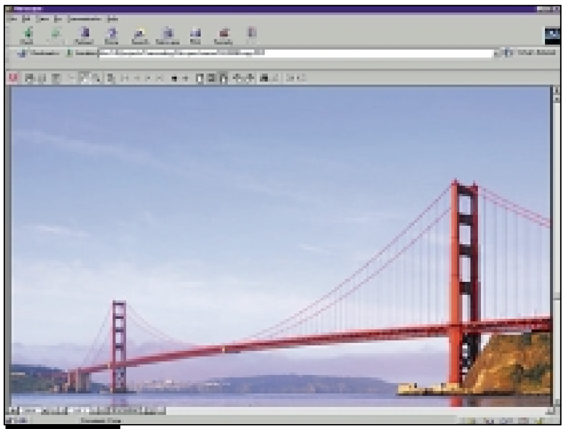


FIGURE 1 Original image on Web site

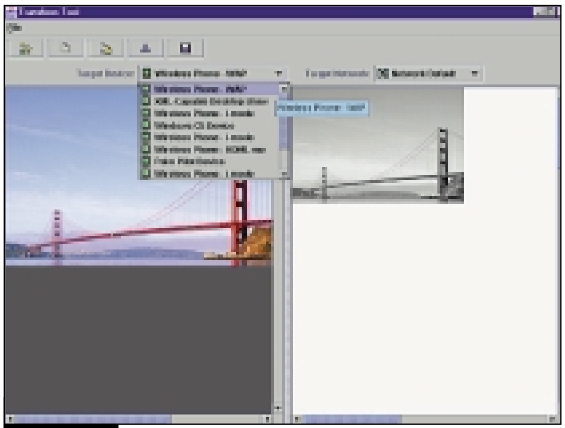


FIGURE 2 Transform Tool of the WebSphere Transcoding Publisher. Original message is on the left, and the view of the transcoded image is on the right.

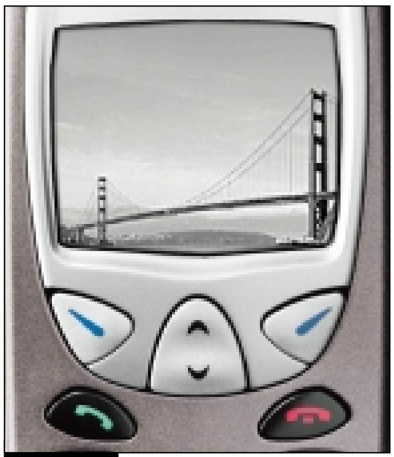


FIGURE 3 Transcoded image as it would be delivered to a phone. Original source photo is unchanged.

This solution allows you to leverage your existing IT assets to reach new markets and customers, and to minimize costly development expenses or time-to-market delays.

Web Content

The latest e-business trends require that your Web content be integrated – part of a larger solution that may include Web-based data from partners and suppliers. The challenge is that not all Web pages are created equally. Because HTML allows for variation in structure and language dialect, integrating Web-based information from multiple sources can be difficult and involve a substantial amount of rewriting and reprogramming.

It's imperative to extend the reach of Web content to users on nontraditional clients. Consider the mobile user who wants to get weather updates or perform bank transactions from a pervasive device, such as a cell phone or an automobile-based browser. The challenge is that Web content is written in HTML, not the specialized markup languages required by the new pervasive devices. The Web-savvy customer expects a solution today that provides a user-friendly experience and convenient access to information.

Transcoding can solve this problem by dynamically bridging the different HTML structures, tailoring the content to the specific device – whether it's a PC on a low-bandwidth network, a cell phone or a Palm Pilot – and conveniently delivering the customized content to the user.

The standard transcoders that come with Transcoding Publisher handle the following transformations:

- HTML to Wireless Markup Language (WML)
- HTML to i-mode (a variant of compact HTML)
- HTML to Handheld Device Markup Language (HDML)
- XML to XML variants through the use of XSL stylesheets
- JPEG images to GIF and wireless bitmap
- GIF images to JPEG and wireless bitmap

The standard Transcoding Publisher transcoders can be used independently or in conjunction for more complex content transforms. Additional transcoders can be added from third-party vendors, future IBM releases, or customized in-house transcoders. The toolkit provides samples and documentation to allow developers to write easily pluggable transcoders.

Streamline Delivery

The new pervasive devices include cell phones, screen phones, voice-capable browsers, and personal digital assistants such as Palm Pilot, IBM WorkPad, and handheld Microsoft Windows CE devices. Industry reports predict that the adoption rate for these devices will continue to rise at an accelerated pace. Both business and consumer users are quickly integrating these devices as part of their everyday habits.

It can no longer be assumed that a customer is using a traditional computer to access data and applications. To reach the same customer tomorrow, you may need to enable your applications and content for pervasive devices. For example, Japan's NTT DoCoMo, the country's largest phone carrier, has almost 13 million people using its i-mode wireless data service, which already offers color and video over phones.

It's difficult to deliver data and applications in an e-business world filled with countless variables (nonstandard device types and markup languages), let alone deliver the content efficiently across the network. After all, the value of data in the mobile computing world is tied directly to the timelines and convenience of accessing the information. In addition, certain business realities unique to the wireless Internet must be taken into account.

For example, it currently costs $3.47 (U.S.) to send a megabyte of data over the cellular network, compared with $0.013 (U.S.) over wired phone lines, according to the Mobile Wireless Internet Forum Coalition. Does this mean that users will prefer the wired Internet because of costs? No. The convenience and future potential of the wireless Internet is compelling end users to migrate to the wireless Internet in unprecedented numbers.

However, businesses need to streamline the delivery of their content to the mobile user so that information is provided in a time-efficient, cost-effective manner. Businesses need to consider certain variables, such as:

- Network bandwidth (LAN, phone line, wireless)
- Network architecture and configuration
- Size of the data packet
- Ease of integration with other software applications

For example, network capacity influences the amount of content that's most efficient to send to the mobile user. The specific network infrastructure will determine where and how the content transformation technology should be deployed in the network. The size of the data packet impacts the timeliness of the content delivery. The ease of integration with existing software applications affects the response time and effectiveness of the overall content delivery solution. All these factors point to the importance of a streamlined content delivery mechanism.

Transcoding Publisher can solve these challenges by streamlining delivery in the following ways:

- Converting HTML to simplified HTML
  - –Converting images to links to retrieve images
  - –Converting simple tables to bulleted lists
  - –Removing features not supported by a device, such as JavaScript
- Reducing image scale and/or color level to make images smaller, easier to transfer, and quicker to render on constrained devices
- Responding to the limited storage capacity of many WML, HDML, and i-mode phones by subdividing content into small sections that can be viewed more effectively on such phones, a process known as *deck fragmentation*
- Tracking network profiles so that the content can be transcoded according to network constraints
- Offering several flexible network deployment options

Customize Presentation for End Users

Not only does content need to be delivered efficiently, it must also be delivered in a way that the user appreciates. By 2003, more than 30% of consumer-facing application user interfaces will be designed to favor emotional experience over user efficiency, according to the GartnerGroup.

In other words, pervasive computing isn't about browsing the Internet; it's about delivering unique content that's optimized for wireless devices and personalized for wireless Internet users. The primary driver for a wireless Internet user is time and convenience. Receiving only the information needed in a friendly, personalized manner is the ultimate objective of the wireless Internet user.

Transcoding Publisher enables you to customize the content that's delivered to end users in the following ways:

- ***Content selection***
  - –*Clipping:* With some custom JavaScript, clippers can be written to extract or clip select content to be delivered to the device. This allows a cell phone user to retrieve only the relevant portion of a Web page, such as the daily stock price from a corporate home page filled with other links.
  - –*Annotation:* Annotation makes it possible to tailor Web pages for a device without programming by using an XML-compliant annotation language for selecting and tailoring content.
- ***Stylesheets:*** With the application of XSL stylesheets, the specific XML content to be delivered to the pervasive device and the content view can be customized for the end-user environment. This facilitates business-to-business information interchange.
- ***Device profiles:*** A breadth of device profiles allows for more detailed device-level personalization of content by taking into account the physical limitations of the devices.

WebSphere Transcoding Publisher

Transcoding Publisher consists of interrelated components that provide an open, extendable, standards-based platform for adapting data to the pervasive environment. The primary components include:

- A set of standard content transformations or transcoders
- An administration console with the ability to control device profiles, and decision criteria for intelligent content modification
- A developer's toolkit that provides specific GUI-based tools, sample programs, and detailed documentation to help the customer add or build custom transformations
- A pluggable infrastructure that allows transcoders to leverage the same core services and enables new plug-ins to be quickly integrated as part of the product.

In addition to Linux, Transcoding Publisher runs on IBM AIX, Windows NT, Windows 2000, and Sun Solaris.

For Developers

The Transform Tool places original content and transcoded content side by side, which allows developers to view the effect of transcoding actions.

The Request Viewer is essentially a visual tool for monitoring the operations of the server. It's especially useful for debugging because it allows the user to monitor the flow of requests through the server and observe which plug-ins are triggered and in what sequence.

The developer's toolkit also includes samples and documentation for creating new transcoders. The transcoding framework also allows industry-standard servlets to be incorporated as transcoders. Servlets written for other environments can be easily ported to run in the proxy, and new plug-ins written to the Java Servlet API will be usable in the many Web servers that support servlets.

Outlook

As e-business evolves, the need for transcoding solutions will continue to grow. Research on voice conversion and machine translation is already opening up new possibilities. Leading software makers are aggressively migrating their applications to Linux, as Linux is critical to the evolution of e-business. Transcoding vendors need to offer support for Linux to allow developers to tie Linux into their enterprises and grow this platform wirelessly into the new e-business landscape.

jca@us.ibm.com

amannar@us.ibm.com

FOCUS

JAVA & LINUX 10 FEET TALL AND BULLET PROOF

BOARD THE HIGH-SPEED ROCKET TO THE FUTURE

WRITTEN BY BEN OKOPNIK

Linux from its inception was written by programmers for programmers. In the years since, the GUI interfaces and other user-friendly items have raised the warm and fuzzy quotient to make Linux accessible to the casual user. However, the core idea remains: provide maximum support and usability to the people who make the software happen. That commitment continues; after all, the people who make Linux grow, the ones responsible for its tremendous success, are almost all programmers – techies! – and as such, are interested in better tools first; the nifty toys to be built with those tools are "a trivial excercise left to the student."

7. Note on Java support. This software product contains support for programs written in Java. Java technology is not fault tolerant and is not designed, manufactured, or intended for use or resale as on-line control equipment in hazardous environments requiring fail-safe performance, such as in the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, direct life support machines, or weapons systems, in which the failure of Java technology could lead directly to death, personal injury, or severe physical or environmental damage.

-- *[Excerpt from the Windows NT license agreement (no copyright notice)]*

"We use Linux for all our mission-critical applications. Having the source code means that we are not held hostage by anyone's support department.

—Russell Nelson, President of Crynwr Software

How do you power off this machine?

—Linus Torvalds, after using the linux.cs.helsinki.fi machine for several months"

FUD (Fear, Uncertainty, and Doubt) tactics are a negative decision factor. It tells you what to stay away from, not what to go toward. How's this for positive info: the Linux kernel has built-in Java support that lets you run your executables and applets right from the command line; it's Java-friendly from the start.

Given its nonhierarchical "bazaar" method of software development and the fact that it's a result of a cooperative effort, Linux has always been a haven for programmers and developers. Those of you used to MS Windows may find that the Linux tools are perhaps not quite as "slick"; however, you'll discover much greater flexibility and power as well as finer-grained control over the development environment. A number of benefits are inherent in the OS itself: I've become rather attached to the idea that computers are indeed reliable tools that don't crash and have uptimes measured in years instead of hours.

If you like the idea of development unrestricted by proprietary licenses – Free or Open Source software is the keystone of Linux/GNU – you'll also have the option of compiling your code without using Sun's Java Development Kit (though the JDK is also available). Linux is all about choice: you can use what's available (a very broad range), modify an existing program (and perhaps release the new version to the Linux community), or write your own (the number of compilers and interpreters that come with the average Linux distribution is nothing short of astonishing).

In addition to the software being Free, almost all of it is also *free*: if you wanted to (and had the patience or the bandwidth) you could connect to, say, ftp.debian.org or ftp.redhat.com, download the entire distribution, and install it. There's nothing to stop you; in fact, that's the very reason for that server's existence. Go ahead – cut CDs from what you download and give a copy to all your friends; you're more than welcome.

Shocking difference in attitude? No threats of doom, gloom, or plagues for copying? No "per seat" licenses? Nobody's even mentioning lawyers?

Welcome to the world of open source software.

More than 90% (my estimate) of Linux/GNU software is open source and noncommercial. Most of the rest is "nonfree" but doesn't require payment (e.g., Netscape Communicator, the JDK); only a small percentage is commercial (and may or may not be open source).

Given the recent tremendous growth of Linux in the marketplace, chances are that one day soon you'll have an opportunity to work with Penguin Java (Linux's logo is a cute little penguin named Tux). Of course, a myriad of questions spring to mind: How difficult will it be? What are the available tools? Is there any support? Is there an IDE?

Full Disclosure

Before we go any further, several items are worth noting. I'm a Linux enthusiast and a computer professional with over 20 years of experience; a fair bit of that time was spent writing code. What I'm not is a professional Java developer or even a skilled Java programmer. The use of large-grained salt is strongly recommended when reading any of my Java-specific comments.

An FYI item for folks unfamiliar with UNIX in the Windows world, command line utilities are deprecated and mostly ignored; in Linux (and UNIX in general) they thrive and flourish. In fact, these tools have been polished to a high gloss by user feedback over the years until they're as close to perfection as possible – you'll be able to do things that are very difficult, if not impossible, in GUIs. For example, a few days ago I had to work with a group of auto-generated HTML files, each of which had a number of links to other HTML files within that group. The problem was that the files themselves didn't have an .html extension – either in the original file names or in the links – which caused at least one browser to read the links as text files rather than HTML. A few seconds of thought and:

```
perl -pi -e s/'<a href="[0-9]*'/'$&.html'/g *
for n in *.[^h]*;do mv $n $n.html;done
```

That's about 10,000 links in around 1,000 files– all fixed in a few seconds using two short command lines. If I had to do it with a GUI... frankly, I'd be lost.

(For the curious among us, in the first line I used Perl ["The Swiss Army Chainsaw"] to loop through every file in the directory and append an .html to the appropriate strings. The second line added the extension to the filenames that didn't already have it.)

One of the "problems" any new Linux user faces is that there are no $\$2x10^8$ ad campaigns with heavenly choirs and dancing hamsters to tell you that this is The One True Software that will do everything you've dreamed of (including washing your dishes and getting you a date with Gillian Anderson). Linux/GNU software, by contrast, is typically produced by individuals or groups who aren't charging any money for it, and thus (oh, horror of horrors in this world of modern marketing!) don't have advertising budgets; even the best software tends to be "advertised" with no more than a two- or three-line description in the package list. Usually, it takes a bit of experimentation – installing the programs and working with them – to decide what you like, what works well for you, and what will enable you to be as effective as possible. It's a very different methodology from that of buying the software and either being stuck with it or having to fight the Battle of Refunds.

Distribution	Pros	Cons
Red Hat	Easy to install, great tech support, very user-friendly	Skimpy administration capabilities, "blinkenlights" design philosophy
Debian	Lots of programs, highly flexible, many configuration options	Somewhat increased complexity, less automated
SuSe	As above, plus a lot of support for unusual hardware	Most of the "interesting" documentation is in German
Slackware	Complete, simple, rugged, great learning platform	Very "manual," simplistic package method

TABLE 1 Some popular Linux distributions

Running Linux requires you to think of software in a different way. My contention is that to a large and rapidly growing number of people – of whom I'm one – it seems to be a better way.

The Various Flavors of Linux

The choice of a distribution seems to be a major and confusing topic for anyone interested in installing Linux. In very broad terms and with a strong dash of personal opinion, here's my 2¢.

The Linux kernel – the part of the OS that provides resource allocation, I/O services, and more – is the same in any distribution for a given architecture (except for the version and compilation options). What varies is the installation procedure, the package method, the number/type of included programs, and the filesystem layout. The most common choices these days are RedHat, Debian, SuSe, Mandrake, and Slackware, with a few companies trying to carve out niche markets (e.g., Caldera's "OpenLinux" claim to fame is complete Novell Netware interoperability). A large number of less well-known distributions are available. Table 1 provides a short list (my familiarity with these ranges from excellent to minimal).

The package method that each distribution uses – the format of the setup files used to install the software, its associated documentation, and more – is somewhat different but convertible. For example, a RedHat Package Manager file (an RPM) converts to a DEB package in a few moments via the "alien" utility and is then installable on a Debian system or vice-versa. Given a standardized method of package installation within a distribution, software uninstallation is a clean and well-defined process without stray files ending up in odd corners.

A handy utility for Debian, by the way, is a program called *apt-get* (RedHat's equivalent is *rpmfind*). If you know the name of the package you want to install (say, the Java Development Kit), simply type:

```
apt-get install jdk1.1-dev
```

apt-get will query the previously specified archive servers, then download and install the latest version of the software you've requested, as well as any other software that's a prerequisite for it. By using other switches of apt-get, such as "update" and "dist-upgrade," you can check for newly released packages or even update your entire system at once.

Which distribution should you get? Like many other Zen questions this one has no Right Answer.

Myself when young did eagerly frequent Doctor and Saint, and heard great Argument About it and about: but evermore Came out by the same Door as in I went.
– The Rubaiyat of Omar Khayyam, XXVII, tr. by Edward Fitzgerald

If you're going to take the time to configure your system to your personal preferences – who among us doesn't? – and given the fact that almost any package can be converted, the differences begin to fade anyway.

The Iron Hand in the Velvet Glove

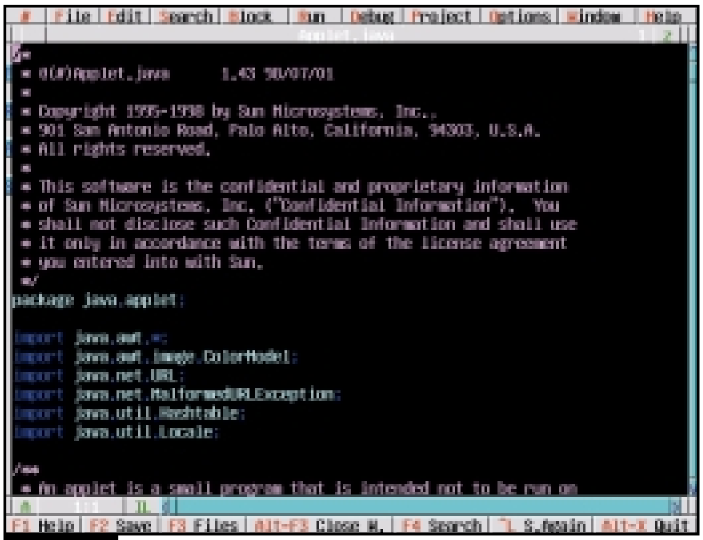
Most Linux distributions default to a GUI upon installation. This gives you a comfortable familiar interface. But what if you need to perform some complex file operation (as I did, above) or run some app that prints to the console? An application called *xterm* (X Terminal) is installed with the X Window system; newer, better versions such as rxvt and gnome-terminal also exist. All of these resemble a DOS window but provide infinitely greater functionality. Bash, the default shell (other command language interpreters are available), has a long list of quite respectable capabilities: subshell support, a powerful scripting language, editable command history, full multitasking, process control, and many others. That plain-looking prompt isn't all that plain anymore.

For those of you familiar with DOS's Norton Commander, a pleasant surprise: Linux has a program that took off on that theme and ran far and fast. Midnight Commander by Miguel de Icaza incorporates things like FTP, virtual file systems, graphical background process control, and many other features. MC is a file manager that runs in an xterm window (a GUI version called *gmc* is the default file manager under RedHat). File operations such as copying, moving, deleting, compressed file viewing, and package installation are handled by a single key press (Norton fans, these are the same keys you're used to!), and you can easily configure MC to perform complex operations based on the file type and the desired action (opening, viewing, or editing).

Setting Up the Workshop

These are the packages I installed for this project (a number of others are related to Java but these are a good start):

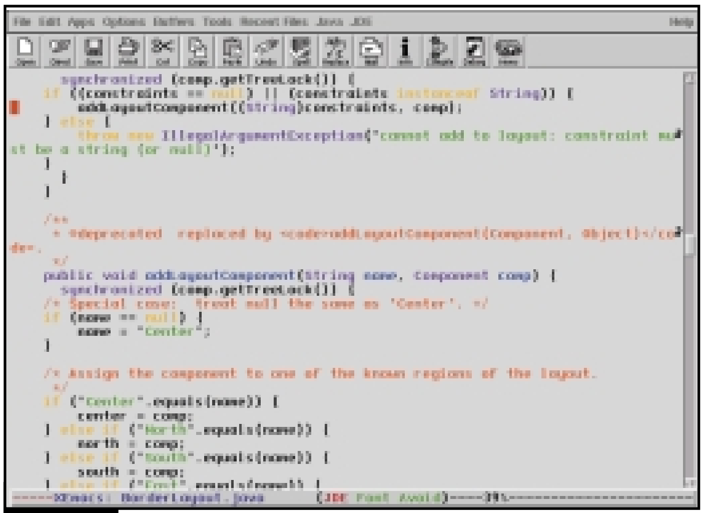
```
//jdk1.1_1.1.7v1a-2.deb      Runtime part of the JDK (JRE)
jdk1.1-dev_1.1.7v1a-2.deb   Java Development Kit
biss-awt_0.87-1.deb         Java AWT
tya_1.1v3-3.deb    JIT compiler (a JVM extension)
guavac_1.0-5.deb   Open Source Java compiler
xwpe_1.4.2-2.1.deb "Turbo-C"-like programmer's editor
jde_2.1.1-4.deb    Java IDE for the Emacs editor
```



# File Edit Search Block Run Debug Project Options Window Help
Applet.java
F1 Help F2 Save F3 Files Alt-F3 Close W. F4 Search ^L S.Again Alt-X Quit

```
/*
 * @(#)Applet.java      1.43 98/07/01
 *
 * Copyright 1995-1998 by Sun Microsystems, Inc.,
 * 901 San Antonio Road, Palo Alto, California, 94303, U.S.A.
 * All rights reserved.
 *
 * This software is the confidential and proprietary information
 * of Sun Microsystems, Inc. ("Confidential Information").  You
 * shall not disclose such Confidential Information and shall use
 * it only in accordance with the terms of the license agreement
 * you entered into with Sun.
 */
package java.applet;

import java.awt.*;
import java.awt.image.ColorModel;
import java.net.URL;
import java.net.MalformedURLException;
import java.util.Hashtable;
import java.util.Locale;

/**
 * An applet is a small program that is intended not to be run on
```

FIGURE 1 The X Window Programming Environment



```
    synchronized (comp.getTreeLock()) {
  if ((constraints == null) || (constraints instanceof String)) {
      addLayoutComponent((String)constraints, comp);
  } else {
      throw new IllegalArgumentException("cannot add to layout: constraint mu
st be a string (or null)");
  }
    }
  }

  /**
   * @deprecated  replaced by <code>addLayoutComponent(Component, Object)</co
de>.
   */
  public void addLayoutComponent(String name, Component comp) {
    synchronized (comp.getTreeLock()) {
  /* Special case:  treat null the same as "Center". */
  if (name == null) {
      name = "Center";
  }

  /* Assign the component to one of the known regions of the layout.
   */
  if ("Center".equals(name)) {
      center = comp;
  } else if ("North".equals(name)) {
      north = comp;
  } else if ("South".equals(name)) {
      south = comp;
  } else if ("East".equals(name)) {
-----XEmacs: BorderLayout.java        (JDE Font Avoid)----78%-------------------
```

FIGURE 2 The Kitchen Sink Editor

The first two packages comprise Sun's JDK. The good folks at www.blackdown.org, the ones who are doing the JDK for Linux conversions, are up to version 1.30, and the latest news and downloads are available right from their site. Note that the JDK is released under the Sun Microsystems' license and thus is nonfree software. You won't be asked to pay for it, but anything you create by using those tools is encumbered by a number of legal restrictions (this is true no matter what OS you use; the license applies to the tools). Under Linux, at least, you have the option of avoiding that by using the Open Source compiler (e.g., guavac) and the JDK tools for, say, debugging.

