Oracle9*i* JDeveloper Preview CD Inside!

JAVA DEVELOPER'S JOURNAL

The World's Leading Java Resource

December 2001 Volume: 6 Issue: 12

JAVADEVELOPERSJOURNAL.COM

SYS-CON MEDIA

JAVA DEVELOPER'S JOURNAL

INTERNATIONAL ADVISORY BOARD

• CALVIN AUSTIN *(Lead Software Engineer, J2SE Linux Project, Sun Microsystems)*, • JAMES DUNCAN DAVIDSON *(JavaServlet API/XMP API, Sun Microsystems)*, • JASON HUNTER *(Senior Technologist, CollabNet)*, • JON S. STEVENS *(Apache Software Foundation)*, • RICK ROSS *(President, JavaLobby)*, • BILL ROTH *(Group Product Manager, Sun Microsystems)*, • BILL WILLETT *(CEO, Programmer's Paradise)* • BLAIR WYMAN *(Chief Software Architect IBM Rochester)*

EDITORIAL

EDITOR-IN-CHIEF: ALAN WILLIAMSON
EDITORIAL DIRECTOR: JEREMY GEELAN
J2EE EDITOR: AJIT SAGAR
J2ME EDITOR: JASON BRIGGS
PRODUCT REVIEW EDITOR: JIM MILBERY
FOUNDING EDITOR: SEAN RHODY

PRODUCTION

VICE PRESIDENT, PRODUCTION AND DESIGN: JIM MORGAN
ASSOCIATE ART DIRECTOR: LOUIS F. CUFFARI
EXECUTIVE EDITOR: M'LOU PINKHAM
MANAGING EDITOR: CHERYL VAN SISE
EDITOR: NANCY VALENTINE
ASSOCIATE EDITORS: JAMIE MATUSOW
GAIL SCHULTZ
BRENDA GREENE
ASSISTANT EDITOR: LIN GOETZ
TECHNICAL EDITOR: BAHADIR KARUV, PH.D.

WRITERS IN THIS ISSUE

BILL BALOGLU, JASON BRIGGS, JAMES CAPLE, GLEN CORDREY, JEREMY GEELAN, ABRAHAM KANG, GIORA KATZ-LICHTENSTEIN, DONALD LEVY, VINCENT MACIEJEWSKI, CHRIS MAIR, BILLY PALMIERI, AJIT SAGAR, ALAN WILLIAMSON, BLAIR WYMAN, JOHN ZUKOWSKI

SUBSCRIPTIONS:

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

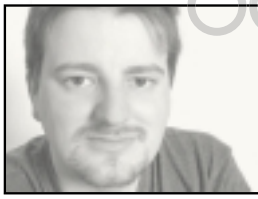
SUBSCRIPTION HOTLINE: SUBSCRIBE@SYS-CON.COM

COVER PRICE: $5.99/ISSUE
DOMESTIC: $49.99/YR. (12 ISSUES)
CANADA/MEXICO: $79.99/YR. OVERSEAS: $99.99/YR.
(U.S. BANKS OR MONEY ORDERS). BACK ISSUES: $10/EA., INTERNATIONAL $15/EA.

EDITORIAL OFFICES:

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

JAVA DEVELOPER'S JOURNAL (ISSN#1087-6944) is published monthly (12 times a year) for $49.99 by SYS-CON Publications, Inc., 135 Chestnut Ridge Road, Montvale, NJ 07645. Periodicals postage rates are paid at Montvale, NJ 07645 and additional mailing offices. POSTMASTER: Send address changes to: JAVA DEVELOPER'S JOURNAL, SYS-CON Publications, Inc., 135 Chestnut Ridge Road, Montvale, NJ 07645.



ALAN WILLIAMSON EDITOR-IN-CHIEF

Back to Work with Ant!

Chestnuts roasting on an open fire, Jack Frost nipping at your nose…damn, isn't this the best time of the year! Ironically, the actual event on the 25th isn't that memorable, but the lead up to this day is what gets me all fired up. And this year, I have two special celebrations.

First and foremost, my son is in a state of motion – he'll be a massive 1.06-years-old (I bet you can tell I'm a developer can't you?). Second, this will be my first Christmas as editor-in-chief of ***JDJ*** and I'll be celebrating six months in this role. When I say things like that, I realize how much I sound like my parents and resign myself to the fact that I *am* getting older. Ho hum.