Once we have a compiler and a debugger, we need some type of an IDE. The X Window Programming Environment (xwpe) is a Turbo-C–like editor/IDE (see Figure 1); for those who like those old-fashioned tools (okay, I'll admit it – call it nostalgia!), it's a well-written piece of software that will run either from the console (in text mode) or under the X Window system. The environment is easily configurable for most of the common languages including Java. Small, simple, and functional – if that's your preference, then xwpe is the way to go.

On the other hand, if you like software that can do everything and then some, FSF GNU Emacs and its GUI cousin, XEmacs, are the unbeatable choice. With the addition of the Java Development Environment (JDE) package, above, Emacs becomes a full-blown and highly configurable Java IDE (see Figure 2).

Some people prefer vi, another (reputedly) powerful and highly customizable editor. For myself, well, even the thought causes me to hunt for strong beverages and turn a slight shade of purple. The first time I tried vi it took me an uncomfortable number of minutes to figure out how to exit the thing without resorting to kill. Even after a close study of its manual page I avoid it when I can… but I'm willing to accept that there are people who love it.

When experimenting with the IDE/compiler combinations, I used my slowest machine – a 166MHz 16MB laptop – to get distinct measurements. I found that guavac was significantly faster than javac, while the choice of interface made little difference (less than one second in the worst case). As a baseline (i.e., when executed right from the command line), a simple "HelloWorld.java" took 19.33 seconds to compile with javac the first time versus 5.34 seconds for guavac. Once caching was in effect (subsequent runs), javac averaged 11.42 seconds, guavac 3.95 seconds. Compiling larger projects (I downloaded some code from the Web) provided comparable results.

"I'd rather spend 10 hours reading someone else's source code than 10 minutes listening to Muzak waiting for technical support which isn't."

—Dr. Greg Wettstein, Roger Maris Cancer Center

A Supporting Cast of Thousands

One of the best things about Linux is the amount of information available on any topic of interest. The Linux community is a Net-enabled phenomenon in which the watchwords are *cooperation* and *sharing*.

The open source idea can be an excellent business model. There's a site (www.cosource.com) where you can browse for open source software contracts, see the amount that people are willing to pay for the creation of that software (the bounty is an aggregate of all the interested sponsors' contributions), and decide if the money offered is worth your time. It's a new method of creating customer/developer relationships and perhaps a chance for an unknown hotshot programmer to make his or her reputation. Say, could that famous person be you?

State of the Union

Today, Linux is a mature, stable, powerful OS that has penetrated deeply into the business and home desktop market. Companies as diverse as IBM, Oracle, Hewlett-Packard, Compaq, Corel, and many others have joined the flood of supporters of open source. Linux has controlled experiments in outer space, been built into watches and car audio equipment, runs the world's smallest Web server at Stanford and the largest clustering network in Australia, and has been nominated by IBM to run on their biggest machines. The plans for the future include exciting technologies that are being implemented even as you read this article. If you have a brilliant idea that may bring the world to your door, there's nothing to stop you from developing it and an entire community to help.

The high-speed rocket to the future has arrived and you're welcome aboard.

References

1. *"The Cathedral and the Bazaar"* by Eric S. Raymond: www.tuxedo.com/~esr/writings/cathedral-bazaar
2. *"GNU General Public License"*: www.gnu.org/copyleft/gpl.html. Also Richard Stallman's "GNU Manifesto" gives a broader view of the original principles of the Free Software movement.

Resources

- *General Linux Information:* www.linux.org and www.linuxos.org
- *Latest Apps and Discussions:* www.linuxlinks.com, www.linuxapps.com, www.linuxberg.com, and www. linuxresources.com
- *Answers and Problem Solving:* www.linuxdoc.org, www.linux-howto.com, www.linuxhq.org, www.linuxsupportline.com, and www.linuxcare.com
- *Online Publication:* www.linuxgazette.com
- *Linux Forum:* www.slashdot.com

Author Bio

Ben Okopnik is a contributing editor to the Linux Gazette. He's been playing with computers since the late '70s and working in the field since the early '80s.

ben-fuzzybear@yahoo.com

Dialog with Data

Insight into the JDBC architecture and using it to build Java database applications

WRITTEN BY
ROBERT J. BRUNNER

Since this issue of *JDJ* is devoted to Linux, it's only appropriate to focus on architectural issues. In our case this means examining the various software architectures that use databases in Java applications as well as some details of the low-level operations that occur between a database, a JDBC driver, and a Java application.

At a fundamental level, Java-based database applications can come in two flavors: two-tiered or three-tiered. The tiers don't indicate how many servers or components are involved; instead, they represent conceptual levels into which the different parts of the overall application are categorized. In fact, a given level may have a single component (e.g., the corporate database), several components (e.g., replicated Web servers), or a million components (e.g., a Java applet running in a Web browser). Each of the two alternative architectures has specific advantages and disadvantages and hence specific application areas in which they excel.

Two-tier architectures (sometimes referred to as client/server frameworks) became somewhat prevalent with the growth of corporate LANs and WANs. In this architecture a server (such as the main database server) interacts directly with multiple clients (see Figure 1). The user interface can be an applet, a graphical application, or a command-line application. This approach has several advantages, in particular a very tight coupling between the client (which is running in its own JVM) and the server. As a result, applications can be prototyped rapidly since everything, including a complete understanding of the hardware and software system, is "in-house." In addition, these systems can use specific performance tweaks in the user interface since the client has explicit knowledge of the server (e.g., the UI knows whether it's talking to an Oracle or SQL Server database).

On the other hand, this tight coupling results in disadvantages that actually limit the applicability of the architecture, especially with the growth of Internet- and Web-enabled technologies. To see

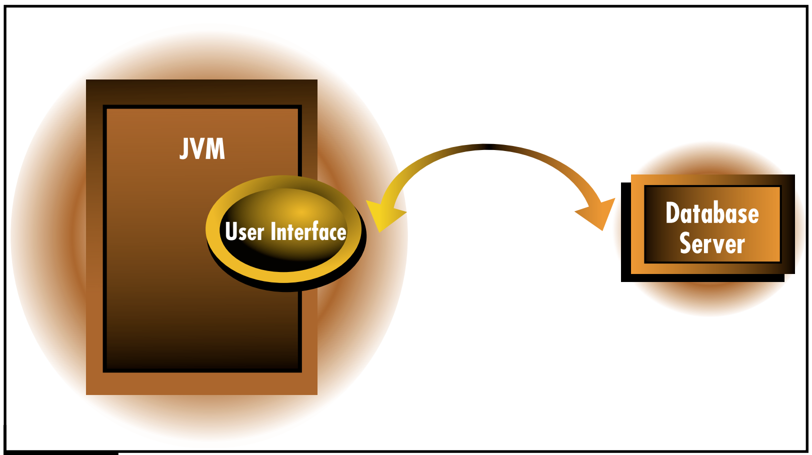


FIGURE 1 Demonstration of the two-tier architecture

why, consider the implications of providing access to your corporate database via the Web – with everything from Denial of Service (DOS) attacks to crackers altering payroll records suddenly becoming possible. The simplicity of reengineering code, or even monitoring outgoing network traffic, makes these scenarios very real possibilities as soon as client software with intimate knowledge of secure servers is released.

Another important disadvantage of the tight coupling is system maintenance. Since the user interface is tightly coupled to the server, changes in the server (e.g., changing the database engine) break the entire application. Finally, this model is verboten when it comes to applet-database systems, since the applet can only communicate directly with the server from whence it came (i.e., the Web server). Even if they're running on the same hardware system, the database will by necessity be running on a different port than the Web server, and the Security Manager prevents the applet from communicating directly with the database server. This can be circumvented by explicitly providing this permission for the applet code, which, while possible in a corporate LAN where the operating systems are "ghosted," is impossible over the Internet. (Would you give an anonymous user full control of your computer?)

The alternative architecture is the three-tiered approach. Essentially, this architecture breaks the tight coupling by introducing a middle-layer between the user interface and the server. As a result, the client no longer has intimate knowledge of the server and instead communicates directly with a middle-layer server that handles the communication with the server (see Figure 2). As in the two-tier case, the user interface can be an applet, a graphical application, or a command-line application. Notice the flexibility of this approach, which mimics the famous Model-View-Controller architecture used in GUI applications. The application server hides the details of the database's server and provides caching performance boosts.

At this point, anyone who has considerable experience in C/C++ (i.e., experience using pointers) will surely question the performance of this approach, which has essentially added an extra level of indirection. By using resource caching, however, this approach can actually provide performance improvements as well as simplify security restrictions (e.g., the applet can now communicate with a servlet running in the original Web server). Finally, this approach can also easily provide fault tolerance and replication capabilities through the addition of more components in the middle layer.

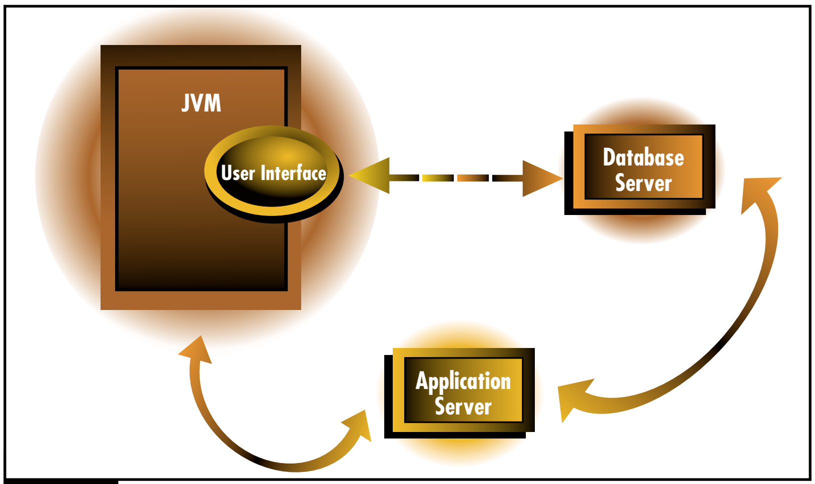


FIGURE 2 Demonstration of the three-tier architecture

Transaction Action

When integrating databases into Java applications, the insulation that Java provides often obscures important database concepts familiar to database developers. Primary among these is the concept of a transaction. Simply put, a transaction is a logical unit of work used by databases to record a user's interactions with a database. If everything completes successfully, a user will commit any resulting changes so that they're persisted. On the other hand, if a problem arises, a user can roll back the database to the state prior to the start of the transaction. While this is easily visualized in a single-user mode, real systems need to support multiple users concurrently. As expected, this complicates the database interactions since different users can be accessing or modifying the same data simultaneously, which can produce dirty, nonrepeatable, or phantom reads (see Table 1). Preventing these conditions requires controlling or limiting access to the data, formally known as *locking*. Different database systems provide different levels, or granularity, of locking, which can significantly impact the performance of an application. (Database locking is remarkably similar to thread-level locking.)

Fortunately for us, JDBC allows a developer to enable or disable autocommit mode, commit or roll back a transaction, and use the ability of a database system to control the isolation (i.e., locking) level of a transaction. These abilities are all provided as methods in the connection interface. Remember that when using JDBC, by default, a database connection operates in autocommit mode; that is, every statement sent to the database is in its own transaction. If you don't want this feature enabled, you need to explicitly disable it.

Dirty Read	Occurs when the results in one transaction are modified by another, uncommitted transaction, which is later rolled back. The first transaction, therefore, contains "dirty" data.
Nonrepeatable Read	Occurs when one transaction retrieves data, another transaction alters this data, and the original transaction retrieves the data a second time. The first transaction, therefore, is unable to repeatedly read the same data.
Phantom Read	Occurs when one transaction retrieves data, another transaction inserts or deletes more data, and the original transaction retrieves the data a second time. The first transaction, therefore, obtains "phantom" data (i.e., a different number of rows in the result set).

TABLE 1 An overview of the common pitfalls that can arise in concurrent, multiuser systems

```
try{

 // Disable autocommit mode
 con.setAutoCommit(false) ;
 // Do some SQL processing
 // If things worked we can commit
 con.commit() ;
 // Otherwise we need to abort
 con.rollback() ;

}catch(SQLException e){
// Process results.
```

Another database concept, batch processing, has been enabled with the release of the JDBC 2.0 API. This feature allows a developer to instruct a database to process multiple SQL data manipulation language statements (i.e., execute a data update) in a single batch process, which provides an additional performance boost. To utilize this feature, you need to turn off autocommit mode:

```
try{

 // Disable autocommit mode
 con.setAutoCommit(false) ;

 stmt = con.createStatement() ;
```

build your batch SQL command:

```
stmt.addBatch("INSERT INTO Contacts
Values('Bill', 'A','Gates')") ;

 stmt.addBatch("INSERT INTO Contacts
 Values('Scott', 'A','McNealy')") ;
```

after which we can execute our batch statement.

```
 int[] counts = stmt.executeBatch() ;

 con.commit() ;
```

To handle batch statement processing, a new exception was added: the BatchUpdateException, which will be thrown if either one of the SQL statements in the batch returns a result set (instead of an update count) or if one of the SQL statements in the batch doesn't successfully execute. Of course, you still need to catch the SQLException in case the other JDBC statements encounter an error condition.

```
}catch(BatchUpdateException e){
// Process results.
}
}catch(SQLException e){
// Process results.
}
```

Grasping Your Results

In addition to the transaction nuances mentioned earlier, with the introduction of JDBC 2.0 API a Java developer can also control the type of result sets that a database will return as well as the level of concurrency they must support. To understand this more clearly, picture a Java application communicating with a database via an appropriate JDBC driver. The application executes a query that the driver sends to the database. The database creates a cursor to iterate over the query results (which, depending on the locking, can be sensitive to other transactions) and communicates this to the application via the driver. The complete details of this process are transparent to the application and clearly a lot of performance operations can be implemented, such as caching of results. For drivers that support it the JDBC API actually provides the ability to specify an a priori fetch size as well as a fetch direction (i.e., top-down or bottom-up). For example, the following line tells the current JDBC driver to fetch 20 rows at a time from the database starting from the last row.

```
rs.setFetchSize(20) ;
rs.setFetchDircetion(FETCH_REVERSE) ;
```

Furthermore, the details of the result types are now controlled during the statement creation.

```
stmt = con.createStatement(
 rsType, concurrencyLevel) ;
```

The first argument, rsType, can take three values:

1. TYPE_FORWARD_ONLY, which is the default value, indicates that a result set will utilize a forward-only cursor.
2. TYPE_SCROLL_INSENSITIVE indicates that the result set will use a bidirectional cursor that's insensitive to changes made in the database while it's open.
3. TYPE_SCROLL_SENSITIVE indicates that the result set will use a bidirectional cursor that's sensitive to changes made in the database while it's open. To explicitly refresh the contents of a TYPE_SCROLL_SENSITIVE result set, you can call the refreshRow method:

```
rs.refreshRow() ;
```

The second argument, concurrencyLevel, controls the concurrent access to the results of a query and can take two values: CONCUR_READ_ONLY, which is the default, indicates that the result set cannot be updated, and CONCUR_UPDATEABLE indicates that the result set can be updated. This point might seem confusing, but another new feature introduced in the JDBC 2.0 API is the ability to update a result set directly using Java commands as opposed to dynamically creating and executing a new SQL UPDATE statement. To accomplish this, result sets maintain two cursors: one for the current row, which you can update directly, and the second for a special, "phantom" row that's used to insert new rows. For example, the following code changes the last name column for the current row to Gates and inserts a new row for the author:

```
rs.updateString("Last_Name", "GATES")
;
rs.updateRow() ; // Save the change

rs.moveToInsertRow() ;
rs.updateString("Last_Name", "Brunner") ;
rs.updateString("First_Name",
"Robert") ;
rs.insertRow() ; // Save the change
```

Conclusion

Hopefully this article has provided some more insight into the JDBC architecture and how you can use it to build Java database applications. Now you can start writing code and, as always, if you have any questions, check the JDBC documentation.

rjbrunner@pacbell.net

Author Bio

Robert Brunner is a member of the research staff at California Institute of Technology, where he focuses on very large (multiterabyte) databases, particularly on KDD (Knowledge Discovery in Databases) and advanced indexing techniques. He has used Java and databases for more than three years and has been the Java database instructor for the Java Programming Certificate at California State University, Pomona, for the past two years.

UniqueID Generator: A Pattern for Primary Key Generation

Overcome scalability and flexibility issues where other patterns fall short

WRITTEN BY
JASON WESTRA

Several patterns exist for generating primary keys for your EJB application. This month I'll provide a pattern for generating PKs that's scalable, generic, and portable.

My format for defining the pattern will follow the catalogued layout presented in the Gang of Four book, *Design Patterns: Elements of Reusable Object-Oriented Software*.

Pattern:
UniqueID Generator

Intent:
Generate unique IDs for persistent objects in an EJB application

Also Known As:
PID (Persistent ID) Factory
GUID (Globally UniqueID) Manager

Motivation

Enterprise JavaBeans is a server-side component model that targets the specific business domain of online transaction processing (OLTP) applications. OLTP applications generally have the need to store information persistently. The data records or objects for each transaction require unique identifiers to allow them to be stored and retrieved accurately. Thus there's a need to generate unique identifiers for the data involved in an EJB transaction processing system.

For the purposes of this pattern, I'm assuming a relational database is used as the data store rather than an object database, which would provide its own ID generator. A relational database table typically has a primary key column that's indexed to prevent duplicate IDs, which would lead to data integrity problems. Thus each row stored in the table is uniquely identifiable. For instance, in EJB the return value from the call to myHome.findByPrimaryKey is a single entity bean, not multiple entities!

There are numerous ways to generate unique IDs in a Java application. Let's review some approaches before discussing the UniqueID generator pattern presented in this article.

System.currentTimeMillis()

An easy way to generate unique IDs is to utilize the System.currentTimeMillis() method to get the current time in milliseconds and use it as your ID. Although it's an easy way to start creating applications, this approach has implications across high-volume applications. A high-volume OLTP application may perform several calls to System.currentTimeMillis() at the same time, resulting in the generation of duplicate IDs. Thus the generator must perform some sort of synchronization on ID requests. Typically, this is done with a wrapper object that uses the synchronized modifier to queue threads accessing it.

Next, you ask, "What about synchronization across multiple JVMs?" Certainly a clustered-server, multi-JVM architecture will be the norm for an enterprise application, but a clustered application poses two problems for this approach toward generating unique IDs:

1. The system time on each machine may be different. Thus, in a multi-JVM architecture, calls to System.currentTimeMillis() can lead to duplicate values.
2. The Java synchronize operator doesn't work across multiple JVMs, so even if the system times were equal, simultaneous calls to different machines could still result in duplicate IDs.

EntityBean Key Generator

This approach uses an entity bean to select the next value from a relational database table, which holds the latest value for unique ID generation. An entity bean can encapsulate the SQL, which it executes in a generic fashion by loading in a bean environment property containing a SQL string for each type of database. For instance, in Oracle the dual table should automatically return a sequence value. The SQL statement might look something like this:

```
"select mysequence.nextval from dual"
```

Oracle sequences are a proprietary feature and increment the value automatically for the caller by a specified increment count. Other databases that don't have this feature must increment a next ID field and select it in one call. Stored procedures are a common cure for the need to execute multiple database operations within a single request. To get the same ID generation capabilities from Microsoft SQL Server, a stored procedure could be called that both returns a value and increments it for the next caller. Since the stored procedure performs its work inside a transaction in one database request, it eliminates a "window" of opportunity when simultaneous calls for another ID get the same row and attempt to update the value.

Thus our entity bean key generator executes SQL, which it retrieved generically from its bean environment property and executed on the database. However, there are some drawbacks to this approach:

- Each call to get another ID is a remote method invocation, which can create unwanted chatter depending on where the entity bean is deployed in your system.
- Some might argue that entity beans are for business entities rather than utilitarian functionality like generating IDs for business entities.
- Few application servers provide synchronized caching across a cluster. Thus the ability to cache a set of IDs, improving ID generation by preventing database operations, is negated when the application is clustered.

A more scalable approach to ID generation that provides both local caching and guaranteed unique IDs is a singleton object that hands out IDs from its local cache (see Figure 1). Each time the cache of the singleton object is depleted, it gets a set of IDs representing the next available IDs for the application. The singleton fetches IDs from a stateless session bean that accesses the database in a portable manner, allowing it to interact with any database.

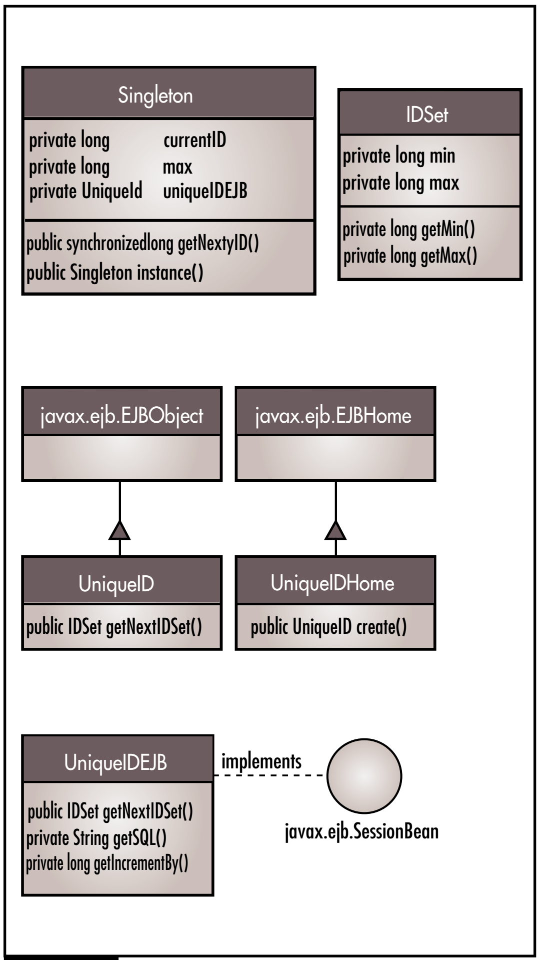


FIGURE 1 Class diagram

Applicability

Use UniqueID Generator:

1. To return a unique identifier for object and/or database row identity
2. To cache sets of IDs across multiple JVMs for scalable ID generation
3. To abstract database implementations from the core ID generation classes

Structure

Participants

- ***Singleton:*** Acts as the wrapper of the UniqueIDEJB and caches sets of unique IDs to prevent chatty remote invocations and database operations
- ***UniqueIDEJB:*** Component made up of UniqueID (remote interface) and UniqueIDHome (home interface); performs generic SQL to fetch a set of IDs from the database and return them to the singleton client

Collaborations

- ***Singleton:*** Calls UniqueIDEJB component to fetch a set of IDs; set is cached locally and a new set is fetched when the maximum is surpassed

Consequences

The UniqueID generator has several key benefits and some limitations:

1. ***Limited remote method invocations and database operations***: The UniqueID generator's singleton object caches a set of IDs that are depleted before another request is made to the UniqueIDEJB. This limits the number of remote method invocations on the EJB and the number of hits the database must handle to generate new IDs.
2. ***Guaranteed UniqueIDs***: The UniqueID generator pattern guarantees unique identifiers for your objects/data across an *n*-tiered solution. If a set of IDs is requested from multiple JVMs at the same time, the transaction isolation of the UniqueIDEJB and/or database layer will automatically queue multiple requests to ensure that only one singleton's request for an ID set is processed at a time.
3. ***Portable across EJB servers***: The pattern represents a component that's portable across EJB servers. It's a session bean, and session beans have been mandatory in the specification since its inception, whereas any EJB server still not up to the 1.1 version of the specification wouldn't be able to utilize an entity bean-based pattern.
4. ***Portable across database servers***: The implementation of the pattern really determines its portability, but the pattern itself allows for database-specific SQL to be loaded generically from the EJB's environment so that no code changes are required to deploy it against disparate databases.
5. ***Singleton knows nothing about the "incrementBy" value***: The singleton object knows nothing about how many IDs are returned in a set. This is controlled at the UniqueIDEJB level by an environment property. To change the incrementBy value, simply redeploy the EJB with a hot-deploy mechanism to avoid downtime and the increment will be changed to the new value without repercussions on the rest of the application.
6. ***ID gaps***: Gaps in IDs will occur with this solution. For example, if a set of 50 IDs is retrieved and the server crashes midway, 25 IDs would be lost. My recommendation is to increment only by one until ready for production, then set the UniqueIDEJB's incrementBy environment property to be sufficient for your application's load. This prevents wasting IDs in development and test.
7. ***Support for other data types***: The structure section's class diagram shows a long data type used for IDs, yet this could be any data type your application requires. However, the UniqueID generator pattern would have to be extended to account for different ID data types in a single application. For instance, if your application mixed doubles, strings, longs, and ints as keys, the singleton would have to be extended.

Implementation

Consider the following issues when implementing a generic, portable UniqueID generator pattern:

- ***Determining incrementBy for your application:*** The amount of IDs contained in each ID set (and subsequently cached in the JVM) should be determined based on the number of users attempting "insert" transactions per JVM. Thus, if your incrementBy is 50 on a single J2EE server, set it to approximately 25 if you cluster two servers. Also, set incrementBy to a low number, even 1, for development and testing. As with any tunable parameter, metrics gathered from stress testing your application should ultimately drive your settings.
- ***Ensure portability across EJB servers:*** When communicating from the UniqueIDEJB to the database, don't use proprietary logic to obtain a connection from a connection pool. Use data sources looked up through JNDI instead.
- ***Ensuring portability across databases***: There are a few ways to ensure that your code to fetch a set of IDs is portable. One way is to store the SQL that will be executed in the EJB's environment. The SQL is set in the deployment descriptor, which allows it to be modified during deployment rather than having to modify the codebase. Another option is to use data access objects (DAOs) to contain the database-specific SQL code. Last, your UniqueIDEJB component couldn't take advantage of database-specific sequence generators. Your component's implementation should create the table at runtime if it doesn't exist already. This approach isn't recommended, but it's an alternative nonetheless.
- ***Generating different IDs for different tables (classes)***: To generate a different ID for different tables, each class in the pattern would have to change its interface to take a "sequence" parameter, which indicates which class you want to get the next ID for and is forwarded throughout the pattern to the JDBC call against the database.
- ***Generating IDs shouldn't rely on a business transaction outcome:*** The UniqueIDEJB should create its own new transaction when getting an ID set. Incrementing the database to the "next ID value" shouldn't rely on whether or not the business transaction currently executing in your application succeeds or fails.
- ***Business transactions rely on generated ID success:*** An error fetching an ID set from the UniqueIDEJB will cause a business transaction to fail. Don't mix different approaches to generating unique IDs when one fails. For instance, let's say your application uses the System.currentTimeMillis() approach to key generation if the first approach, calling UniqueIDEJB, fails. This provides an extra layer of fault tolerance to your application, but you could encounter duplicate IDs! If your database counter is near the number generated by the System.currentTimeMillis(), you'll have problems.

Sample Code

The code in Listing 1 shows how to implement the singleton to cache the ID set, fetch a new set, and handle failures when calling the UniqueIDEJB.

Summary

This month I provided an EJB pattern to a common problem in the OLTP world, generating unique IDs. While there are variations of the pattern, the UniqueID generator overcomes many scalability and flexibility issues where other patterns fall short, such as local ID caching, limiting remote method invocations, and portability. I hope this pattern is helpful for your EJB engagement. For other EJB patterns see www.theServerSide.com. And let me know if you're interested in ***JDJ*** featuring more patterns in the future.

jwestra@vergecorp.com

Listing 1

```
/**
* Singleton exists on a per JVM basis and returns a new, unique ID
* to the caller.
*/
public class Singleton {
    private long max = 0;
    private long currentID = 0;
    private static Singleton singleton = new Singleton();
    private UniqueID uniqueIDEJB;
    /**
    * Accessor to get handle to singleton object.
    */
    public Singleton instance() {
 return singleton;
    }
    /**
    * Cannot instantiate.  Must use instance() method to get a handle
to the object.
    */
    private Singleton() {
 // GET HANDLE TO UNIQUEID EJB
    }

    /**
    * Returns a unique ID to the caller. This is synchronized to pre-
vent
    * multiple callers from getting the same ID back if they called
getNextID() at the
    * same time.
    *
    *
    * @return long                 next ID
    */
    public synchronized long getNextID() throws Exception{
        currentID++;
 if (currentID > max) {
     try{
  IDSet set = uniqueIdEJB.getNextIDSet();

  // reset current id
  currentID = set.getMin();
  max = set.getMax();
     }
     catch(Exception ex) {
  ex.printStackTrace();
  throw ex;
     }
        }
 return currentID;
    }
```

Download the Code!

The code listing for this article can also be located at

www.JavaDevelopersJournal.com

Author Bio

Jason Westra is the CTO of Verge Technologies Group, Inc. (www.vergecorp.com), a Boulder, Colorado-based firm specializing in e-business solutions with Enterprise JavaBeans.

Visual Programming Comes of Age

Make your beans more appealing to a wider market

WRITTEN BY
STEPHANIE PARKIN

Who are you writing beans for? Like most bean developers, you probably think your customer is someone just like you, a Java programmer. You design your beans with yourself in mind, adding features that would help you as a programmer. Seems reasonable. But what if you could reach a wider audience?

Programmers are only a small slice of the computer-user population. If any computer user could pick up your bean and wire it into an application without programming, imagine the increased potential for selling your bean, and for Java programming in general. This article looks first at how visual programming is poised to bring Java to a larger audience, then tells you how to make your beans more appealing to a wider market.

Why Design Your Beans for Visual Programming?

Remember a time before spreadsheets? Well some of us can anyway.... If you wanted a computer to perform a calculation on a set of data (say, a 10% price hike), you got a programmer to write a COBOL program to process your request. When spreadsheets came along, they hid all the complexity from the user, and suddenly anyone could do all sorts of complex calculations. Most people don't consider working with spreadsheets to be programming, yet they're providing all the instructions the computer needs to do what they want it to.

Spreadsheets don't do everything users want. In fact, few programs ever exactly match the end user's specifications because they're designed for a wide audience. So what would happen if users could design their own apps? If they had the right components and could easily assemble them, they might be able to build these apps themselves.

Beans...They're Not Just for Programmers Anymore

Beans (when used with the right visual builders) have the potential to change how applications get created and who creates them. Visual builder tools could tap into a larger customer base, but they need your help. To lure nonprogrammers into the world of Java and visual programming, we need more beans that anyone can pick up and use. To design them, you need to consider a new and different type of user, a nonprogrammer who shies away from writing code.

Most people can follow instructions for wiring a keyboard to a computer, connecting a stereo, or assembling a child's toy. These instructions invariably use diagrams – visual representations of the assembly – rather than textual instructions. So why give computers textual instructions?

Many people can assemble physical components without any instructions at all, because they can see how the parts fit together (see Figure 1). Well-designed components fit together only in ways that work (the plug for the keyboard doesn't fit the socket for the printer), or they provide visual clues for assembly (icons on the plug and socket, or matching colors).

What Makes Java Special?

I risk preaching to the converted here. You know Java is special, not only because it's cross-platform but because of how well beans work as pluggable components. But what you may have forgotten is that Sun's JavaBean component model was specifically designed with visual builders in mind.

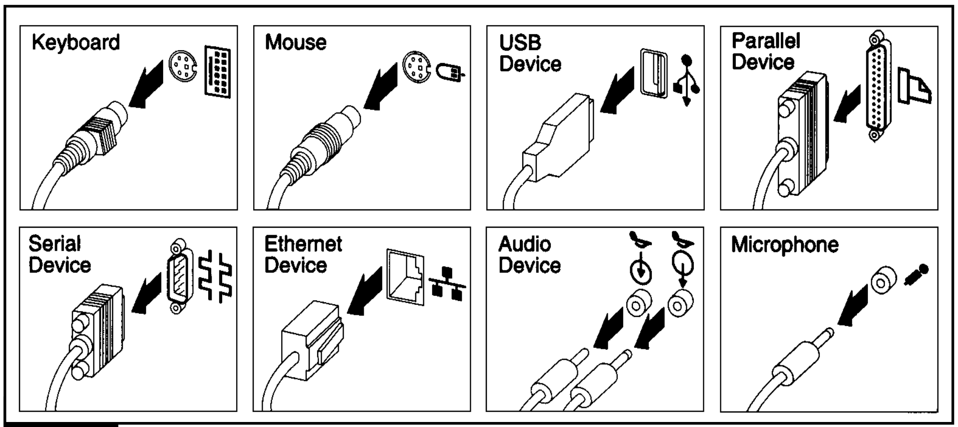


FIGURE 1 Snap-in components

But You Have to *Code* Java...

Still resisting this line of reasoning? You're probably imagining why nonprogrammers can't use beans:

You can do interesting stuff only in methods, not with beans.

Not if your beans are designed right. For example, take some of the complex, yet easy-to-use beans on the IBM alphaWorks (ibm.com/alphaworks) and VisualAge Developer Domain (ibm.com/software/vadd) Web sites:

- ***JMF player***: Video and sound media player that lets you play multiple media clips
- ***Gauges***: Visual devices that display a single value that can vary continually within a range, such as LED displays, counters, and sliders
- ***SmartMarkers***: A palette of color "markers" that highlight patterns and trends in data tables
- ***VisualScheduler***: Project management bean suite that lets you visually schedule a set of tasks
- ***XML beans***: Beans for manipulating XML content, such as viewing, searching, editing, or processing XML documents

If your bean successfully encapsulates a particular task that a lot of people want to do, you can provide the methods for the most common functions. The programmers can add their own methods.

You can't really make beans interact well without code.

Again, the magic is in the design (and in the visual builders). You have to put some thought into how these beans will connect with other beans, and do lots of usability testing. You might also need some helper beans to replace some coding techniques, like a Step bean that lets you specify the order in which events get fired.

Using the visual interface is too limiting.

It doesn't have to be. Since beans were originally intended for visual builders, the JavaBeans specification gives you a good start on how to expose all your bean's functions visually. The JavaBeans Guidelines at VisualAge Developer Domain pick up where the JavaBeans spec leaves off, giving you more tips on how to design your bean.

The tools aren't good enough.

You're right. Many of the tools around today are still designed mainly for programmers. They provide lots of extra function that nonprogrammers don't need. They're improving but we need better visual tools such as debuggers and ways to visually encapsulate function – visual subroutines to help manage the clutter on the workspace. But some current visual composition editors are intuitive enough for nonprogrammers to use. Have you checked out VisualAge for Java lately?

Java is better suited for server-side programming than UI design.

Java is becoming very popular for server-side programming, but JSPs still need beans to encapsulate rich presentation function. JSPs make it more important than ever to encapsulate the complex function and easy customization into beans so that visual designers have all the tools they need to concentrate on visual design rather than coding.

Who Are These People Anyway?

The first thing you need is an introduction to your new audience. Members could be your nontechnical manager who wants a tool to track your project but doesn't want to spend money or programming time to get it. Or your Uncle Fred who wants to connect his computer to his barn's thermostat so it wakes him up when it's too cold for his cows. A COBOL programmer who doesn't know Java but wants to put a Web front-end on his or her legacy application. A Web site designer who needs beans to add dynamic elements to Web pages. Anyone who doesn't know how to code Java.

How Do You Know What They Want?

The thing is, nonprogrammers know what they want their computers to do, but they don't especially know what beans they want or how they should perform. There are two keys to making good beans that will make them more appealing to programmers and nonprogrammers alike:

- ***Encapsulate some complex, really useful function that can be used in many applications:*** The world has enough UI widgets for now in all sorts of styles. Beans need to combine lower-level functions so that the bean really hides complexity from the user. Programmers are good at providing complex function to users but not so good at hiding the complexity. You must carefully analyze what most users will do with your bean and cater to them.

- ***Make sure the bean is easy to use in visual builders:*** This is easier said than done, especially with the wide range of builders on the market. It includes everything from providing good short names and BeanInfo classes to more complex issues, like serialization and customizers.

The real promise of beans is not write once, run anywhere, but write once, reuse everywhere.

Making Your Beans Easy for Nonprogrammers to Use

So what makes beans easy for nonprogrammers to use? In this section we'll run through a bean that plays nicely in visual builders.

- ***What's in a name?*** The most basic usability improvement is good naming. This includes everything from the bean name, to package naming conventions, to names and descriptions of properties and methods. As with good interface design, if you give objects intuitive, simple names, even nonprogrammers can often use them without assistance.

- ***Provide BeanInfo files***: A BeanInfo file is a Java class that implements the BeanInfo interface, named to correspond to the bean it describes. (For example, a BeanInfo file for a LessThan bean would be called "LessThanInfo"). You must provide a BeanInfo file if you don't follow the bean naming rules laid out in Sun's JavaBeans specification. These rules allow bean tools to gather information about your bean using introspection. You should follow the naming conventions and provide BeanInfo files to distinguish between "preferred" and "expert" features and provide property descriptions. VisualAge for Java (and other visual builders) can automatically generate BeanInfo files for you. Nonprogrammers can also use the BeanInfo tab in VisualAge for Java to get valuable information about your bean.

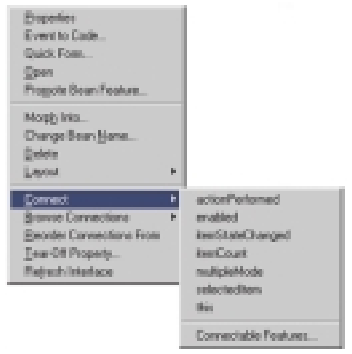
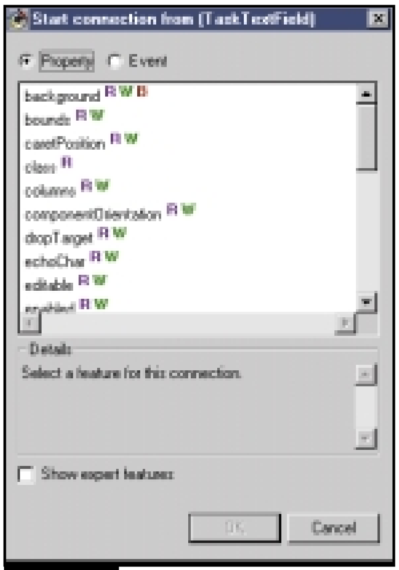


FIGURE 2 Selecting preferred features

- ***Mark commonly used features "preferred":*** Preferred features show up in visual builders at a higher level than nonpreferred ones. For example, in VisualAge for Java, preferred properties



Select a feature for this connection.

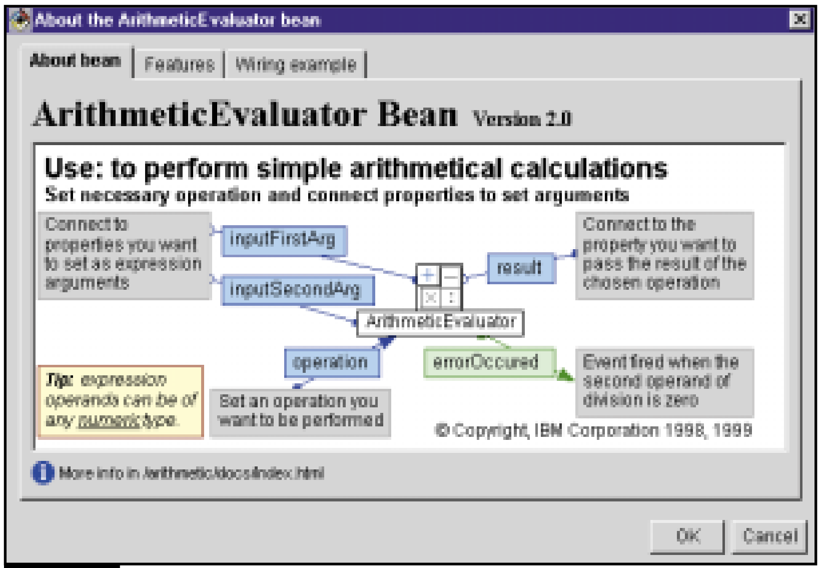
FIGURE 3 Selecting nonpreferred features

appear in the selection menu when wiring an application (see Figure 2). You have to select nonpreferred features from a dialog after selecting the Connectable Features option (see Figure 3).

- ***Use bound properties where appropriate:*** Bound properties notify other components when their value changes. This lets other components do some action based on that changed value. So when nonprogrammers wire bound components together, the components actually perform an action when run. Bound components help nonprogrammers make the bean do its work. For example, a LessThan bean would compare the two values and report the result as soon as it receives two input values rather than waiting for some notification. From the perspective of nonprogrammers, when they wire unbound properties, their bean doesn't do anything when it receives all the necessary data. Bean users have to write their own code to find out when a property changes.

  On the other hand, unbound properties are more efficient than bound ones, so you should make a property bound only when another bean might need to know when that bean changes. In general, input variables don't need to be bound because their value doesn't change by itself (and the end user doesn't change it); it gets changed only by the underlying application.

- ***Be easy to listen to:*** Events generally indicate that a component in a program has reached a certain state or that an externally generated action has taken place. Event listeners are responsible for registering their interest in an event with the source of the event. When the event occurs, the event source informs all listeners, who can then act on the event according to their own needs. When designing your beans, think how nonprogrammers will use the final bean and program in any events they'll probably need. For example, a bean that performs file I/O would fire events at key points, such as after it reads a new line or when an I/O error occurs. Nonprogrammers can then wire appropriate responses to these events in their apps, like issuing a message when an error occurs.



ArithmeticEvaluator Bean Version 2.0

Use: to perform simple arithmetical calculations

Set necessary operation and connect properties to set arguments

Connect to properties you want to set as expression arguments

Connect to the property you want to pass the result of the chosen operation

Tip: expression operands can be of any numeric type.

Set an operation you want to be performed

Event fired when the second operand of division is zero

© Copyright, IBM Corporation 1998, 1999

More info in /arithmetic/docs/index.html

FIGURE 4 Visual help for beans

- ***Provide state-of-the-art visual design and visual customization:*** Users will buy beans only if the visual design is professional and matches the look and feel of the rest of their system. In most cases this means using Swing for your visual elements and taking advantage of its look-and-feel customization.

  Try to get a professional graphic designer involved; many powerful beans today suffer from a poor visual design. For an example of combining powerful visual design with functionality, check out the SmartMarker and Gauge beans at the VisualAge Developer Domain Web site.

- ***Give in-your-face documentation:*** The last thing a power programmer wants to think about is documentation, yet your bean will never sell if no one can figure out how to use it. With today's visual builders you needn't limit yourself to narrative how-tos and Javadocs. Try to provide creative, yet simple help for your beans, such as the "About this Bean" property used on IBM alphaWorks Web site. You can provide visual help for using your beans (including wiring examples) right in the property editor for the bean. For example, Figure 4 shows visual help for a LessThan bean. (Once again, enlist a professional graphic designer to help out.)

- ***Use common design patterns:*** Design patterns are another tool to help nonprogrammers learn how to use beans quickly. They also help other component developers understand and maintain your code. Design patterns are templates for creating applications. They provide a set of rules and naming conventions that help you create and maintain your applications. If you're developing a set of related beans, you should provide a common interface for them to make it easier for your customers to understand the structure and usage of your beans and easier for you (or someone else) to upgrade them. Design patterns provide a way for you to describe this structure and usage.

- ***Create your own customizers and property editors:*** Really useful beans with complex functionality often have complex property sheets. You can help nonprogrammers navigate property sets by providing property editors and bean customizers. A property editor is a tool for customizing a particular property type. You should consider creating your own property editor when you have a property that doesn't fit the standard editors (such as a Swing look-and-feel editor). On the other hand, use customizers when property editors seem too primitive to customize a particular bean. A property editor is associated with a property; a customizer is associated with a bean. Customizers provide more advanced customization of the bean and let you completely replace the IDE's standard property sheet for that bean.

- ***Cope well with errors:*** A bean that plays nicely in visual builders also does rigorous error handling. Nonprogrammers don't know much about handling errors, so provide cues for error handling in your bean's features. To appeal to nonprogrammers, make information about all possible error conditions available in ways that are easy to program visually.

  You can provide information about error events either through exception events or properties. In general, properties are easier for nonprogrammers to work with, but they're not always suitable for all beans (for example, when the property can't access information about an event). You need to carefully consider the situation where errors might arise in your beans before designing your error-handling techniques.
- ***Deliver the whole package***: An article on selling beans wouldn't be complete without a word on packaging. You should follow standards for Java package-naming guidelines, Javadoc formats, and icons. One last task for your visual designer is to create a recognizable, snazzy image that represents your bean (no easy job!). VisualAge for Java helps with this process by providing wizards that walk you through the export process, and version control to help you keep track of your packages.

Where to Learn More

These tips for writing beans for visual programming are expanded on in a set of tutorials on IBM's VisualAge Developer Domain site. These tutorials walk you through creating beans in VisualAge for Java, and along the way teach techniques for making your beans easy to use in visual builders.

The growing popularity of online bean brokerages shows there's a market to sell your beans. We have the right tools, the right technology, and the right markets in place to greatly expand the audience for the beans we develop. What we don't have are the broad scope of beans needed to really develop apps without programming.... What useful bean is lurking in your grounds?