After six months I'm beginning to understand the pressures and responsibilities of being editor-in-chief. It is fun, that I cannot deny. I love working with our wonderful team of editors (yes, even working with Mr. Briggs, our esteemed J2ME editor!), and between us, we're shaping ***JDJ*** into the magazine you want to read. My e-mail traffic has definitely gone through the roof and to this end, I thank each and every one of you for e-mailing me your comments, good or bad. I do answer all of them eventually, so keep them coming.

In November we took to the road and began what we have affectionately named the ***JDJ*** *World Tour*. (I'm keen to learn more about the Java vendors and how they see the future of Java.) Where else to start but in San Francisco. Over the space of a week we visited many of the top Java vendors and listened at length to what they had to say. It was very interesting to see who mentioned Microsoft and who didn't; who mentioned Web services and who didn't. I learned a lot and I'm looking forward to the next leg of our tour when we visit some of the companies based on the East coast. It's too soon to make any real conclusions as I'm still processing all the information.

Recently at n-ary we took a bold step and took away all the IDEs from our Java developers. For a long time I've been listening to rave reviews of tools, such as Apache's Ant, and decided to take the plunge to see what all the fuss was about. The notion of a "make" file for Java wasn't sitting too well with me; I couldn't see the need. But now I'm a convert! We love it. Building our flagship product tagServlet proved to be so much easier to coordinate using Ant. So with that, we've all gone back to a basic text editor (EditPlus at www.editplus.com), which does nothing more than simply syntax highlight our Java code; we manage the projects with Ant, including all CVS control.

The biggest thing we've all noticed is the speed of development. No more waiting for the IDE to boot up and no more large memory consumptions. It's an absolute joy to get back to the way code used to be developed, before all those fancy windows and wizards came in and pretended to do the work for us. With respect to documentation we haven't lost anything there either; we trained our office in IRC-BOT, Cormac, to deliver on demand the official JavaDoc information on a particular class. One major benefit of this move back to basics is the ease and speed with which I can move my development onto my relatively slow laptop and work remotely.

I feel in control again. I feel like a real developer again. It's wonderful!

And on that note, have a good Christmas, and I look forward to hearing from you.

alan@sys-con.com

AUTHOR BIO

Alan Williamson is editor-in-chief of Java Developer's Journal. *During the day he holds the post of chief technical officer at n-ary (consulting) Ltd, one of the first companies in the UK to specialize in Java at the server side. Rumor has it he welcomes all suggestions and comments!*

Where Are the Listings?

First off, good magazine! I'm relatively new to Java, after working for many years with (and finally giving up on) Microsoft technologies. ***JDJ*** has a good mix of beginner and expert articles so I can quickly get up to speed on the wide range of technologies out there. I hope you keep an overall expert/cutting-edge level as well as some occasional beginners material in there.

My only gripe and major complaint is that most listings are available online only. I understand that you do this to save space, but I tend to read magazines like this on the bus or underground on the way home from work. As a professional developer in a finance company, I don't have time to sit beside a PC at work and read the articles (unless it's to develop a specific bit of code); when I come home after staring at a screen for 10–12 hours, I don't want to log on and read online or have to print the article.

More important, I bought the magazine in Australia (where perhaps the publishing schedule is out of date as I'm only reading Vol. 6, issue 6 now). I'm now online to print the code of an otherwise unreadable article, "The Impact of EJB 2.0" by Tyler Jewell, which continuously refers to the listings. I have gone online to ultimately print these out and it's available only to subscribers! First, it was hard to find this out as the listings were not in the source code section of Vol. 6, issue 6. I only found it by an author search, which told me it was for subscribers only, then referred me to the magazine subscription page.

The fact that I purchased ***JDJ*** and am unable to read the listings for the articles is an unacceptable situation.

Mark Lamont
lamontmark@hotmail.com

From the editor:

Thanks for your note. We are working on a new system to make the source code more accessible. Watch this space.

Alan Williamson

Why Java?

This is my fifth year as a software developer. What I hate most is rewriting the same routines again and again – one time you're writing it in VB on Win32, another time you're working in C on an embedded system, and so on!