References

1. *JavaBeans Guidelines:* www.ibm.com/software/vadd/Data/Document2398?OpenDocument&p=1&BCT=1&Footer=1
2. *Visual Programming with Beans tutorials:* www.ibm.com/software/vadd/Data/Document3328?OpenDocument&p=1
3. *Designing Beans for Visual Programming tutorials:* www.ibm.com/software/vadd/Data/Document3436?OpenDocument&p=1
4. *alphaWorks:* www.ibm.com/alphaWorks
5. *VisualAge Developer Domain, complimentary downloads of VisualAge for Java, Entry Edition:* www.ibm.com/software/vadd/

Acknowledgments

Many thanks to Dave Pullin for the initial idea, and Tatjana Apine, Colette Burrus, Tatjana Lukovnikova, Juris Krikis, and Pavels Nikolajevs for all their hard work on the visual programming tutorials.

Author Bio

Stephanie Parkin is the managing editor of IBM's VisualAge Developer Domain and WebSphere Developer Domain Web sites. She coauthored the visual programming tutorial series, and is currently turning the series into a book.

stephp@us.ibm.com

A Spoonful of TopLink Helps the Medicine Go Down

WebGain's TopLink speeds deployment up to 75% at Glaxo Wellcome

WRITTEN BY
GABE ALLAN

If there's one industry that challenges the enterprise Internet for growth and excitement, it's medicine and biotech. The pace of medical research and innovation rivals that of any dot-com startup or enterprise computing gambit.

At Glaxo Wellcome Inc., the giant pharmaceutical firm in Research Triangle Park, North Carolina, the latest technologies – Web, enterprise, and database – are being used to accelerate the company's pace of medical and biotech research.

Glaxo's aim is to gain competitive advantage over its rivals worldwide, and drive newer, safer, and more effective therapies and outright cures to the public...faster. Now helping the company's researchers meet that challenge: WebGain's TopLink for Java.

WebGain's TopLink is just one part of a larger company effort to integrate new, pure object applications and solutions with multiple Oracle8*i* SQL databases that house years of Glaxo's intellectual property – product, research, and drug discovery data.

The other 99 parts: the company's team of object-oriented programmers now working in Java. Their task is to integrate Glaxo's crucial relational database foundation with pure-object frameworks and applications architected in-house.

Integrating pure-object applications with relational database architectures is normally the programming equivalent of putting a size 12 foot into a size 10 sneaker: it's a tight, forced fit that's painful to work with.

This poor fit between pure OOP (object-oriented programming) languages such as Java, and relational database lexicons such as SQL, is called the "object/relational impedance mismatch." It's a classic conundrum that has challenged the best OOP programmers.

Using object/relational mapping, TopLink erases the impedance mismatch that's long slowed development of object-oriented relational database systems. And according to one of Glaxo's leading Java developers, that's not mere marketing talk.

Accelerate Time to Market

Roger Cornejo, systems manager for Glaxo Wellcome, says Glaxo's Java developers have eased TopLink into the company's development technology mix and, in his experience, have accelerated time-to-market of new object/relational Java solutions by as much as 75%.

"You can do the work in about a quarter of the time it would take if you had to code it manually, without using TopLink," says Cornejo.

Glaxo Wellcome's research is centered around Research Triangle Park. With work focusing on cancer and viral and metabolic diseases (HIV, asthma, sexually transmitted disease, migraine, etc.), the research information system handles dozens of projects for Glaxo Wellcome and provides IT support for the organization's 2,500 research scientists.

One of the most successful projects has been Glaxo's Targets Knowledge Base (TKB), a three-year-old effort to help scientists find, collect, organize, and distribute information related to early research phases of drug discoveries.

"When [research scientists] have new targets, they enter them into the system," says Cornejo. "They also enter related information to the target that will help other scientists understand them."

That information is now shared through an enterprise Java application distributed over the Web. Building the database, and making it easy to access for the Glaxo Wellcome researchers involved mapping complex Java objects to a nonobject relational database.

"The company has a large investment in relational database technology, and a huge amount of research data around the world that it wants to share," says Dennis Leung, director of product management for WebGain's TopLink.

The IT department's expertise is "in building applications that add value to Glaxo Wellcome," says Leung. "They don't want to have to worry about infrastructure or details; they want their developers to focus on rapidly building applications." That's where TopLink fits in.

Meeting the Market Need

OOP may be the flaming passion for Java professionals, but relational databases remain the order of the day. "People have a massive investment in relational technology," Leung says. "They feel comfortable with it, and they want to leverage their investment."

Joseph Feiman, research director for the GartnerGroup, in Stamford, Connecticut, agrees. "Object databases tried and failed to knock off SQL database environments," he says. "If you're building a distributed object architecture, it's still likely that you'll do it with a relational database on the data tier."

But merging the object and professional worlds has never been easy. The mindsets are different, and the programming skills required to master both are often hard to come by. Now, with TopLink for Java, it's actually easy for Enterprise Java developers to merge the OOP and relational worlds.

"The vast majority of developers haven't discovered the benefits of O/R mapping," says Gartner's Feiman. "They're still using manual techniques or wizards to map directly, but that's inelegant. It's the disciplined developers interested in quality and longevity that are using products like TopLink."

Shopping for Solutions

There weren't a lot of choices when Glaxo began looking for an easier way to map Java objects to relational database tables in early '97. "Three years ago, you had to develop the persistence framework yourself, and make it generic and reusable," says Cornejo. "It wasn't for the faint of heart."

Though not faint of heart, Glaxo's Java team decided that building their own O/R mapping layer by hand didn't fit the company's plans. "We didn't want to write our own persistence layer," he says. "We wanted something that was essentially transparent to the bulk of our developers, and allowed us to lose some of the complexity the relational model brings."

Ideally, Cornejo says, the Glaxo developers would focus solely on the object model while developing. "We knew of TopLink," he says. "We decided to bring it in, and give it a whirl."

Glaxo's team began working with beta versions of TopLink for Java and Oracle's Oracle8*i* database, assured by both vendors that production versions of their products would be shipping before Glaxo was ready to deploy.

"They were definitely trailblazers, on both the O/R mapping and Oracle8*i* fronts," says WebGain's Leung. "Even those early releases allowed developers to easily store Enterprise JavaBeans and business objects in relational databases. It's a nonintrusive, metadata approach, similar to object database semantics in the way developers deal with things at the object level."

Glaxo's Cornejo agrees: "Having early access to both TopLink and Oracle8*i* allowed us to run our application in a number of advanced architectural configurations," he says. "We used TopLink's three-tier support, and Oracle8*i*'s CORBA capabilities, to build a three-tier Web application that runs on the Oracle8*i* [Java] VM."

WebGain's TopLink turned out to be an ideal solution that helped Glaxo's Java developers abstract the thorny relational database model away and maintain an all-Java, pure-object view of the application logic and business rules.

Building the System

"I've used the product to map approximately 60 domain objects to approximately 50 relational tables," says Cornejo. "We chose to share the data at the relational database level, and present that data in the HTML format. TopLink has dramatically reduced the amount of time that would have been required to make all this work."

Cornejo says TopLink's support for inheritance enabled them to apply different mapping strategies for different objects. Persistent objects were used to hold a lot of data, and were mapped to relational tables using one-to-one, one-to-many, and many-to-many relationships.

The evidence inheritance hierarchy was mapped by TopLink without affecting an application-level code or requiring a recompilation of any classes. Ultimately, Glaxo's then-beginning Java developers formally launched the TKB project.

All with a little help from TopLink. "When I started working with TopLink, I had about two months of Java experience," Cornejo says with a chuckle. "In fact the level of Java experience on the team was pretty low. Now we're building Java objects exclusively...and I'm afraid my SQL could get rusty!"

Since the initial deployment, Glaxo's Java team has architected six new releases, all of which transparently leverage O/R relationships. The company's scientists say the project has increased the efficiency of their leading research and is bringing product to market faster.

While WebGain officials say TopLink can deliver 30–40% productivity gains, Glaxo's Cornejon says it's even higher. "I think you can probably do the work in about a quarter of the time it would take if you had to code it yourself," he says.

Looking to the Future

The Glaxo Wellcome TKB project is now stable, but the company continues to use TopLink in its development efforts.

"The idea is to help scientists take new molecules further into discovery," Cornejon says. "Take a hit on a new molecule and progress it into a lead. There's no question that TopLink is helping us get there faster than we could have before."

Author Bio

Gabe Allan is a technology journalist based in New York who writes for many leading technology publications.

gabeallan2000@hotmail.com

MM.MySQL and the MySQL Database System

Flexible tools for developing Web-based applications

WRITTEN BY
MARK MATTHEWS

MySQL, and MM.MySQL, a JDBC driver for MySQL, are open-source tools that allow Java developers on Linux and other UNIX-like systems to develop full-featured applications. Both have similar histories – they started out as internal tools to fix a particular problem and grew to be much larger projects than the developers originally intended.

I started developing MM.MySQL in May of 1998, to learn socket and JDBC development and to fix some problems with the original JDBC driver for MySQL, GWE. GWE development had ceased at that point and some features I required in a JDBC driver for MySQL weren't present.

Since that time MM.MySQL has developed into a fairly capable Type IV (all Java) JDBC driver for MySQL. Depending on the JVM you're using, it supports the JDBC-1.2 or JDBC-2.0 API. It's been used with a majority of open-source Java tools like JBoss, Enhydra, and Tomcat, as well as Cocoon and Turbine. It's also supported by popular development environments such as Forté, JBuilder, IBM VisualAge for Java, and Macromedia UltraDev.

The MySQL Database System

MySQL is an open-source relational database management system developed by MySQL-AB. It's fast, reliable, and easy-to-use with a full client/server protocol. MySQL also has a very practical set of features developed in close cooperation with MySQL's users. Originally developed to handle large databases much faster than existing solutions, it's been successfully used in highly demanding production environments for several years. Though under constant development, MySQL offers a rich and useful set of features such as:

- Full multithreaded operation using operating system–level threads and SMP if available and supported by the operating system
- C, C++, Eiffel, Java, Perl, PHP, Python, and Tcl APIs
- The server runs on many different platforms, including Windows, Linux, FreeBSD, and Solaris, as well as many common UNIX variants
- Rich data-type support, including signed/unsigned integers 1, 2, 3, 4, and 8 bytes long, FLOAT, DOUBLE, CHAR, VARCHAR, TEXT, BLOB, DATE, TIME, DATETIME, TIMESTAMP, YEAR, SET, and ENUM types
- Full support for SQL GROUP BY and ORDER BY clauses. Support for group functions (COUNT(), COUNT(DISTINCT), AVG(), STD(), SUM(), MAX(), and MIN())
- Support for LEFT OUTER JOIN with ANSI SQL and ODBC syntax
- Ability to mix tables from different databases in the same query (as of Version 3.22)
- A privilege and password system that's flexible and secure and allows host-based verification; secure passwords because all password traffic is encrypted when you connect to a server
- Handles large databases; installations of MySQL with some databases contain 50,000,000 records
- Full support for several different character sets including ISO-8859-1 (Latin1), big5, ujis, and more
- Current versions of MySQL support the creation of full text indexes on tables

Installing MySQL

If you're using a Red Hat-based Linux distribution or your Linux distribution has support for Red Hat Package Management (RPMs), it's easier to install MySQL from RPMs. The RPM files you want to download from www.mysql.com/ are MySQL-VERSION.i386.rpm and MySQL-client-VERSION.i386.rpm, where VERSION is the version number of the server you want to install. These RPMs contain the MySQL server and the MySQL command-line client utilities that allow you to administer and query your databases. Once you've downloaded the RPMs, issue the following command as root:

```
 rpm -i MYSQL-VERSION.i386.rpm MySQL-
client-VERSION.i386.rpm
```

This will install the MySQL server and client and also place entries in /etc/rc.d/ to start the MySQL server at boot up.

If your system doesn't support installations via RPM, you'll need to download a binary distribution of MySQL. Their names are similar to mysql-VERSION-OS.tar.gz where VERSION is the version of the MySQL server and OS is the operating system the binary distribution has been compiled for. Once you've downloaded the binary distribution for your operating system, perform the following steps as root:

```
shell> groupadd mysql
shell> useradd -g mysql mysql
shell> cd /usr/local
shell> gunzip < /path/to/mysql-VER-
SION-OS.tar.gz | tar xvf -
shell> ln -s mysql-VERSION-OS mysql
shell> cd mysql
shell> scripts/mysql_install_db
shell> chown -R mysql
/usr/local/mysql
shell> chgrp -R mysql
/usr/local/mysql
shell> bin/safe_mysqld --user=mysql &
```

After installing MySQL using RPMs or from a binary distribution, you need to create logins for users in MySQL. If you have Perl and DBI installed on your system as many standard Linux distributions do, you can use the "mysql_setpermissions" script to create users and passwords for your MySQL server. If you don't have Perl or DBI installed, read Chapter 6 in the manual that comes with the MySQL server since the topic of MySQL security could take up an entire magazine article!

Installing MM.MySQL

The latest versions of MM.MySQL are always available at the MM.MySQL distribution Web site, www.worldserver.com/mm.mysql. There's usually a stable and a beta release.

If you're using a Java-2 JDK, the easiest installation method is to place the mysql-bin JAR file in the jre/lib/ext subdirectory that exists in your JDK's home directory. The JVM will then automatically load the driver when needed.

If you're using a Java-1 JDK or don't have access to the jre/lib/ext subdirectory of your Java-2 JDK, add mysql-bin.jar to your CLASSPATH environment variable. For example, in UNIX with CSH you'd do the following:

```
 setenv CLASSPATH
$CLASSPATH:/path/to/mysql-bin.jar
```

where /path/to/mysql-bin.jar is the full path to where you've placed the mysql-bin JAR file.

Often application servers don't refer to the CLASSPATH variable when determining where to load third-party libraries from. For example, with any J2EE-compliant application server you're deploying JSPs on, you can place the mysql-bin JAR file in the WEB-INF/lib sub-directory of your application. The application server will automatically load the driver from there. Other application servers require CLASSPATH to be set via their configuration files. Details for this configuration option can be found in the documentation for your application server.

Typical Usage Patterns

In most cases, to use a JDBC driver from your code you should follow these steps:

1. Call Class. for Name ("org.gjt.mm.mysql.Driver").newInstance()
2. Get a java.sql.Connection by calling DriverManager.getConnection("some jdbc URL")

MM.MySQL's JDBC URL syntax is the following (items in square brackets are optional):

```
jdbc:mysql://[hostname][:port]/[dbnam
e][?param1=value1][&param2=value2]
```

"MM.MySQL makes it extremely easy to store and retrieve Java objects"

Where the hostname is the host that the database server is running on (defaults to localhost), the port is the TCP/IP port the server is listening on (only if it's listening on a nonstandard port), and dbname is the name of the database to connect to. The parameters (see Table 1) after the "?" allow you to pass more configuration information to the driver.

3. Create a java.sql.Statement by calling Connection.createStatement() or Connection.prepareStatement() on the Connection instance you created in Step 2.
4. Issue a query on the statement from Step 3 by calling executeQuery() for queries that select rows, or executeUpdate() for queries that insert/update/delete rows.
5. Use Connection.prepareStatement() to create a PreparedStatement that allows you to issue parameterized queries that also take care of quoting special characters and dealing with binary data.

Using MySQL-Specific Functionality

MySQL has some features that can't be accessed from the methods provided in the standard JDBC API. To access them you need to cast the Statement or PreparedStatement object you're using to org.gjt.mm.mysql.Statement or org.gjt.mm.mysql.PreparedStatement respectively.

Name	Use	Default Value
user	The user to connect as	none
password	The password to use when connecting	none
autoReconnect	Should the driver attempt to reconnect if the connection dies? (true/false)	false
maxReconnects	If autoReconnect is enabled, how many times should the driver attempt to reconnect?	3
initialTimeout	If autoReconnect is enabled, the initial time to wait between reconnects (seconds)	2
maxRows	The maximum number of rows to return (0 means return all rows)	0
useUnicode	Should the driver use Unicode character encodings when handling strings? (true/false)	false
character Encoding	If useUnicode is true, what character encoding should the driver use when dealing with strings?	none
ultradevHack	Should the driver generate CallableStatements to support Macromedia UltraDev (true/false)?	false

TABLE 1 Parameters pass configuration information to the driver.

From either of these classes you can call the methods getLastInsertID() to get the value created for any AUTO_INCREMENT field and getLongUpdateCount() to get the larger update count that MySQL can produce as a long. Listing 1 shows how to do this.

The JDBC API version 3, under development by JavaSoft, will have a portable way of retrieving AUTO_INCREMENT values; MM.MySQL will support that method when it becomes available.

Storing/Retrieving Serialized Java Objects

Because MySQL has support for BLOBs, it's relatively easy to store serialized Java objects in the database. The easiest way to do this is to use java.sql.PreparedStatement's setObject() method to get the objects into the database. If you're using a version of MM.MySQL earlier than 2.0.3, you'll need to use org.gjt.mm.mysql.Util's readObject() method to read serialized objects from the database. If you have MM.MySQL version 2.0.3 or newer, use java.sql.ResultSet's getObject() method to retrieve serialized objects from the database. Listing 2 provides an example of how to do this, assuming you have a table named "serObject" in your database with a column named "user" and one named "cert" (e.g., to store a cryptographic certificate).

As you've seen, MM.MySQL makes it extremely easy to store and retrieve Java objects. Because MySQL requires no special syntax to work with BLOBs, it's easy to manipulate data that contains them, which makes it easier to write queries that deal with BLOBs.

Conclusion

I hope you've seen that MM.MySQL and MySQL are simple to use, yet robust and powerful. They're both flexible tools to use in developing Web-based applications, but simple enough for beginning Java developers to learn.

As MM.MySQL is an open-source product, I'm always looking for suggestions for further development and assistance with development, so please contact me at mmatthew@thematthews.org for more information.

mmatthew@thematthews.org

Author Bio

Mark Matthews is a senior consultant for marchFIRST where he helps clients develop solutions using the Java platform.

Listing 1

```
// Assume conn is a java.sql.Connection to a MySQL database
// that has already been created using DriverManager.getCon-
// nection()

Statement stmt = conn.createStatement();

stmt.executeUpdate("INSERT INTO myTable VALUES ('abc',
'123')");

 // Get the value for the AUTO_INCREMENT primary key

long autoKey = ((org.gjt.mm.mysql.Statement)stmt).getLastIn-
sertID();

 // Get the (potentially) large update count (in this case
 // should be "1"

 long bigUpdateCount = ((org.gjt.mm.mysql.Statement)stmt).get
 LongUpdateCount();
```

Listing 2

```
 // Assume conn is a java.sql.Connection to a MySQL database
 // that has already been created using DriverManager.getCon-
 // nection(), and that we have a java.security.cert.Certifi-
 // cate named "cert" given to us by some client that we want
 // to store in the "serObject" table in our MySQL database

 PreparedStatement pstmt =
 conn.prepareStatement("INSERT INTO serObject values (?, ?)");

pstmt.setString(1, "username");
pstmt.setObject(2, cert);

// By calling executeUpdate here, a row will be inserted,
// where the user column will be set to "username" and the
// cert column will have a serialized version of
// the certificate

pstmt.executeUpdate();

pstmt.close();

// Okay, now let's retrieve the certificate

Statement stmt = conn.createStatement();

ResultSet rs =  stmt.executeQuery("select cert from serObject
where user='username'");

while (rs.next()) {
// If you're using MM.MySQL 2.0.2 or earlier, retrieve the
// certificate using the utility class

java.security.cert.Certificate aSerializedCert =
 (java.security.cert.Certificate)org.gjt.mm.mysql.Util.readOb-
 ject(rs, 1);

// Or, if you're using MM.MySQL 2.0.3 or newer, use this
// format:

java.security.cert.Certifcate aSerializedCert =
(java.security.cert.Certificate)rs.readObject(1);

// Now, do something with the certificate

}

rs.close();

stmt.close();
```

Download the Code!

The code listing for this article can also be located at

www.JavaDevelopersJournal.com

JRun 3.0

by Allaire Corporation

REVIEWED BY JOE MITCHKO

AUTHOR BIO
Joe Mitchko is a principal engineer with eTime Capital, Inc., where he specializes in Internet architecture.

jmitchko@rcn.com

Test Environment:
Dell Inspiron 3800 notebook with Pentium 3 600MHz processor with 256M RAM. Windows 2000 Professional with SP1 installed.

Allaire Corporation
725 Grove St.
Newton, MA 02466
Web: www.allaire.com
Phone: 617 219-2000
E-mail: info@allaire.com
Pricing: Developer – Free
Professional: $795 per processor
Enterprise: $4,995 per processor

One of the first things that crossed my mind after being asked to review JRun 3.0 was this: How could I objectively evaluate this product without being biased by my own experience working with a competing product? To give you a little background, all but one of the major Internet application projects I've worked on used this other product (which will remain nameless) as their Java application server. In addition, I've seen this product used so often in conjunction with Oracle Enterprise and Sun servers that the combination in my mind has become a semiofficial architectural standard. Despite its prominence in the Java application server space I had to come to the realization that other application servers on the market are just as capable and have unique and desirable features of their own. One such product is Allaire's JRun Application Server 3.0.

JRun 3.0 ships with several new features that make it very competitive, including an Enterprise JavaBean (EJB) server, advanced transaction and messaging services, and improved load-balancing and fail-over services. Add some new development features, and you have a viable alternative to develop large-scale Internet applications. It's not that JRun doesn't have a large installed base. With over 20,000 licenses sold JRun is one of the most widely adopted JSP/servlet engines around. The full adoption of the Java 2 Platform, Enterprise Edition (J2EE) makes this product all the more capable.

Reviewing a Java Application Server for J2EE can be a daunting task. Instead of providing you with a laundry list of JRun's features (which will only put you to sleep), I decided instead to concentrate on those things a developer would need to know to get started, including installation, licensing issues, hardware/software prerequisites, and J2EE features. I'll also provide an administrator's view of the product. Finally, I'll cover some of the unique features of JRun, especially those that help it stand apart from the competition.

Getting Started

Okay, you want to get started as a JRun developer. First, you'd typically ask: How do I get a copy of the product and how much time do I have before the evaluation period ends? In addition, you'd need to know how much a development license costs and if enough has been budgeted in the project plan. The latter two questions become a nonissue as we'll see later. The question of what kind of machine you need to install JRun on is also a lot easier because JRun is a 100% Java product. You can run it on a variety of machines, including Windows NT, Windows 2000, and Solaris. So far, getting JRun installed and running on your desktop is easy. How well is it documented? As I'll discuss later, the documentation is excellent and helpful in getting you going as a developer. Starting up with JRun is as easy as it sounds.

Key Developer Features

JRun 3.0 provides several new features that improve overall performance. Resource pooling is automatically implemented for various server objects, including database connections, JSP pages, and query statements. In addition, JRun uses a server side data cache that reduces disk I/O operations. Admittedly, outside of system logging I can't imagine what the server would need to cache; database I/O is typically processed by a separate database system.

Often when testing Enterprise JavaBeans, you need to shut down the server to reload your components. The time it takes to shut down the service and bring it back up again time after time adds up. With JRun's Dynamic Bean, loading capability you can redeploy a beans implementation without having to restart the application server. There are some restrictions, though. For instance, you can change and reload only the bean's implementation and not its home and remote interfaces. If the interface does change, you'll need to recompile the JAR file and restart the service. The second limitation is that you need to start the service in stand-alone mode (from the command line).

Overall, this feature can be especially helpful when you have many developers cranking out beans and deploying to a central development server. You can't afford to be constantly restarting the server every time a bean changes. The same mechanism used to redeploy EJBs is similar to the one used to redeploy servlets.

JRun comes with a set of JSP custom tags that allow you to work with common J2EE API calls without the need to directly program it in Java. The JRun Tag Library provides a convenient way to invoke common J2EE API calls, handle HTML form processing, populate dropdown lists, and use conditional JSP block expressions. For example, you can send a message to a specific JMS subscriber queue using a single JSP tag statement. Another surprisingly useful tag in the library does XSL processing and transformation of an XML source document.

In addition to the limited documentation provided out of the box on the Tag Library, you can download full documentation from Allaire's DevCenter located on their Web site. They do provide a quick reference card for most of the JSP tags. Although the JSP tag library is useful, there's no way to handle the J2EE exceptions occurring within the embedded HTML files. This essentially leaves you at the mercy of the default exception handler resident in the JSP/servlet engine.

JRun provides out-of-the-box J2EE functionality, including Java Message Service (JMS) and Java Naming and Directory Interface (JNDI). You can use the JRun implementation of these services or configure a third-party product to use with the standard APIs.

One of the points stressed by Allaire is that JRun 3.0 is fully compliant with the J2EE architecture and the server doesn't alter any of the interfaces or modify their implementation. In fact, JRun 3.0 is listed on Sun's Web site as a fully licensed Java 2 Platform, Enterprise Edition server. It also passes Sun's write once, run anywhere (WORA) requirements by getting a passing grade on the J2EE verification test suite. This means you're guaranteed that JRun fully implements the J2EE interface and can use components developed on other J2EE-compliant platforms on an interchangeable basis.

The JRun 3.0 application server also provides built-in benchmarking tools through its Instrumentation services. The Instrumentation service is a nonintrusive way to benchmark various method calls within a servlet and present the information to the log file. It's also flexible in that you can optionally change where the benchmarking information is sent. For instance, you can route the logging information to a different file or provide access to the information through an HTML page. The one limitation is that Instrumentation records only metrics for servlet method calls.

Although full J2EE compliance is a big selling point, Allaire still decided to include API extensions. It seems that every vendor needs to include some sort of unique capabilities to their product. Repeatedly you go through the same decision-making process on a project: Do you or don't you use the extensions provided by a vendor? The use of extensions can help make a more robust application, but they do have a tendency to lock you in to a particular product. It all depends on weighing the advantages and disadvantages.

The good news is that JRun 3.0 extensions are limited in scope. For example, one of the extensions involves extending the HttpServlet-Response interface allowing servlets to programmatically call a JSP document. There are a few more, but that's it. It's up to you whether to use them or not. In the case of JRun, use of extensions shouldn't pose a significant problem.

Finally, JRun 3.0 provides network connectors for the following Web servers:

- Apache
- Microsoft IIS
- Netscape Enterprise/FastTrack/iPlanet
- Zeus WebServer
- O'Reilly WebSitePro

The request is processed by first passing it to JRun, then passing the results back to the Web server.

Key Administrative Features

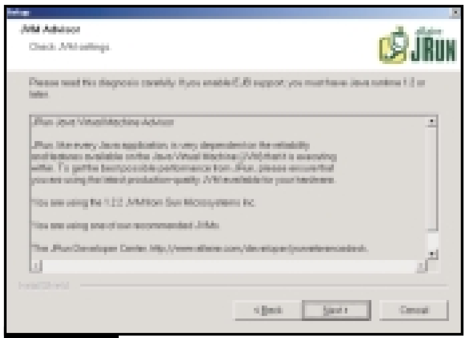
The JRun 3.0 Enterprise Edition includes a copy of ClusterCATS, which provides robust features for Web site availability, load balancing, and fail-over services.

You can configure ClusterCATS to automatically restart a server if it's not responding using the Remote Method Invocation Daemon (RMID) protocol. The restart command is issued by the JRun monitor and associated probes and occurs when the server isn't responding to HTTP "GET" requests from the monitor. By design, the ClusterCATS service can only shutdown and restart the full JRun service. It doesn't have the ability to restart specific subsystems (for instance, only shut down and restart the EJB services). Since all services run under a single JVM, it's all or nothing.

One interesting feature ClusterCATS provides is the ability to feed load-balancing metrics to a Cisco LocalDirector router. Using Cisco's Dynamic Feedback Protocol (DFP), ClusterCATS provides the router with various load and volume metrics. LocalDirector then uses that data to make better load-balancing decisions across a clustered server environment. To use this feature you must have LocalDirector version 3.1.4 or later.

Clustering occurs at the HTTP request level (primarily servlets) and provides a high availability mechanism for your application. In addition, you gain a level of scalability by having more than one application server servicing HTTP requests.

JRun supports session-level failover by using session persistence in conjunction with ClusterCATS. Similar to other clustering strategies, ClusterCATS provides a "sticky" mechanism that will bind a particular user application session to a server, therefore maintaining session state on a single server. One area of



JVM Advisor

Check JVM settings

Please read this diagnosis carefully. If you enable EJB support, you must have Java runtime 1.2 or later.

JRun Java VirtualMachine Advisor

JRun, like every Java application, is very dependent on the reliability and features available in the Java Virtual Machine (JVM) that it is executing within. To get the best possible performance from JRun, please ensure that you are using the latest production-quality JVM available for your hardware.

You are using the 1.2.2 JVM from Sun Microsystems Inc.

You are using one of our recommended JVMs.

The JRun Developer Center: http://www.allaire.com/developer/jrunreferencedesk

FIGURE 1 JRun 3.0 setup

improvement would allow an EJB client to use EJB services on any machine in a clustered server environment. JRun supports EJB servers distributed anywhere on the network.

Another smart feature is the ability to program the ClusterCATS administration module to diagnose whether a server is "up." To accomplish this, JRun allows you to configure small self-test operations called JRun probes that can be assigned to an individual monitor. In turn, each server can be assigned a monitor containing multiple probes. Each probe requires some custom configuration by the site administrator and works by comparing retrieved HTML content (from a specific URL) against a known good set. Any variation in the content or nonresponse by the server is considered by the probe to be a server failure, and the monitor will act accordingly to shutdown the service.

Last, JRun 3.0 provides an administrator with some flexibility in configuring the servers. For instance, you can run JRun as a standalone servlet engine or a servlet engine/EJB service depending on the configuration settings.

Installation

Installing JRun is a two-step process involving a straightforward setup program (for Windows 2000) and an HTML configuration step (see Figure 1). The setup program guides you through the usual application server questions, including what JVM you want to use (I had six on my machine – decisions, decisions), TCP/IP port designations, and more. The setup phase of the installation performed without any difficulties and has a clean, well-laid-out design. The individual setup frames provide just enough information so you can make an intelligent choice. Allaire took some extra care here. The installation CD also includes a JVM you can install if you don't currently have one on your machine. The Professional version requires JVM 1.8 or later (support for EJBs isn't included), while the Developer and Enterprise versions require JVM 1.2 or later.

One particular annoyance in the installation process occurred when I was asked to enter the initial administrative login on the JRun Management Console (see Figure 2). Although earlier in the setup you were required to enter an administrator password (that I remembered), I wasn't quite sure what the login name was. After trying different combinations of "Administrator," "Admin," etc., I finally stumbled on "admin" as the correct one. The proper login was mentioned on the password assignment page, but the help message was a bit confusing. I think the message "JRun Admin Password (Admin username is 'admin')" threw me off. It would help if the login defaulted to "admin" in this case to get you started.

The second phase of the installation is browser based and executed automatically by the setup program if you want to connect an external Web server to JRun. Here's where it became fun. The installation performed so cleanly that I decided to try to set up a connection between a Microsoft Internet Information Server (IIS 4.0) running on a separate NT server to my JRun service.

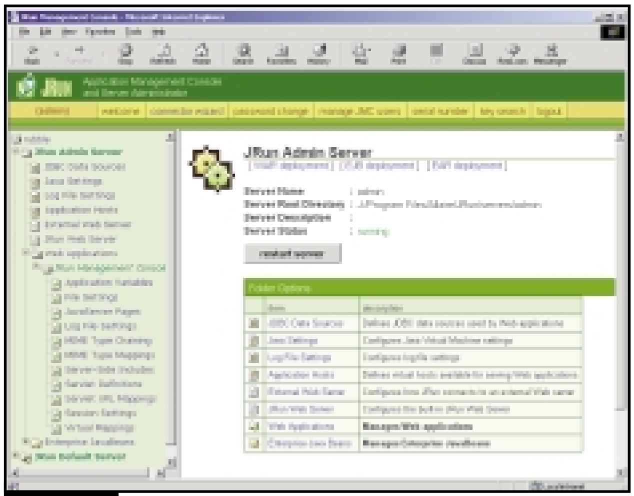
Having previously had the opportunity to set up and configure Microsoft's IIS 4.0, I realized that getting IIS to redirect HTTP requests to JRun wouldn't be an easy task. Getting the connection set up and working correctly is part skill, part luck, and part voodoo. Being somewhat adventurous, I let JRun's Connector Wizard walk me through the process (see Figure 3). This part of the setup seemed to be working correctly, and it was definitely going through the proper steps. For instance, it knew where to place the DLL file under the IIS virtual directory (c:\inetpubs\scripts) and provided directions as to when to restart the IIS WWW services on the NT server.

When the decisive moment came to test the connection from IIS to JRun, it didn't work. The only option at this point was to finish the installation process using the management console on IIS. For those of you familiar with IIS 4.0, you know what a hassle it is to successfully set up connections to third-party application servers using the administrator. There was also a slight glitch in the administrator where the connector wizard wouldn't proceed to the fourth step when the "Install as Global Filter" checkbox was off. There was no user feedback requiring you to check the box before proceeding. Aside from the problems encountered in the connection wizard, the JRun Management Console appears to be well designed and organized.

Documentation and Support

I've always enjoyed getting a boxful of manuals whenever I buy software. I guess it's something tangible to hold onto when you first learn how to use a product and you're not quite sure what's going on. At least you can strategically place the box in your cubicle to make a statement – hey look at me, I'm an expert (even though you don't have the slightest clue). Don't get me wrong. I think it's wonderful that you can download and install a sophisticated development product nowadays without even getting a CD, and through the vendor's Web site you're able to get enough PDF files to keep you busy reading for years. It's still nice to get a box. I guess you can call me old fashioned.

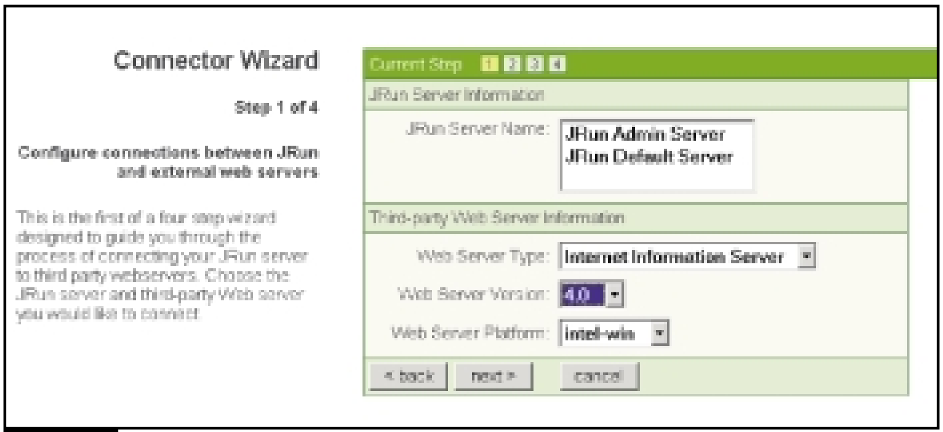
Anyway, what I am leading up to is that


JRun Admin Server

[WAR deployment] [EJB deployment] [EAR deployment]

Server Name : admin
Server Root Directory : [illegible]
Server Description :
Server Status : running

FIGURE 2 JRun 3.0 Management Console


Connector Wizard

Step 1 of 4

Configure connections between JRun and external web servers

This is the first of a four step wizard designed to guide you through the process of connecting your JRun server to third party webservers. Choose the JRun server and third-party Web server you would like to connect.

JRun Server Information

JRun Server Name: JRun Admin Server / JRun Default Server

Third-party Web Server Information

Web Server Type: Internet Information Server

Web Server Version: 4.0

Web Server Platform: intel-win

FIGURE 3 JRun 3.0 Connector Wizard

with JRun 3.0 you get a box full of the best-written documentation I've seen in a while for this type of product. Besides an installation guide, you get a comprehensive development guide, a samples guide, a guide to using ClusterCATS, and some handy quick-reference cards. In addition, you can go to their Web site for a wealth of additional information and development support. The *Developing Applications with JRun* manual is exceptional and not only provides you with a step-by-step approach to developing J2EE applications in JRun, it's also an excellent tutorial on J2EE technology itself. The guide provides plenty of examples and is very polished. Allaire gets extra credit for this one.

Licensing

JRun 3.0 comes in three versions, Developer, Professional, and Enterprise. As a developer you often need to obtain a copy of a development product in advance of purchasing a license. For example, you may need the product for evaluation purposes. The most common reason is that it usually takes time to purchase and get a license key. Allaire gets some extra brownie points again – the licensing for JRun Developer is free. This takes a bit of the stress off the development team.

The Developer version limits you to three concurrent connections (plenty for the individual developer) and an unlimited number of concurrent JVMs. Of course, this version isn't meant for deployment purposes. The Professional version allows an unlimited number of JVMs and sessions for servlets and JavaServer Pages (JSP). This version is good if you're deploying a simple application using servlet and JSP technology without the EJB middle tier. The price is considerably less, too, so that you're not paying for unused functionality.

The Enterprise version comes with the full-blown J2EE environment, including EJB, JMS, and JTA. The Enterprise version also includes load-balancing and fail-over capabilities using Allaire ClusterCATS (mentioned earlier). All three versions are packaged without any type of Integrated Development Environments for EJB or JSP/servlet development. For that a good companion tool would be Allaire JRun Studio that's based on the HomeSite HTML editor.

Summary

JRun provides a competitive product in the Java application server space. Besides implementing J2EE, JRun provides some nice additions that make developing Java Internet applications a whole lot easier. The price is competitive, and it does provide better licensing flexibility than the competition. It could use some improvement in its current clustering implementation by providing fail-over mechanisms below the HTTP request level, but outside of that it has about everything you need to create sophisticated Java applications using J2EE architecture.

Interview...with Paul Lipton

Director of Object Technologies at Computer Associates

An interview by David Johnson

***JDJ:* Paul, tell us about some of the things going on at Computer Associates.**

Lipton: In the past, most people knew CA (www.ca.com) as a big mainframe company, and they may also know us as the company that acquired Sterling Software in the last year and Platinum Technologies the year before that. Certainly we do have a lot in the areas of enterprise management and e-business for many platforms. But not enough people in the Java community realize how powerful our Java solutions are. For example, we have Jasmine ii, which is something incredibly interesting for the Java community to look at.

***JDJ:* What does the ii stand for?**

Lipton: It stands for intelligent infrastructure, and that's what Jasmine ii is all about. Once you get past the "What app server am I going to buy?" stage, you start to look deeper at the real-world problems that companies, their suppliers, and their customers are really facing, especially large, modern corporations. They have a vast amount of legacy applications. Many Java developers tend to say, "Oh, yes, those are the legacy applications. I'll integrate with them later. Let me write some JSPs and servlets, get into my presentation logic, do the fun stuff. I'll worry about that old nasty stuff later on."

What we ignore when we do this is that these legacy applications are at the core of the enterprise. You have to remember that these systems were developed first, because they addressed the most important and fundamental problems these companies had. For example, "Who are my customers, and how do I send them a bill?" or "What products do I have, and which ones are in my warehouse right now?" Often, these systems have been used, literally, for generations. I don't mean generations of technology, I mean generations of human beings. Now, in a sense, they're no longer IT business systems. They've actually become part of the company's business processes. The company couldn't exist without them. They've become part of the heart and lungs of that corporation.

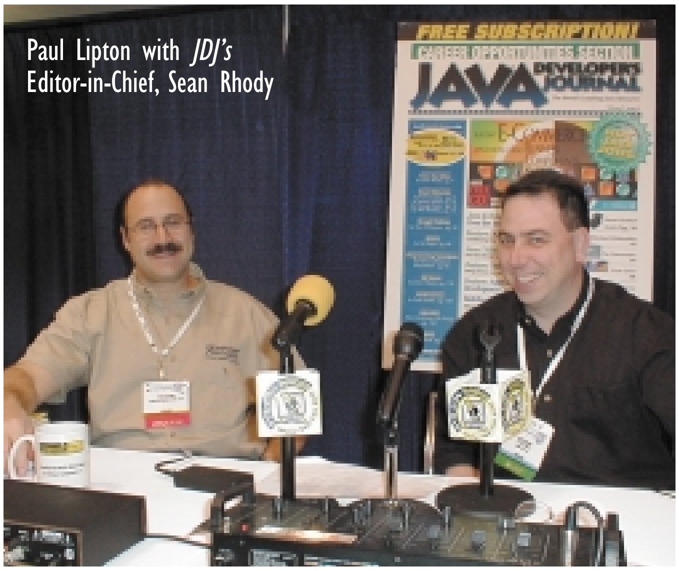
These essential business systems are incredibly diverse – not just UNIX-based relational databases sitting on a UNIX box – but everything: giant mainframe network databases, VSAM files, terminals hooked up to CICS or AS/400 machines, you name it! Right now, there are VT100 terminals hooked up to machines that may never see a Java Virtual Machine, because there's simply no money for that to happen. There's just an incredible range of technologies: messaging, accounting systems, ERP, and decades of custom development, not just in C++ and VB, but also in COBOL, Fortran, and MODULA-2. Think about the importance of these systems. What did IT pay for Y2K remediation; wasn't it something around $2 trillion? Think what the cost of construction for those systems must have been.

So, on one hand you have these essential legacy systems, which will never go away. On the other hand there's a very compelling demand for these new e-business initiatives that companies want to have: B2B, B2C, exchanges, portals, and so on. It's essential for them to build to stay competitive in this new era, and the problem is, of course, connecting those two very different worlds.

That's just part of the value of Jasmine ii. It's an infrastructure for connecting the modern world and the modern technologies that are hitting us at a mile a minute – Java, XML, and everything else coming down the road – with this vast world of legacy systems. And it does so in a compelling and natural way for Java developers and people who are XML-centric.

Our goal was to make Jasmine ii something easy for Java developers, but it's also open to people using other technologies. As Java developers we tend to want to ignore the rest of the universe, such as Microsoft. I think that we don't really serve our function as developers very well by pretending that companies like Microsoft don't exist. They happen to be, I believe, the most heavily capitalized corporation in the world, and there are some vast technical resources there.

We have to be realistic, and we have to be open. We have to stop playing in our own little sandbox and hoping the bad guys go away. Part of what Jasmine ii brings is this openness, openness to Microsoft technologies, to Java, to CORBA, to XML, to applications such as SAP and PeopleSoft, and to the vast world of legacy systems. It's all very important if we're going to address the challenge of bringing all these technologies and systems together.


Paul Lipton with *JDJ's* Editor-in-Chief, Sean Rhody

***JDJ:* What about the Neugents product? What can you tell us about the neural network-based technology and some possible applications?**

Lipton: Neugents technology is a remarkable, patented, neural network technology that we've developed at Computer Associates. It started a few years ago, when we acquired AIWARE, which was a leader in this technology. We'd already been doing a considerable amount of R&D on fuzzy logic and related topics, so we had an acute interest. Since then Neugents have been used with great success in CA products, such as Unicenter TNG, in the areas of prediction and optimization. For example, wouldn't it be useful to be able to predict that there's a 95% chance the server will go down in 45 minutes? You might be able to do something about it ahead of time, be proactive as opposed to tearing your hair out and being reactive.

Neugents aren't just an expert system or a bunch of rules. They're a technology patterned after the biological neural network in the human brain: adaptive, intelligent, and self-learning. They're integrated into Jasmine ii, as many of our products are. You can think of Jasmine ii almost as a motherboard, and a great number of complementary CA technologies – like our repository technology, UML modeling tools, and so on – are daughterboards that can plug into it. This infrastructure adds tremendous value as a solution to people who are trying to build and deploy modern e-business solutions. Our Neugents technology is yet another important daughterboard that plugs into the Jasmine ii infrastructure.

We've spent a lot of time and money making this technology accessible to business programmers instead of PhDs, because it can be tremendously useful for all sorts of e-business scenarios. For example, Neugents can be used to uncover a wide range of e-business opportunities with suppliers and customers, providing a sort of dynamic, self-learning personalization to many e-business problems and tasks.

And, Neugents are useful to tasks in manufacturing, finance, and many others. We have clients doing things such as manufacturing paint. When you manufacture paint it's difficult to get that right formula, the right characteristics of paint, such as the sun resistance, luminescence, and abrasion resistance. In the past, it's been almost an art form, and often it was an expensive and laborious process to come up with a formula that had the required characteristics to paint a certain kind of building or bridge, for example. Neugents have made that process far less expensive and more reliable. And, clients are using it for more than manufacturing – all sorts of optimization and prediction.

Going back to e-business, Neugents technology uses its pattern recognition capability to effectively identify all sorts of business opportunities that may be otherwise missed. For example, you could use Neugents for a kind of rich, dynamic personalization. I'm not talking about the static personalization you see out there so much today. Say you order a book on deep sea diving. A Neugent-powered site could offer you two more books on diving, as well as related equipment, such as tanks or fins, and stand a good chance to score a hit there. I'm talking about the kind of adaptive learning that in real time examines all the demographic data, behavioral characteristics, and clickstream data, and can come up with a level of personalization that can be far more useful. We have clients doing that as well.

Again, the applicability of the Neugents technology is profound. The key was to make it accessible. Because Neugents intelligence services are part of the Jasmine ii infrastructure, it's open to everybody.

***JDJ:* What does a director of object technology do? What's your typical workday like, and where do you take your job on a daily basis?**
Lipton: That's actually an impossible question to answer, but it's one of the great things about working for Computer Associates. We have so many solutions that touch so many different areas of the IT universe. Because of that, I've had fun learning about technology in many different areas and learning about business in a lot of different areas, as well.

My day is a curious mix. I do occasionally get to be really technical. If I'm careful I can write a little bit of Java code before they chase me away from it. I have the opportunity to talk to many of our clients and business partners about CA technologies. I've also participated in some of our architectural efforts. I was one of the people who worked with Jasmine ii in the early days.

Designing Jasmine was interesting for us. Of course, XML wasn't a big thing then; now it's the flavor of the month. It was a good test of our architecture and design to see how hard it would be for Jasmine ii to support XML. It turned out to be almost effortless, proof that we'd done the job right. Jasmine ii has XML support on both the front and back ends. And it has unique XML capabilities.

People, especially in the Java community, see an XML document and want to run SAX or DOM against it and pull in this document. That's a document-oriented approach. We think XML documents are also information resources. A real solution would support things like concurrency and transactions, and that's what we do. We support XML as a richer information source and integrate that XML information into the rest of the enterprise, including legacy applications. Our ability to do that is proof to us that we can handle the newest technologies as well as the old ones in this transparent manner.

***JDJ:* I've been seeing some interesting presentations at your booth.**
Lipton: We're here showing off our Jasmine ii infrastructure as well as COOL:Joe, which is our EJB development and deployment tool. COOL:Joe has far too many features for both experienced and inexperienced J2EE developers for me to discuss here. But, for a technologist and architect like me, who doesn't always get the chance to write much code, COOL:Joe has a tremendous amount of built-in knowledge about EJBs. All kinds of task advisors and wizards make the job of getting up to speed, using and testing EJBs a lot easier. Experienced EJB developers are scarce, so having something that can help get you up to speed is wonderful, and even experienced developers appreciate its modeling and other advanced capabilities.

For this show, we wanted to zero in on two particular CA solutions that were Java-centric, get closer to the Java community, and share knowledge. CA understands the importance of industry standards like Java, but we also understand that Java has to play in the real world of legacy and other important, sometimes competing, technologies. Some people have said that CA is the Switzerland of software companies because we tend to be neutral and focus on solutions for our clients. We aren't here to bang the drum for any particular narrow view of the IT universe. We're here to help solve problems, particularly for the Java community.

AUTHOR BIO

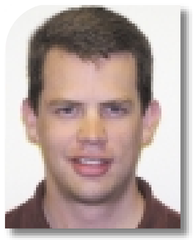
David Johnson is CEO of Verge Technologies Group, Inc., a Boulder, Colorado–based Enterprise Java consulting and hosting firm.

djohnson@vergecorp.com

WebKing 2.0

by ParaSoft Corporation

REVIEWED BY BRIAN R. BARBASH



AUTHOR BIO

Brian R. Barbash is a consultant for the Consulting Group of Computer Sciences Corporation. He specializes in application design and development, business and technical analysis, and Web design.

bbarbash@csc.com

JAVA DEVELOPER'S JOURNAL JDJ ★★★★★★★★★ WORLD CLASS AWARD

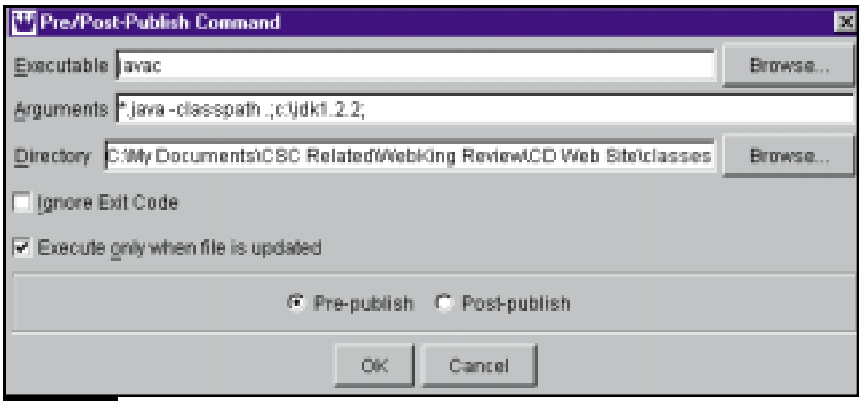
ParaSoft Corporation
2031 Myrtle Avenue
Monrovia, CA 91016
Web: www.parasoft.com
Phone: 888 305-0041
Fax: 626 305-3036
E-mail: info thewebking.com

Test Environment
Machine: Intel Pentium III, 500MHz, 128Mb RAM
JDK: Standard edition
Web Server: Apache with Tomcat module for Servlets and JSP
Platforms: Windows NT/2000/98/95, Linux, Solaris
Pricing: Starts at $3,495 (per set)

ParaSoft's WebKing 2.0 Web testing software is a tool to assist developers in testing and deploying dynamic Web sites. It applies traditional C/C++ and Java testing techniques to the Web environment to identify a multitude of potential errors in a site. WebKing also provides a publishing mechanism to automate the testing and transfer of a site's components.

Installation

The installation process for WebKing is very simple. The package is available in both a trial and full version. The Java Runtime Environment is required (JDK2); WebKing is available with the JRE if the host machine isn't loaded with the JDK. For this review the installation was performed on Windows NT Workstation 4.0 with the JDK installed.



Executable: javac
Arguments: *.java -classpath .;c:\jdk1.2.2;
Directory: C:\My Documents\CSC Related\WebKing Review\CD Web Site\classes
[ ] Ignore Exit Code
[✓] Execute only when file is updated
[✓ Pre-publish] Post-publish

FIGURE 1 Adding a prepublish command

Working with Web Sites

To evaluate the deployment capabilities of WebKing, I developed a small Web site that manages a CD collection. The static pages provided basic site navigation options and the ability to add to the music collection. Each page was deliberately left with empty links and invalid tags for WebKing to discover. The pages that were devoted to modifying the collection also included JavaScript for managing and validating user input on the form.

To add the site to WebKing you must specify the source for the Web pages. Sites may be sourced via HTTP, FTP, Telnet, or a file location. *Note:* Specific testing features are available only if the site is sourced by an HTTP connection. In addition, the source code for the site's back-end applications are stored in separate directories away from the site content.

WebKing allows these files to be managed via indirect links, also established by an HTTP, FTP, Telnet, or local directory connection. By combining the site with indirect links, all the content from disparate sources can be managed as one site. Adding this site into WebKing was easy. Within a few seconds it was able to create the tree representation of the site's contents and the material housed in the indirect links.

To set up the publishing, indicate a target point for the content and apply pre- and post-publish commands and custom tests. The target for the evaluation is a local directory served by Apache. The prepublish command issues a Java compile on all back-end source code. Figure 1 shows WebKing's interface for creating publishing commands. All commands are required to receive command-line arguments. Once in place, WebKing automatically executes the compile commands and halts publication if the command fails.

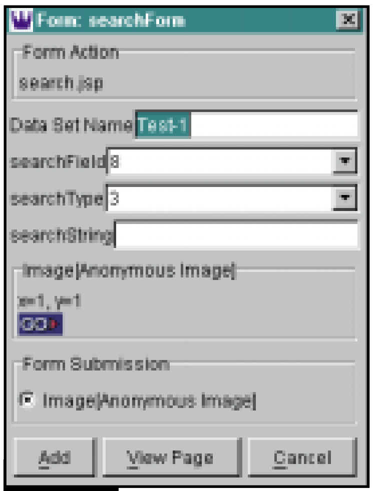
The development process may be further monitored by the enforcement of coding standards. WebKing provides a graphical scripting utility, CodeWizard, that checks the source (HTML, stylesheets, and JavaScript) for errors and standards violations. Furthermore, common components that are designated to appear on all pages within a site may be coded in CodeWizard; the developer is notified if those components don't exist.

The total setup for publishing the site is relatively simple. The time taken to set up the process is well worth the time saved when updates to the site are required. As WebKing publishes the site, all tests are executed, and it completes the publishing operation only if all conditions pass. This allows the developer to identify and fix errors as soon as possible before release. In this case, WebKing caught all the errors in the site's static pages, stylesheet definitions, and JavaScript source code (intentional and otherwise), and identified the file and line number of the error. If the site is sourced from a local file, WebKing allows the developer to specify and launch an editor to make changes to the file in question.

WebKing's testing facilities provide the developer with a wide range of capabilities for putting a site through its paces. Testing functionality is encapsulated in four types of tests: white-box, black-box, Web-box, and regression.

White-box testing examines a site's construction by determining a path through the site, loading static pages, executing server-side programs with automatically generated inputs to retrieve dynamic pages, and loading the entire site into a tree view. It then accesses every page and checks links, HTML, JavaScript, and cascading stylesheet code for errors, and standards deviations. WebKing then executes server-side applications to expose exceptions and other errors.

As the music collection was put through white-box testing, WebKing continued to identify errors in the site's construction. The tests executed in the white-box test facility will continue to be executed as publishing progresses. One key feature of the procedure is the handling of forms. WebKing provides the capability to create customized form data, called a *data set*, and apply it as submitted data against a form (see Figure 2). This allows the developer to test the execution of back-end systems and the construction of the output pages. The data is treated as if a user had navigated the site; therefore, all back-end data stores are


Form: searchForm

Form Action
search.jsp

Data Set Name	Test-1
searchField	8
searchType	3
searchString	

Image[Anonymous Image]
x=1, y=1

Form Submission
Image[Anonymous Image]

FIGURE 2 Form data set creation

updated with the test data if applicable.

As a Web site evolves over time, regression testing is key to ensuring no new errors are generated by code modifications. WebKing performs regression testing by executing all previously saved test conditions. Built into the publishing process, regression testing is automatically executed every time a site is put through publishing.

Regression tests, as indicated above, were automatically executed as dynamic content was added to the music collection Web site. JSP pages and a servlet were added for dynamic display. The JSP pages are set up to behave differently based on the value of the attributes in their URLs. WebKing automatically generated an instance for each possible attribute value. Any errors in the construction of the JSP page are displayed in the same manner that static pages are displayed. In addition, if a wide collection of parameters is available to a

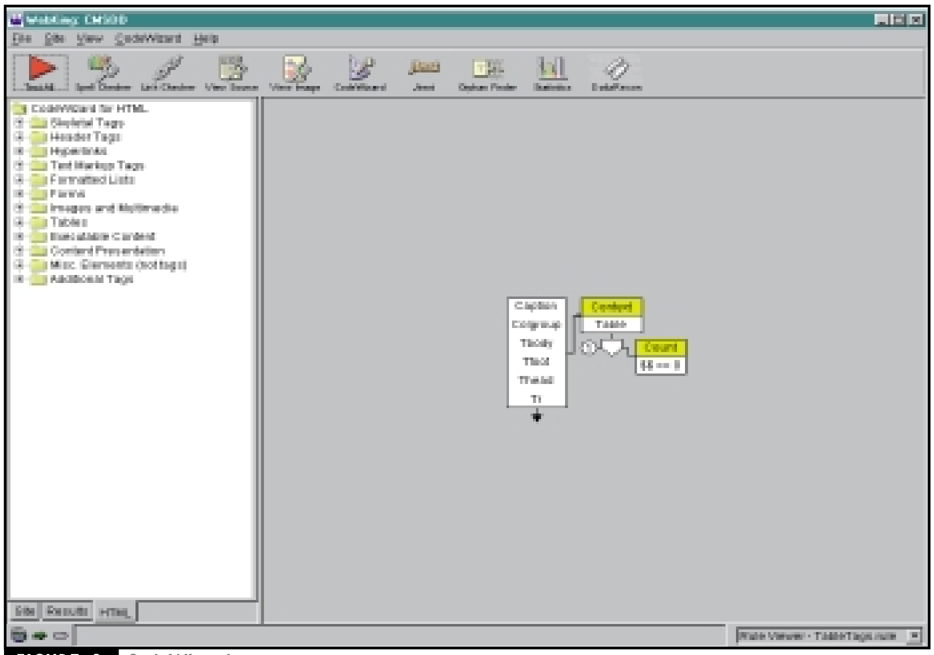


FIGURE 3 CodeWizard

JSP page, WebKing lets you select only the important ones in order to limit the number of page instances created and to allow the developer to focus on key content.

The output of the servlet was created as a page instance based on the data set of its related form. This page is also displayed in the same manner that static pages are displayed and shows any page errors the servlet may generate.

Web-box testing is WebKing's methodology for individually testing dynamic pages. It allows developers to test dynamic pages for their output as well as their source code. Web-box testing revolves heavily around the publishing process and provides a standardized process for developers to release content. Prior to publication, each individual page may be put through tests specific to that page. If any conditions fail, as with standard publishing, the publishing process is halted. Source code for any back-end program may be tested through add-ons to WebKing such as Jtest for Java. For the purposes of this review, only the output piece of Web-box testing was examined. The difference between straight publication and Web-box testing is the concept of critical paths, also seen in black-box testing.

Black-box testing allows the developer to test the functionality of a Web site. WebKing enables sites to be tested via a critical path through the site to ensure it flows and functions properly. Black-box also tests individual pages for elements that must be standardized throughout the site. For the critical path test, WebKing will generate a default set of paths through a site. The developer can then modify the paths and the inputs associated with individual pages, and set up loops and chains to test specific functionality.

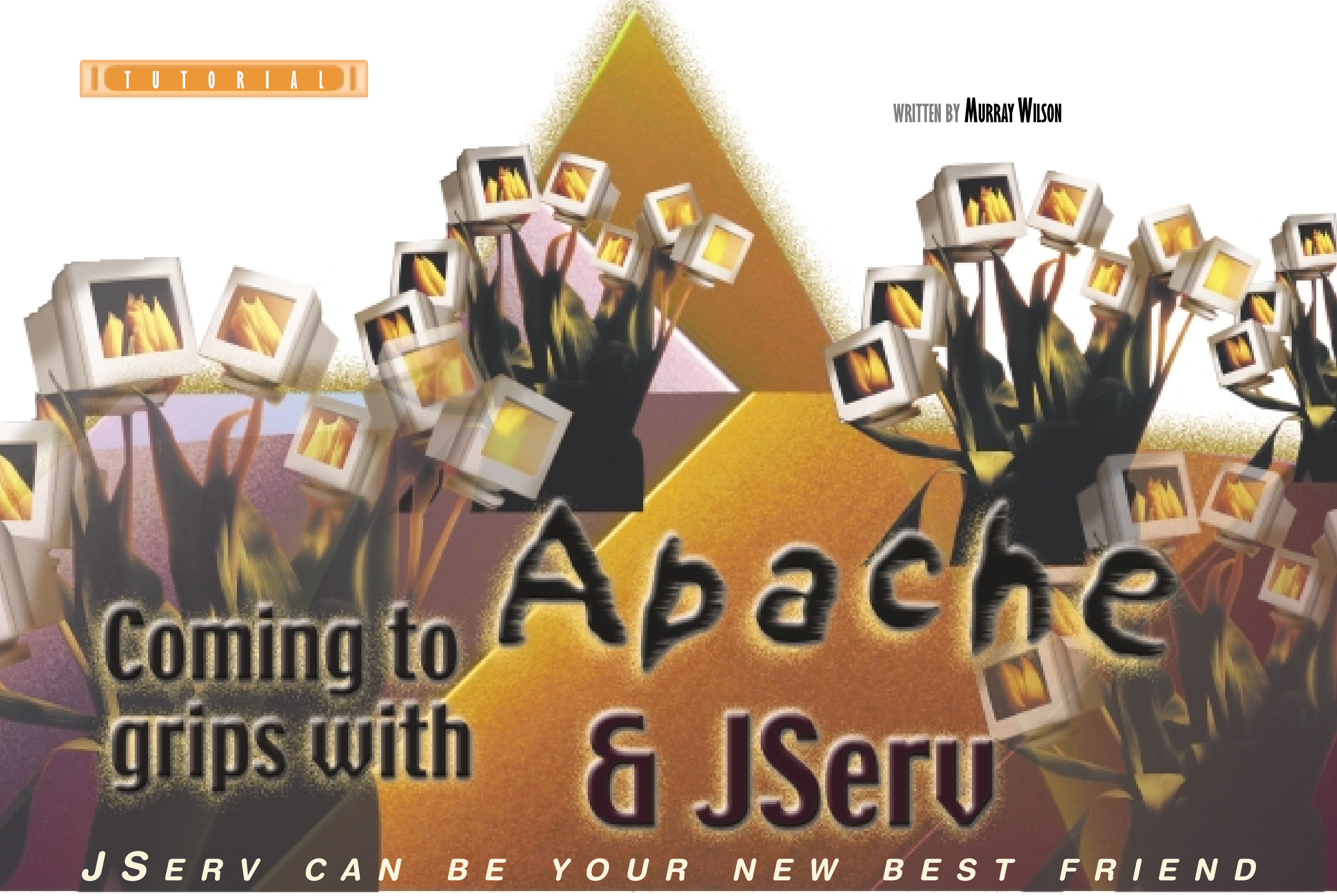
To examine black-box testing, I use a larger site that's a combination of JSP pages and static HTML and already in a production environment. This site relies heavily on parameters passed between pages and a section of it is generated entirely from users' input. WebKing quickly generated the navigation paths through the Web site. To identify only a subset of functionality, specific data sets were entered and paths were either removed or identified to be skipped by WebKing. WebKing then executed all tests and documented the results in the same format as the white-box, regression, and Web-box testing functions.

As for the standardized components of the site, CodeWizard defines the specific patterns of code that make up the specific component. For example, if every page in a Web site will include a specific header, the CodeWizard may be used to define a rule to look for the header. Figure 3 is a snapshot of the CodeWizard and a sample rule. Each time a test session is run against a site, the CodeWizard tests may be executed.

Summary

With the ever-increasing complexity of dynamic Web sites, having a handle on the testing and deployment of site components is a key element to a successful implementation. The testing and deployment capabilities of ParaSoft's WebKing provide an effective solution for organizing, testing, and deploying site components. Its testing functionality allows developers to catch and correct errors and nonstandard code prior to deployment. The pre- and postpublishing commands provide developers with the tools to automate many of the tasks associated with compiling source code and moving files to the production environment. Overall, ParaSoft's WebKing is an easy-to-use and effective testing and publishing utility, and should be considered for the development environment.

WRITTEN BY MURRAY WILSON



Coming to grips with Apache & JServ

JSERV CAN BE YOUR NEW BEST FRIEND

JServ for me represents the best-value servlet engine on the market today. "Why?" I hear you ask. Well, that's easy. It's free, of course. To this day, the fact that I can download the Apache Web server and JServ for Linux and have the beginnings of an excellent enterprise server for nothing surprises me. And why anybody would use any Web server/servlet engine combination other than Apache/JServ escapes me. "What about the lack of support?" the voices cry out. Is that a real reason? I'm not convinced, but I've heard lots of remarks on the "difficulty" of setting up JServ with Apache on Linux. This article provides a step-by-step guide on getting JServ up and running.

We'll start by looking at how to install JServ and how to test the setup of the installation. Then I'll explain some of the more advanced configurations that can be applied to the product. This article won't be a comprehensive investigation into JServ – that would take a book – but I'll cover the more popular configurations that developers use.

Installation and Configuration

Step 1

Before you start, I can't stress strongly enough how much a logical and clear directory structure on your Linux installation helps. Many times I've tried to fix an installation of JServ and have wasted 20 minutes finding where it was installed in the first place. Having Apache installed under /usr/local and JServ installed under /var/etc makes no sense, does it?

Let's start nice and agree on where we're going to install our applications. For me it'll be /usr/local; for you it could be somewhere else, as long as it's consistent.

To run JServ you need the Apache Web server and, of course, JServ itself. JServ and Apache are both available for download from the www.apache.org/ and http://java.apache.org/ sites. Keep these sites bookmarked as they provide an excellent reference point for help when things aren't going too well. When you're choosing which files to download, you'll have to decide between the source distributions or the binary distributions.

I personally like to have a lot of control over what goes on in my system, so I generally choose the source distribution. It means more work in configuring and compiling each application, but I find I learn far more about an application than I would from installing the binaries. For this reason, within this article I'll discuss installation of the source distributions for both Apache and JServ.

You need to install Java on your system and you can download a Linux port of Java from Sun's site, http://java.sun.com/, or you can get a port from www.blackdown.org. I find the guys at Blackdown have slightly better documentation for installing Java on your Linux system as they deal with Linux exclusively.

You'll also need the jsdk.jar (the Java Servlet Development Kit), which can also be downloaded from Sun's Java site. Remember to place this JAR file in a logical place on your file system. I like to set up a specific directory just for JAR files – say, for example, /home/jarfiles/.

Once you've downloaded the necessary archive files – for example, apache_1.3.12.tar.gz and ApacheJServ-1.1.2.tar.gz – they need to be untarred into a temporary directory for configuring and compiling.

Remember, these archives represent the source of the applications, not the applications themselves, so why have them cluttering up the applications directory? I'm going to untar them in /usr/src. It's close enough that I won't have to do too much moving around while installing, but far enough away that I can happily delete it after installing the application without losing any files required to run it. You may want to keep the source, of course; it'll come in handy if you want to compile other modules into your Apache Web server at a later point.

```
tar –xvzf apache_1.3.12.tar.gz
tar –xvzf ApacheJServ-1.1.2.tar.gz
```

After untarring both files we now have the directories /usr/src/apache_1.3.12/ and /usr/src/ApacheJServ-1.1.2/ packed full of the source files necessary for building the applications.

Hurray! We're now ready to start building the two applications.

Step 2

We'll build Apache first; this needs to be done if you've never had Apache compiled on your system before. So we go into the Apache directory.

```
cd /usr/src/apache_1.3.12/
```

We're now going to run the Apache configuration script. We"ll do this with a minimum of parameters, just passing in the path we wish Apache to be installed in. When we compile it, we use the --prefix= option to specify this path. As stated previously, I want my applications installed in /usr/local:

```
./configure --prefix=/usr/local/apache
```

And that should be that. Apache is preliminarily built.

Step 3

Now we need to go into the JServ directory to carry out a similar operation. Let's get into the correct directory:

```
cd /usr/src/ApacheJServ-1.1.2/
```

We have to run the configure script in this directory, and for this we need to pass in more parameters. These parameters will tell JServ where various resources have been placed on your system.

We need to know where the Apache source has been placed if we're going to build the Web server:

```
--with-apache-src=/path/to/source
```

JServ also needs to know where the JDK (Java) has been installed. The JDK_HOME and JAVA_HOME environment variables will be looked for and used as default if this isn't specified. If these variables don't exist, the PATH environment variable will be looked for:

```
--with-jdk-home=/usr/local/java
```

If the servlet classes in the form of the jsdk.jar aren't in your CLASSPATH environment variable, we'll need to specify where these classes are:

```
--with-jsdk=/home/jarfiles/jsdk.jar
```

We also need to tell the configuration where to install JServ when we build it:

```
--prefix=/usr/local/jserv
```

Now we run the configure script with these parameters:

```
./configure --prefix=/usr/local/jserv --with-apache-
src=/usr/local/apache_1.3.12 \
--with-jdk-home=/usr/local/java –with-jsdk=/home/jarfiles/jsdk.jar
```

This should successfully configure JServ for us.

Step 4

We now need to compile and install JServ by running the "make" command. This compiles all the source code binaries, but it doesn't install the application. To install we need to run the "make install" command. This takes the binaries created by "make" and installs them in the directory we specified with the --prefix option in the configuration script. We can run both of these commands together:

```
make; make install
```

Congratulations! You now – hopefully – have JServ installed. If you have any errors, read the output from the "make" and the "make install" commands carefully to see if it identifies the problem. If it doesn't, check carefully that the parameters you passed into the configure script were correct. If you installed successfully, you'll see some instructions printed to the screen; follow them as closely as possible. The next step is a slightly more in-depth explanation of these instructions.

Step 5

We now need to compile and install Apache just as we have for JServ. We do this in the same way; we run "make" and "make install," but this time we do it from the Apache directory:

```
cd /usr/src/apache_1.3.12/

make; make install
```

Once again, "make install" should install the application in the directory specified in the --prefix option in the Apache configuration script.

Hopefully, Apache will now be installed in the /usr/local/apache directory. However, did your make fail? If it did, try the following:

```
ls /usr/local/apache_1.3.12/src/modules/jserv
```

Is the mod_jserv.c file there? If not copy it from /usr/local/ApacheJServ-1.1.2/src/c/mod_jserv.c into /usr/local/apache_1.3.12/src/modules/ jserv/mod_jserv.c

Go to the /usr/src/apache_1.3.12/src/ directory and edit the "Configuration" file. Add the following line to it:

```
AddModule modules/jserv/mod_jserv.o
```

Save this file, and from the /usr/src/apache_1.3.12/src/ directory run:

```
./Configure
```

This should ensure that the JServ module is compiled and installed when we try to "make; make install" again. Now go back to the Apache source directory:

```
cd /usr/src/apache_1.3.12
```

Run "make" and "make install" again".

```
make; make install
```

You should see the glorious sight of a message telling you that you have successfully installed Apache.

Step 6

The next thing is to add a line to the Apache configuration file "httpd.conf". Go to the directory you specified as your Apache installation directory (that was when you ran "./configure" in the Apache directory; remember the --prefix value?):

```
cd /usr/local/apache
```

Now go into the conf/ directory:

```
cd conf/
```

Edit the httpd.conf file in this directory, adding the following line:

```
Include /usr/local/jserv/etc/jserv.conf
```

Save this file and go back up to the installation directory:

```
cd ../
```

Now check that the mod_jserv module has been added to Apache:

```
./bin/httpd –l
```

Look for mod_jserv.c in the list this returns; if it's there, hurray! If not, check the steps you've already done, paying particular attention to variables being passed into the configuration scripts.

Now try to run the Apache server:

```
./bin/apachectl start
```

Hopefully you'll get a nice little message "httpd started".

Now stop the server:

```
./bin/apachectl stop
```

Edit the file /usr/local/jserv/etc/jserv.conf, and look for the following text:

```
<Location /jserv/>
 SetHandler jserv-status
 order deny,allow
 deny from all
 allow from 127.0.0.1
</Location>
```

Change this to:

```
<Location /jserv/>
 SetHandler jserv-status
 order deny,allow
 deny from all
 allow from all
</Location>
```

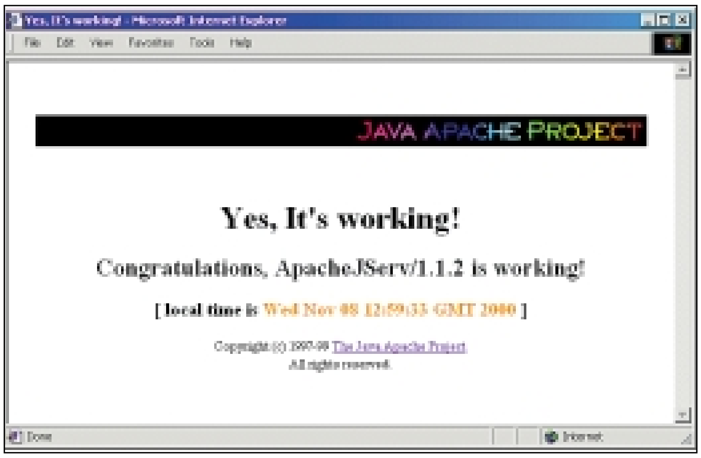
This change allows us to check the system status from a browser running from a different IP address than the Web server.

Start the Web server once again.

Step 7

Once you've completed the steps above, check the system status. The easiest way to do this is to surf to the JServ test Servlet, the HelloWorld.class Servlet:

```
http://www.yourdomain.com/servlets/IsItWorking
```

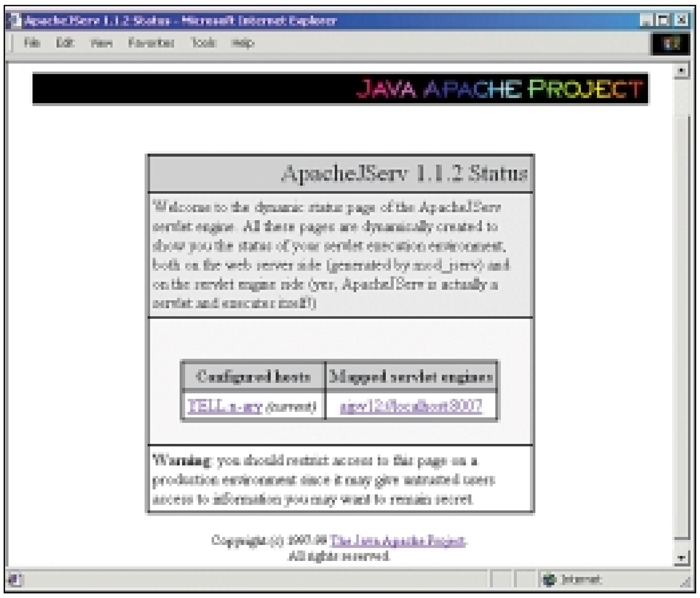

Yes, It's working! - Microsoft Internet Explorer
File Edit View Favorites Tools Help
JAVA APACHE PROJECT
Done
Internet

Yes, It's working!

Congratulations, ApacheJServ/1.1.2 is working!

[local time is Wed Nov 08 12:59:33 GMT 2000]

Copyright (c) 1997-99 The Java Apache Project.
All rights reserved.


ApacheJServ 1.1.2 Status - Microsoft Internet Explorer
File Edit View Favorites Tools Help
JAVA APACHE PROJECT
Internet

ApacheJServ 1.1.2 Status

Welcome to the dynamic status page of the ApacheJServ servlet engine. All these pages are dynamically created to show you the status of your servlet execution environment, both on the web server side (generated by mod_jserv) and on the servlet engine side (yes, ApacheJServ is actually a servlet and executor itself!)

Configured hosts	Mapped servlet engines
[illegible] (current)	ajpv12://localhost:8007

Warning: you should restrict access to this page on a production environment since it may give untrusted users access to information you may want to remain secret.

Copyright (c) 1997-99 The Java Apache Project.
All rights reserved.

If you can't surf to this page, check the "zone.properties" configuration file in the /usr/local/jserv/etc/ directory, and make sure that it has a repository pointing to where the HelloWorld.class file is sitting. This will most probably be in /usr/local/jserv/servlets/.

To view how your JServ is set up in a friendlier fashion than viewing the configuration files directly, you can surf to:

```
http://www.yourdomain.com/jserv/status
```

From here you'll see a list of configured hosts and mapped servlet engines that are currently running. If you've kept to most of the default values in jserv.conf, the configured host will be "localhost" and the mapped servlet engine will be "ajpv11://localhost:8007". Follow the link from the mapped servlet engine through to the next page and you should be met with links to the "current status" and the "root". Follow each of these links and you should be met with the friendlier-formatted details that you've placed in the configuration files.

At this point you have JServ installed. By making small changes to this setup, you can now run your own servlets. The small changes involve the "zone.properties" file in the /usr/local/jserv/etc/ directory.

In this file there are lines similar to:

```
repositories=/usr/local/jserv/servlets
```

This line tells the servlet engine to look in the /usr/local/jserv/servlets/ directory for class files to execute. If you want to execute your servlets, you need to add the directory in which you've placed your servlets to this file. For example, if I wanted to run servlets from /home/myclasses/, I'd add the line:

```
repositories=/home/myclasses
```

And there we have it. You have successfully installed and configured JServ and are able to run your own servlets. Congratulations!

Mount Points and Zones

I'd like to discuss mount points and zones for a bit. These are key areas you should know about if you want to carry out some advanced configuration of JServ.

Mount Points

What is a mount point? It's a virtual file space on the Web server that servlets can be called from. Still not clear? Well, let's go through it with some examples.

You already have a couple of mount points set up in your installation of JServ. If you look in the "jserv.conf", which is in the directory /usr/local/jserv/etc/, you'll see the directives:

```
ApJServMount /servlets /root
ApJServMount /servlet  /root
```

These directives are creating two mount points, both pointing to "/root". So what does this mean? Simply that two virtual file spaces have been created:

```
http://www.yourdomain.com/servlet/
```

and

```
http://www.yourdomain.com/servlets/
```

Okay, so what does this do for me? Remember when you ran the test servlet HelloWorld? You surfed to:

```
http://www.yourdomain.com/servlets/IsItWorking
```

and the IsItWorking Servlet was executed. You can go to:

```
http://www.yourdomain.com/servlet/IsItWorking
```

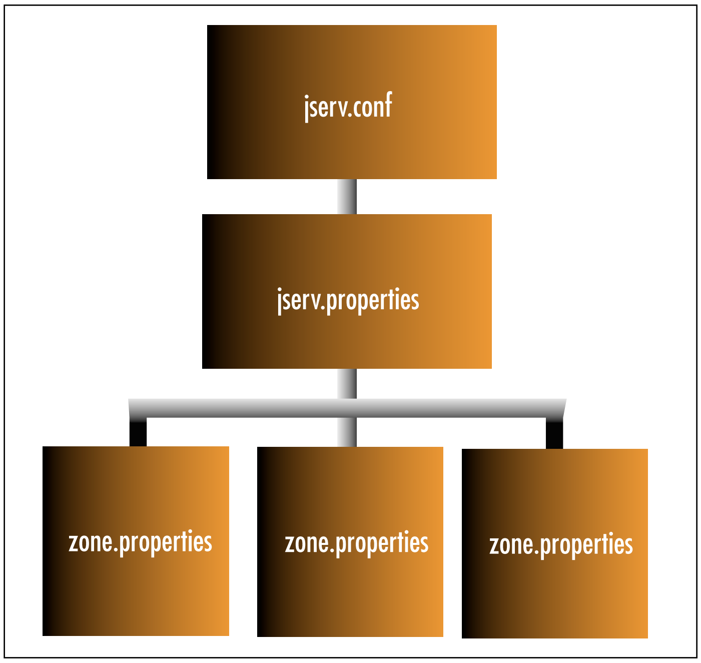
and you should see the same page that you saw from the previous URL.

How does this work? Well, if you look at the "jserv.conf" file once again, you'll see that the two mount points, "/servlets" and "/servlet," are both mapped to "/root." This "/root" is what is called a *zone*.

Zones

Just as a mount point is where a servlet is called from, a zone is where a servlet resides. The mount point has no concept of where to get a servlet from on its own. It requires a zone to hold the information pertaining to where the servlet is sitting on your system.

When we looked at the "zone.properties" file earlier, we were looking at the configuration file for the "/root" zone. This contains all the information needed for a servlet to execute – specifically, where the class files are on the system. The directive that controls this is the "repositories" directive I discussed above – a list of paths to the class files required for execution.



So we have our mount point and we have our zone, and through the zone we know where to find our servlets. We now need to link the two.

Linking Mount Points and Zones

To do this we have to look at the "jserv.properties" file that's also in the /usr/local/jserv/etc/ directory. This file contains a fair amount of information, but the part we're interested in here is the zone parameters.

Go through the file until you see the directive below:

```
zones=root
```

This directive lists the available zones to be mapped onto mount points in the "jserv.conf" file. At the moment we only have the "root" zone mapped; if you added any more, you'd just create a comma-delimited list of zones. For example:

```
Zones=root,otherzone,yetanotherzone
```

We now need to tell the servlet engine where to find the configuration files for these zones. You'll see the following line:

```
root.properties=/usr/local/apache/conf/zone.properties
```

This tells the servlet engine where and in what file a zone's configuration is kept. The syntax is:

```
<zone name>.properties=/path/to/file/containing/configuration.properties
```

As you can see here, the file name doesn't need to be the same as the zone name. I like to keep them the same as it makes it easier for someone else who might need to investigate your system.

Now we have the "/servlets" and "/servlet" mount points set up; both point to the "root" zone. This zone is named and the path to the properties file containing the information for this zone has been specified in "jserv.properties".

Now the servlet engine knows which servlets can be executed from which mount points and, through each zone, where to find the class files to carry out the execution of these servlets.

Adding Mount Points and Zones

A time may come when you want to add your own mount points and zones. After the discussion above, I don't think you'll have problems, but let's take a step-by-step look at it.

The first thing you need to do is add the mount point and zone to "jserv.conf", so add the line:

```
ApJServMount /murray /muz
```

Tada! You've now created a mount point and pointed it to a zone that doesn't exist at the moment. So you have to create it. Open up "jserv.properties" and go to the list of zones. Add your zone to the list of those available:

```
zones=root,muz
```

Now let's point this zone to the correct configuration file for it. Add the following line to "jserv.properties":

```
muz.properties=/usr/local/jserv/etc/muz.properties
```

We now have our mount point set up: it knows which zone it's connecting to, and it knows where to find the configuration file for that zone. Now we have to create this configuration file.

Make a copy of "zone.properties", renaming it "muz.properties". Remove all the "repositories=" directives from this file – we're not going to use any of the paths that the "root" zone is using. Let's say, for example, I had a servlet "MurraysExample.class" sitting in a JAR file. I have created "application.jar", which in turn is sitting in my JAR files directory, /home/jarfiles. Now we create the repository directive for this class:

```
repositories=/home/jarfiles/application.jar
```

Note that as the class file is in a JAR, I have to specifically name the JAR. If it had been in the directory, I could have put:

```
repositories=/home/jarfiles
```

And that's that. You have successfully added a new mount point and zone. You can now (if the class file actually existed) surf to the following URL to execute the servlet:

```
http://www.yourdomain.com/murray/MurraysExample
```

Actions Based on File Extensions

With JServ we can tell the Web server to pass any files with a certain extension through the servlet engine before passing it on to the browser that initially requested the page. We do this by using the "ApJServAction" directive, the syntax of which follows:

```
ApJServAction <file extension> <Servlet mount point/Servlet within
that mount point>
```

If we want to activate the servlet "pageServlet" that resides in the "root" zone each time we come to a page with the file extension ".serve", we check which zone the servlet is in, "root", and then check which mount point this zone is mapped to, "/Servlets", as follows:

```
ApJServAction .serve /Servlets/pageServlet
```

So every time we come to a .serve page, we activate the "pageServlet" servlet in the "root" zone from the "/Servlets" mount point.

Passing Parameters to Servlets

We may wish to pass a parameter into a servlet or group of servlets, which is achieved by using the default arguments directive in the relevant zone file. The syntax is:

```
servlets.default.initArgs=<variable name>=<variable value>
```

This is the default init arguments directive. Say we need to pass in the name of the directory where we have application-specific initialization files. Our directive could be:

```
servlets.default.initArgs=INI_PATH=/path/to/application/files
```

Summary

We can now install and configure JServ, we can add different mount points and zones to extend the functionality of our JServ installation, we can even send files with certain extensions through the servlet engine before the Web server serves the page.

What's left to do? The first thing I'd mention is that this article doesn't cover security. Each area discussed here – Apache, JServ, JServ zones – has security considerations, all basically dealing with who can access what on a particular server.

Some smaller aspects you may want to look at are the different logging parameters that can be configured. I personally have had no problem with the default settings in "jserv.properties" so I won't try to convince you to do something I haven't done.

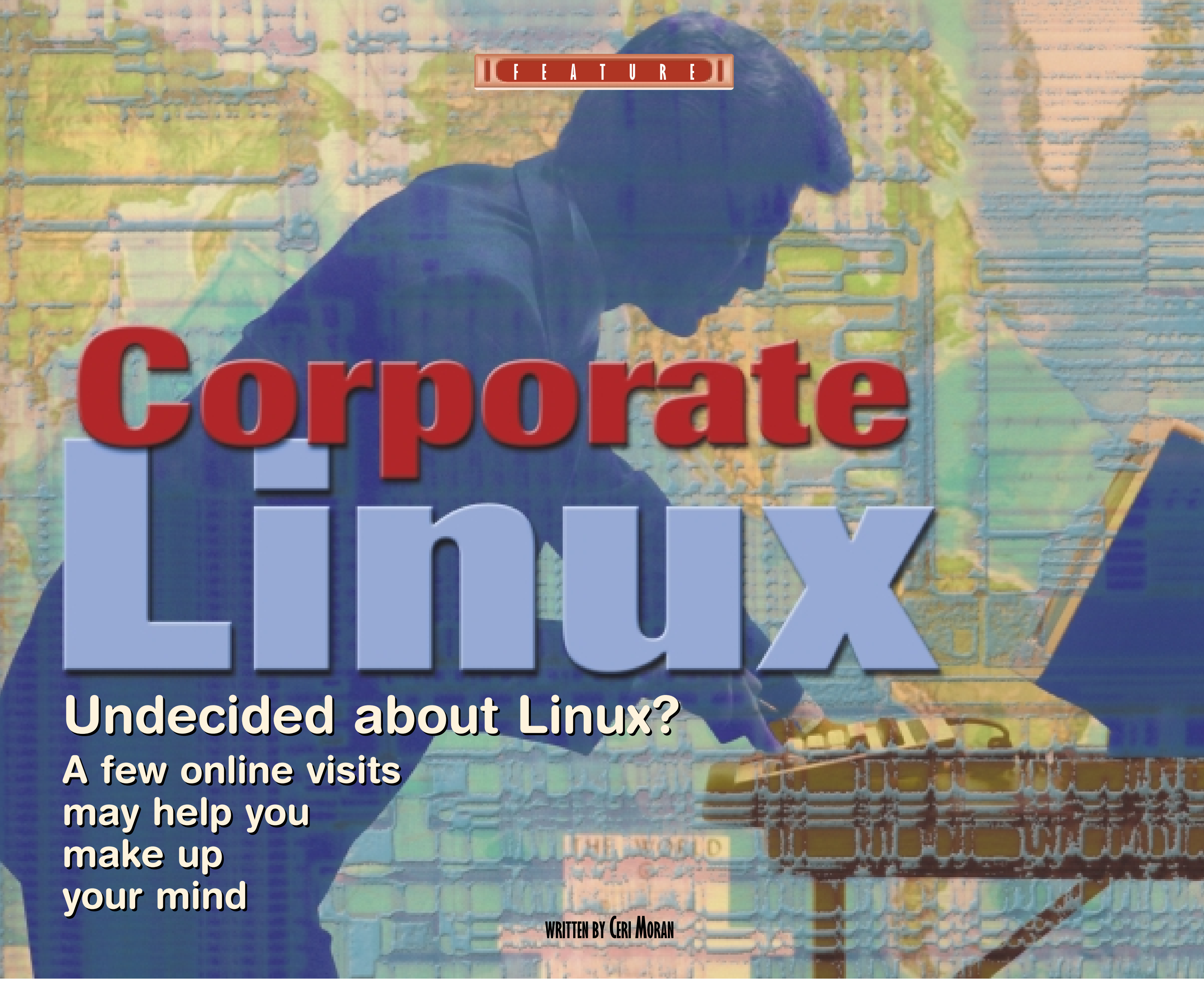
Don't be afraid to play with JServ. It's not your enemy, it's your friend. Go through the configuration files and read the documentation on the Apache Web site. Most of it is very clear and useful. But the best way to learn is to change the settings and go through the configuration files with an attitude of "What will happen if I do this?"

I won't say Hack JServ, but please don't be afraid to at least fiddle.

Author Bio

Murray Wilson, a software developer with n-ary (consulting) ltd, is one of only a few developers who began their careers with Java. He has spent the last 18 months on a number of high-profile e-commerce projects, where he quickly learned that coding Java was only half the job.

murray@n-ary.com



FEATURE

Corporate Linux

Undecided about Linux?

A few online visits may help you make up your mind

WRITTEN BY CERI MORAN

If you think Linux is the choice of geeks only, think again. Many of the large software vendors are now shipping Linux versions of their software. In this article I'll take you through some of these product offerings. Be prepared, though: if you're new to this Linux world you're going to find some interesting facts. The majority of vendors are bringing their wares to this new and exciting platform.

Hardware

Before you can begin to use Linux, you'll need to find a platform on which to run the beast. Contrary to popular belief, you're no longer restricted to the Intel–PC platform. Yes, Linux runs largely on a PC platform, but many other hardware manufacturers have recently been helping the Linux revolution come to their doorstep.

If you have a Sun UltraSPARC, you'll be able to run Linux on it. Although Sun Microsystems doesn't support Linux directly, it did help the team that ported Linux onto the UltraSPARC by providing free hardware. For more information visit www.sun.com/software/linux/ultralinux/index.html.

If you run primarily on Solaris, all is not lost if you desperately want to run Linux applications. Sun has embraced the open development project lxrun. This platform sits on top of the Solaris platform and performs various translations on the Linux–Intel executable to allow it to execute without modification. For example, it cites Linux-Quake as an example of this working-with-no-problems. Sun tried such translation layers before with its WABI product, but dropped it from their product line in the summer of 1997. For more information on lxrun visit www.sun.com/software/linux/lxrun/.

IBM, not to be left out, also offers a Linux solution to its Netfinity and X-Series of servers. Unlike Sun, IBM has completely embraced the Linux revolution. The company offers full support from installation and start-up for up to 90 days after you purchase a server. This is very good service, especially as the first steps into the Linux world can be daunting. It's good to know that one of the world's largest computer companies will be there, at the end of a phone, to assist. It's this sort of support that's helping Linux gain ground over many of the alternatives. For more information on the IBM Linux initiative go to www.pc.ibm.com/ww/eserver/xseries/linux/index.html.

If your budget is a little more modest and you don't want to go for either Sun or IBM, one of the world's largest PC manufacturers is shipping Linux-ready servers. Dell is shipping their range of PowerEdge servers with full Linux support. For more information visit www.dell.com/us/en/bsd/topics/segtopic_linux_002_linux_center.htm.

You can always convert an old PC to Linux, of course. You'd be surprised at the performance an old P233 will give you with Linux installed.

Database

Whether your Linux system is going to be used as a server or a desktop machine, chances are you'll want to look at a database solution for it. The news in this department is good. The major database vendors haven't left you out in the cold. Take the world's richest database vendor, Oracle. They can now offer you Oracle8*i* for the Linux platform. Be forewarned, though: to install Oracle you need to have both a Java and Window Manager installed. This is a little shortsighted of Oracle. Many Linux installations that are destined for a server environment don't have the x-windows installed as it just takes up disk space and unnecessary processor cycles. For more information visit http://technet.oracle.com/tech/linux/.

If you're not an Oracle fan, try Ingres from Computer Associates. They're now shipping a beta of their best-selling database for the Linux platform. The good news is that CA is offering support for the beta. Find out about it at www.ca.com/products/betas/ingres_linux/ingres_linux.htm.

Not to leave them out, Sybase is also doing their bit for the Linux platform with their Adaptive Server Enterprise database. Go to www.sybase.com/products/databaseservers/ase/ for more information.

In the original spirit of Linux, there are a number of free databases that are more than up to the job. MySQL is one of the more popular choices and shouldn't be overlooked. Go to www.mysql.com/.

With the database angle covered, let's move on to the next category.

Application Server

As I hinted at earlier, in the majority of cases Linux will end up running as a server, quite peacefully and undisturbed. Its up time is quite remarkable – in my own experience, once it's up and running you can pretty much forget about it. The majority of the application server vendors have Linux releases and I'll go through a small number of the more popular ones.

Oracle ships its application server for Linux featuring all the functionality that you've come to expect from Oracle's solution. For more information you can read the document at http://technet.oracle.com/tech/linux/htdocs/AppServer_datasheet.pdf.

IBM's WebSphere application server ships on all the major platforms including Linux. WebSphere is a fully featured application server with more functions than you can shake a stick at. Count them at www-4.ibm.com/software/webservers/appserv/.

Silverstream is another major vendor that's begun shipping Linux versions of their application server. More information at www.silverstream.com/website/staticpages/solutions/products/applicationserver/systemrequirements.html.

I have to say that I couldn't find any information on whether or not another well-known application server vendor, BEA Systems, ships their WebLogic application server on Linux. If they do, they keep the information well hidden on their Web site.

Allaire ships their JRun product on Linux in addition to all the other platforms they support. For more information visit www.allaire.com/Products/JRun/.

There's some confusion about whether Sun Microsystems ships their iPlanet server on Linux. If you go to the Sun Web site, they claim they do. However, if you head over to the iPlanet Web site, there's no obvious mention of Linux.

I can't leave a section on application servers and not mention Apache. Apache is probably one of the world's most popular Web servers, thanks largely to its stability, flexibility, and – most of all – its price: it's free! It's well worth checking out at www.apache.org/.

Java Development Kit

Of course, not all Linux installations will end up as servers. Some of you may decide to use them for development and general day-to-day usage. Linux is a good choice for a development platform, and with it – once you've got Java installed – you're well on the way. First thing you need to do is install a JDK.

There are many places you can get a JDK for Linux from. IBM ships a version that always receives favorable reviews and comments. You can find out more about it at www.ibm.com/java/jdk/118/linux/.

Sun Microsystems also ships a Linux version of the JDK. This version is strongly based on the version that Blackdown.org was developing. For more information go to http://java.sun.com/j2se/1.3/download-linux.html.

In my experience the ease of installation and stability has been very impressive. Once installed, you can pretty much forget about it. A number of vendors rely heavily on Java for their installation programs. For example, InstallShield ships an installation program for Java that can run on any Java-enabled platform. As you can imagine, a number of vendors use this tool to deploy their applications on the Linux. So it's important to get a good strong JVM that you can rely on.

Development Environment

The development tools for Linux have moved forward significantly in the last couple of years. Gone are the days of using VI and having to rely on command-line tools. The wondrous world of the IDE has finally arrived on the Linux platform.

Sun Microsystems ships Forté for Java on Linux in addition to its Solaris and Microsoft editions. See www.sun.com/forte/ffj/ce/download.html.

IBM has a very strong IDE in its VisualAge product. They have continued to embrace the Linux community right through to their development tools. For more information go to www-4.ibm.com/software/ad/vajava/vaj35platforms.html.

If, on the other hand, you entrust your development to Borland's JBuilder, you can rest easy. They provide a Linux port of their popular IDE. Go to www.borland.com/jbuilder/jb4/sysreq.html.

Sadly, I could find no information regarding Visual Café on Linux, but with IBM, Sun, and Borland all shipping Linux versions, you have a lot to choose from. This of course isn't your lot. Many of the smaller IDE vendors also ship Linux incarnations of their products.

Other...

If you're thinking of moving away from your familiar MS Windows desktop toward Linux and are concerned over losing some of your more popular desktop utilities, fear not. Granted, Microsoft doesn't offer their Office Suite on Linux, but Sun Microsystems can offer you a compatible alternative through their StarOffice product. Find out about it at www.sun.com/products/staroffice/5.2/get.html.

Summary

I've given you a small taste of the offerings from the "big boys" of computing. As you can see, the platform for geeks is no longer confined to the recesses of the hobbyist. The Linux platform offers a cost-effective and flexible solution to the world of computing. The availability of software isn't a problem anymore.

Linux has blown open the doors, with really only Microsoft choosing not to enter. History alone will show whether this is a wise or a strategically suicidal move.

If you are one of the people sitting on a fence about Linux, please – come off it and have a look. It may surprise you.

Author Bio

Ceri Moran is chief technology officer for a Java consultancy company based in the UK. She is responsible for recommending and deploying client projects.

ceri@n-ary.com

Idetix Upgrades Its Revize Content Management System

(*Troy, MI*) – Idetix Software Systems Inc. has introduced a Revize 3.0 content management system (CMS) based on a completely new and robust architecture. This major upgrade includes two new products – the Revize 3.0 Web application and the Revize 3.0 Developer's Tool. Built in 100% Pure Java the 3.0 Web Application is designed for service providers and corporations that want to reduce their total cost of ownership for Web content management. The 3.0 Developer's Tool enables any developer – regardless of skill level – to build advanced Web sites without programming language prerequisites.

www.idetix.com

Brokat Ships GemStone/J Version 4.1

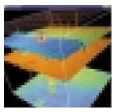
(*Beaverton, OR*) – Brokat has announced the general availability of GemStone/J version 4.1, a high-performance, scalable application server platform built on J2EE technology.

Developed by GemStone Systems, a Brokat company, GemStone/J 4.1 includes support for Entrust Technologies' Entrust/PKI 5.0 software, an e-business security infrastructure for enabling the privacy of transactions over wired or wireless networks.

www.gemstone.com

Interactive Network Technologies Releases J/View3DPro 1.2

(*Houston, TX*) – Interactive Network Technologies, Inc., has released J/View3DPro 1.2, its Java 3D toolkit that allows programmers to visualize, manipulate, annotate, and edit complex 3D scenes. New features include support for object geometry editing, additional utilities to facilitate creation of a stereo display, clipping plane, polygon triangulation, and hardcopy output.



A free 30-day evaluation of the product is available on all platforms that support JDK 1.2.2 or higher and Java 3D 1.2.

www.int.com

iPlanet and WebGain Team Up

(*Santa Clara, CA*) – iPlanet and WebGain have introduced a plug-in that integrates the iPlanet Application Server runtime environment with the WebGain Visual Café 4 Enterprise Edition and WebGain Studio development environments. The integration of these products offers organizations a fully integrated development and deployment solution based on the J2EE specification.

The plug-in for the iPlanet Application Server is available for free download from the iPlanet and WebGain Web sites at www.iplanet.com/downloads/developer/ and www.webgain.com/download/iasplugin.

RSW Software Introduces e-TEST Suite 5.0

(*Waltham, MA*) – RSW Software, a business unit of Empirix Inc., has announced the newest version of its full life-cycle Web application testing solution. e-TEST suite 5.0 combines the ease of use of a visual scripting approach with the flexibility of advanced programmability using Microsoft Visual Basic for Applications (VBA) as well as other standard languages. It consists of major enhancements to RSW's flagship product, e-Load, including UNIX-based load generation and comprehensive server-side diagnostics.

www.rswsoftware.com

Informix Strengthens Position As Leading DBMX Provider

(*Menlo Park, CA*) – Building on 20 years of database management system (DBMS) product and technology excellence, Informix Software, a database company, will focus on delivering industry-leading, cost-effective database management systems to meet the needs of today's business users in the Internet era.

To this end Informix Software has realigned its management for optimal focus on the development and delivery of leading-edge DBMS products for current and future customers. The new focus will be achieved by leveraging Informix technologies that provide a superior combination of performance, scalability, availability, and extensibility.

w.informix.com

Softletter Honors Flashline.com

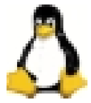
(*Cleveland, OH*) – Software industry newsletter Softletter has named Flashline one of the "Year's Ten Best Online Software Stores." The award recognizes Flashline.com as an exceptional Web site offering a wide range of component products and services for the software development community. More information on the award can be found on the Softletter site at www.softletter.com/beststore.html.

www.flashline.com

Test Version of New Linux Kernel Now Available

The next version of the heart of Linux moved a significant step closer to reality as Linux founder Linus Torvalds posted a new test version free of major bugs.

Torvalds originally wanted to release the 2.4.0 version of the Linux "kernel" in 1999, but cramming in a host of new features took longer than expected. The 2.4 kernel improves Linux's ability to run on high-end servers with several CPUs and adds support for desktop features such as universal serial bus.



SYS-CON Names Silverman VP, Marketing and Sales, Wireless Business & Technology

(Montvale, NJ) – **SYS-CON Media, Inc.** (www.sys-con.com), headquartered in Montvale, New Jersey, has named Miles Silverman as vice president of marketing and sales for its newest title, ***Wireless Business & Technology*** (www.wireless-magazine.com).

"We're delighted to have Miles Silverman on board," said Fuat Kircaali, founder and CEO of **SYS-CON**. "Miles will have a critical role in the launch of our most recent and largest new title, ***Wireless Business & Technology***."

"I'm very excited to join the **SYS-CON** team and look forward to being part of one of the biggest magazine launches of recent years," said Silverman. "***Wireless Business & Technology*** aims to be the world's leading publication serving the wireless industry and Internet technology developers utilizing new and emerging wireless technologies around the globe."

Silverman spent years managing media buying/planning for such New York ad agencies as Dentsu/Young & Rubicam and Ted Bates before embarking on his career in print publishing. His publishing track record includes a 10-year run with 101 Communications LLC (formerly SIGS Publications, Inc.), where as ad director he played a key role in the founding and publishing of two successful tiltles: *The X Journal* and *Java Report*.

WRITTEN BY STUART ZIPPER

Why Linux Lovers Jilt Java

Java and Linux should have been a natural: on the one hand we have a language that can run on any OS; on the other, an OS that can be custom tailored to a vast range of computing needs.

Should have been, would have been, could have been. Maybe some day even will be. Today, however, the failure of Java to evoke more than relatively mild interest in the Linux community is the complex result of both philosophical and technical differences between the Java and Linux communities and technologies.

The philosophical differences revolve around the terms *open source* and *free*. Java is neither, all claims to the contrary. Linux, at least in theory, is both.

Further, there's a none-to-subtle bifurcation in the Linux community that Java advocates must understand. A significant segment of the Linux community stresses the word *free*. The other half lives by different economic rules, but nonetheless is devoted to open source. At the risk of making a sweeping generalization, the bastion of free software advocates is the students and researchers of the world's universities. They're a critical piece of the Linux formula, providing a massive amount of free brainpower that's helping develop Linux and the body of software that surrounds the OS.

The commercial strength of open source and Linux lies with those who use Linux to build products for market, folks whose love of Linux is in their ability to custom-tailor every aspect of the OS to their own particular needs. These are folks who will and do pay for Linux if they have to, as long as it's open source. They're also a critical piece of the Linux formula without which Linux would be merely the stuff of computer science classes and esoteric research projects.

For the first group, the free software advocates, Java will never be popular until the day Sun releases the source code under an acceptable license, either the well-known GNU General Public License (GPL) or one very much like it. Now let's remember what GNU stands for: "GNU's Not UNIX." Then let's remember that Java comes from Sun, and Sun's forte is definitely UNIX. Let's not talk about Sun's concept of a "community" license. It doesn't cut bait with the open source community.

The dislike of Microsoft in the Linux community is, of course, legion. But spend enough time talking with Linux folks, especially the free software part of the community, and it soon becomes clear that Sun isn't popular either.

The result is that for a significant chunk of the Linux community Java is simply not an alternative. It's the wrong religion. If Sun ever does make good on its occasional hints that maybe it will GPL Java, and that's a very big if, it won't be the Linux folks changing religions.

That leaves the rest of the Linux community, the guys that don't got religion. Folks at commercial Linux companies say a large segment of their community would love to use Java but it just doesn't fit.

Where Java's missed the boat so far is in its acceptability to the embedded community. Most of the talk about Linux on the desktop is just that – talk. Linux's real strength lies in embedded widgetry of all ilks. In that market Linux is starting to make a major dent in what had been a purely proprietary OS market.

Embedded Linux developers sneer at the footprint of a Java Virtual Machine. There's simply too much code to stuff into the smaller embedded applications, they say. There's no good implementation of Java for real time, the criticism continues. Real time – be it deterministic hard real time, soft real time, "near" real time – is the holy grail of the embedded community.

Another worry of the Linux community, indeed of the embedded community in general, is the stability of the Java code base. Unlike desktop computers with a few years' life span, it's not unusual to find embedded devices that are expected to run unattended for 10–15 years. Java is simply changing too fast right now to provide the comfort level that embedded developers need.

To be sure, at the higher end of embedded, particularly anything hooked to the Internet, Java support is de rigueur. But it's grudging and doesn't go far beyond stuffing a JVM into widgetry such as Internet terminals. Linux-based Web servers, ironically, may dish up more Java than any other type of server in the world. But they don't have to run Java – that's done on the client. And Windows owns the clients.

The fact is that the Linux world starts with good old C. The more adventurous Linux developers are discovering C++ with all its warts and pimples. Many say they'd love to try Java, but they think Java tools and compilers need more time to mature. Some are even saying that time may be running out. The really avant garde are already poring over Microsoft's C#, which Microsoft fancies is a sort of super-Java blending the best of C++ and Java, and keeping their fingers crossed that it won't be a Windows-only creature.

zipper@g2news.com

AUTHOR BIO

Stuart Zipper is the senior editor of Client Server News, The Online Reporter, *and* LinuxGram. *He reports on a broad swath of the industry including Windows, Linux, and Java. Stuart has a BA in political science from Queens College of the City University of New York.*

Linux Beach Calls for Developers of a Linux PC

Linux Beach is forming a Linux PC development group to challenge Microsoft and Apple desktop computers.

The project calls for the creation of a specific "Linux distribution" by taking a closed hardware platform and a suitable software mix that will characterize it as "The Linux PC." This will result in a Linux box that will boot up to a simple GUI interface that's easy to work with and understand.

www.cosmoseng.com

Self Test Software Releases i-Net+ Practice Test

(*Atlanta, GA*) – Self Test Software has announced its third CompTIA practice test. In addition to tests for A+ and Network+ that enable career changers to enter the field of information technology, their new i-Net+ practice test now enables candidates to prepare for CompTIA's new Internet certification. With more than 650,000 new jobs created in 1999 as a result of the Internet economy, according to a study from the University of Texas, the demand for certified Internet professionals is growing exponentially.

www.selftestsoftware.com

Storm Linux 2000 Promotes Deluxe Edition

(*Vancouver, BC*) – Stormix Technologies Inc. has launched Storm Linux 2000 Deluxe Edition. It features a comprehensive selection of applications, utilities, and packages such as Storm Administrative System 2.0, a suite of utilities designed to make it easy for new Linux users to administer their system, and Storm Package Manager, a graphical user interface for automatically configuring, installing, and updating Debian software packages.

www.stormix.com

IBM to License Linux Mobile Middleware

(*White Plains, NY*) – IBM is making commercial licenses available for its Linux-based mobile middleware BlueDrekar on alphaWorks, IBM's online resource for emerging technology. BlueDrekar is IBM's first Linux-based middleware based on the Bluetooth specification for connecting devices wirelessly.

www.alphaWorks.ibm.com

Intuitive Announces Optimizeit 4.0 for Linux and Sun Solaris

(*Cupertino, CA*) – Intuitive Systems, Inc., has announced the availability of Optimizeit 4.0 for Linux and Sun Solaris Sparc.

Optimizeit 4.0 allows developers to test and improve the performance of any Java program including servlets, Enterprise JavaBeans, JavaServer Pages, and complex server-based Java applications. New features include universal JVM support, offline profiling for testing of programs in production environments, and integration with even more application servers.

www.optimizeit.com

Allaire Deepens Java Commitment

(*Newton, MA*) – Allaire Corporation has acquired the Kawa Java integrated development environment (IDE) from Tek-Tools, Inc. Its addition to the Allaire Business Platform gives Allaire the ability to offer customers powerful visual tools for building advanced Web applications on the industry-standard J2EE platform. The enterprise development capabilities of Kawa complement Allaire JRun Studio 3.0, which is targeted at JavaServer Pages (JSP) and Java Servlet development.

www.allaire.com

Borland Acquires E-Services Firm

(*Scotts Valley, CA*) – Borland has completed the acquisition of Chicago-based Bedouin, Inc., and has created a fourth business unit, Borland Developer Services.

Borland Developer Services will leverage Bedouin's expertise in building and deploying e-services to deliver a new platform that will allow customers to build, deploy, and manage applications via the Internet. Bedouin's product management and software development team will form the core management of this new division.

www.borland.com

Esmertec Releases Jbed Micro Edition CLDC

(*Zurich, Switzerland*) – Esmertec, Inc., released its Jbed Micro Edition CLDC, the fastest small-footprint Java Virtual Machine designed for memory-constrained devices such as personal digital assistants, mobile phones, and Internet appliances.

Jbed Micro Edition CLDC is initially available for Palm OS devices or on the bare metal on 68k-, PowerPC-, or ARM-based embedded systems.

Esmertec has also released Jbed RTOS Package 1.3, which offers new features and enhancements to its core industries.

This updated release incorporates major enhancements in the Jbed TCP/IP networking performance and additional protocols such as PPP, DNS, and daytime protocol.

Enhancements to the Jbed IDE and new boot loaders are part of the new release as well. Additional functionality for Interrupt Service Routines (ISRs), which allows notification from an ISR to a waiting real-time task, has been added.

www.esmertec.com

NQL Releases Network Query Language Java Edition for Linux

(*Santa Ana, CA*) – NQL Inc. has announced the full release of a Java Edition of its Network Query Language technology, which it has qualified for the Linux operating system.

Network Query Language is NQL's core technology, a scripting language suited for the development of bots, intelligent agents, and content management solutions with capabilities in communications, data conversion, automation, and intelligent behavior. Capabilities offered in the Linux version match those of the Windows version and contain essentially the same deployment features.

www.nqli.com

Halcyon Software Announces Free ASP Support

(*San Jose, CA*) – Halcyon Software, Inc., is making Instant ASP (iASP) available for free to users of Intel-based Linux systems. iASP is a Java implementation of the industry-standard Microsoft Active Server Page (ASP) framework and will allow businesses and individuals a cost-effective way to create and run interactive Web sites.

iASP includes full support for ADO and CDO and interfaces with multiple component architectures, including JavaBeans, Enterprise JavaBeans, and CORBA components. In addition, it can also be tightly integrated with popular application servers including IBM WebSphere, Sun iPlanet, Oracle Application Server, and BEA WebLogic.

Instant ASP for Linux can be downloaded from www.halcyonsoft.com/products/chooseproduct.asp.

Caldera Systems Names New CFO

(*Orem, UT*) – Caldera Systems Inc. has named Robert Bench as chief financial officer. Prior to his appointment, Bench was president and CFO of WebMiles.com.

www.caldera.com

Interview...with Tony de la Lama

VP and General Manager, Java Business Unit, Borland

Interviewed by David Johnson

From an interview at JavaCon 2000 in Santa Clara, California

JDJ: The latest version of JBuilder is being released today, so that's exciting news. Tony, why don't you go ahead and talk a little bit about what Borland is doing with JBuilder and also with Java.

de la Lama: We're so excited about JBuilder 4, and this is a great place to really reveal it to the public for the first time. We have some interesting features that I think are really going to take JBuilder 4 to the next level for development of Java applications. For starters we have a new team development solution that's based on the CVS [Concurrent Versions System] repository. You're probably familiar with our heritage of having JBuilder run on Solaris, Linux, and Windows – all three platforms. We wanted something that would work and scale across the Internet and run across many different operating systems. The CVS-based repository we've chosen is fully integrated into the product. It's a very good technology – one of the main features we want to let people know about in JBuilder 4.

I also want to mention a couple of other features related to debugging. Some folks may not know that we have full JSP debugging with JBuilder 4. You can actually set a breakpoint in JSP when you get there, and view the variables. From there you can go ahead and jump off into an EJB that may be tagged in the JSP itself. It's an interesting environment for debugging.

We also have support for Servlet 2.2 and JSP 1.1 – the latest standards in both those specs – and on top of that, we now include Tomcat and Apache. If you look at JBuilder 4 as a whole package you now get at the enterprise level, you have Borland application server, which includes VisiBroker. You get a JSP runner, and a servlet runner, and Tomcat. So you have a really good environment that you can actually build right out of the box, debug, deploy locally, and test your EJBs. We're very excited about the success in this environment.

“… some interesting features … are really going to take JBuilder 4 to the next level for development of Java applications ”

JDJ: Maybe you could talk a little more about the different features, wizards, code generation facilities in JBuilder, and what differentiates it from some of the competing products.

de la Lama: We get what we call the EJB modeler to create container management persistence – EJB management persistence. By default we'll go ahead and create the whole EJB interface for you for the primary classes. That really saves you a lot of time. Also with the JBuilder 4 release we're going to be supporting WebLogic for the first time. So you'll be able to build EJBs, deploy them, run them, and debug them on the WebLogic server from BEA. So now we actually support two application servers out of the box – our own Borland application server and WebLogic. Also one of the features that customers are really excited about is the ability to deploy, or actually hot deploy, onto WebLogic. You'll be able to actually drop your EJB without having to restart the server, and it will pick up where it left off. It saves a lot of time in your development.

JDJ: Now with CVS, it's obviously something you can do over the Internet. Is that encrypted and secure?

de la Lama: That's a good question. Just the other day I was working with a few of my developers. We were talking about an open-search project based on CVS and were kind of interested in taking a look at the code for this. So we fired up JBuilder, pointed JBuilder to create a project, then pointed it to the repository, put in the URL and the password, and within minutes we were downloading the project right inside JBuilder. Five minutes later, with one button, we were compiling the whole project. That was very cool, because there are so many different open-source projects out there based on CVS. I think it opens up a lot of new code relatively quickly from inside JBuilder.

JDJ: Right, the alternative is probably command line, which can be a little difficult with newer developers. Now I see the comments on open database support. What are some of the database features in JBuilder?

de la Lama: We're especially proud of our database support inside JBuilder. Borland has a long heritage of building really solid development environments for database development, and we created something about three releases ago called DataExpress. It's a great framework to build your applications on top of. It'll insulate you from the rudimentary JDBC type of coding that most people have to do. So using DataExpress and the functionality it provides, we also give you something called JDataStore, and we're up to version 4, which is an all-Java database. It's a small footprint, fully object-relational database that allows you to build very solid applications locally. So we're really excited that we could include this functionality in our Professional version as well as our Enterprise version. And in the Enterprise version you get the actual source code to our framework for DataExpress. It's a really solid value, we think, in database programming.

JDJ: I imagine that people can check out an evaluation copy of JBuilder. Also, maybe you could talk about your Web site and what's available there.

de la Lama: We used to sell a version of JBuilder called JBuilder Standard. It cost $99 and would build a core function – editing, debugging, compiling, and project management use. But now we've come out with something called *Foundation,* which, we give away for free. So the CD that's distributed at our booth is actually a fully functional version of JBuilder that does not expire and is not limited by classes. It just doesn't have the database and enterprise features of our Professional and Enterprise versions. It runs on Solaris, Linux, and Windows, and has some swing components included. It has a high performance compiler and other technology already in it. If you don't need anything more than that, just try Foundation and enjoy it – go out and have a good time and build your applications! If you need additional functionality, check out our Professional and Enterprise editions.

Author Bio

David Johnson is CEO of Verge Technologies Group, Inc., a Boulder, Colorado- based Enterprise Java consulting and hosting firm.

djohnson@vergecorp.com