I like to do new things, to investigate new solutions. At this time I'm developing embedded software and am very interested in J2ME. I'd love to develop my applications on a workstation and then see it run on my embedded system regardless of the underlying hardware!

Yes, I know Java is sometimes slow... Well you can compile it! There is a GCC front-end for doing that.

Mirco Bova
mirco.bova@milmail.com

Visual J#.NET

I guess Microsoft is trying to jump on the Java train again since Java is now one of the major players in the software development process. I've been working in the insurance industry for quite a while and noticed that Java is used most of the time in new application development. There are exceptions, for example, if new applications are pure mainframe and are developed in Cobol or PL/1. It's the same in the banking industry. Since both the insurance and the banking industries are the *big* customers for every software vendor, it's obvious that Microsoft has to support Java, but doing it the way it is now – running only on the .NET platform, not supporting the Java standards defined in the Community Process – I think Microsoft is shooting itself in the foot.

Gregor Schneider
gregor@2smart4u.de

Comments on Vince Bonfanti's Editorial

I think Vince Bonfanti's editorial, "J2EE Without EJBs?" (***JDJ***, Vol. 6, issue 11), is dead-on. So many times projects get warped by marketing bull. Web development is already expensive and swaying customers toward an "enterprise" server that they don't need is ridiculous. I agree that there are great reasons to use EJB solutions, but a lot of solutions can be handled without them and, yes, they are J2EE compliant.

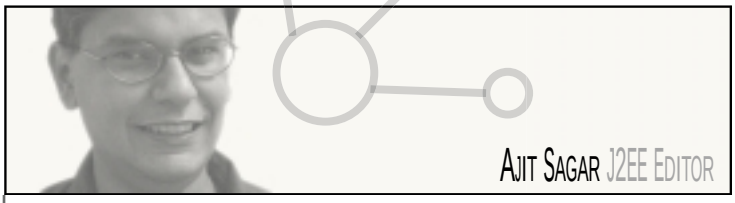
Richard Dean
richarddean@yahoo.com

Maybe EJB was a marketing tool exploited by the vendors. The problem is that system architectures and designers also fell for the trap and instead of giving their customers the best value for their money by using JSP/Servlets and JDBC, they went for EJBs, which resulted in the loss of time, effort, etc. Not to say that EJB shouldn't be used; however, it has its own place and purpose.

Lalit Popli
lpopli@consultant.com

We've built large applications without a single EJB – we used JSPs/Servlets with logic encapsulated in JavaBeans. I think the most important factors that make EJBs almost mandatory from a design perspective are the transactional nature of the application and the handling of distributed enterprise resources (multiple databases, messaging, ERPs, etc.). I believe the use of EJBs are justified even if these two factors are a medium to long-term likelihood for a project.

Chandika Mendis
chandika@java-shop.com



AJIT SAGAR J2EE EDITOR

We've Got It All...

To help cut the cost of travel in today's economy, I flew ATA from Philadelphia to San Francisco last weekend. You know, if you live within the constraints defined by these airlines, it's not really a bad way to go. Of course, you travel in a full plane (and I mean packed to the brim), can't change your itinerary, spend an extra hour or so at the connection point, and so on. However, you end up paying less than you would at the major airlines, especially with a last-minute booking.

With the current state of the economy in the technology sector, one of the markets that has been hit pretty badly is the J2EE application server market. Nowadays, companies are very cautious about the cost involved in embedding a product with a large sticker price in their product. Especially when that product costs several thousand dollars per CPU.

There are three main outcomes from the downturn in the economy in this market. One, several small vendors are now out of business, or have been acquired by other vendors who were looking to expand their footprint. Another outcome is that the vendors who have survived are all offering more or less the same development and deployment platforms. The third is that a lot of businesses are taking a serious look at open source alternatives.

A part of the soul-searching that companies have undergone involves determining how much of an "enterprise" product they're selling. With fewer customers for large cross-enterprise B2Bi requirements, more companies are focusing on intraenterprise EAI functionality. Many of these applications don't need the capabilities of a full-blown application server. For example, the EJB market is one that's being evaluated very carefully. Many applications don't require the full power of a heavy and complex middle tier.

There are alternatives to building business objects as EJBs; for example, going directly from the servlet/JSP layer to a middle-tier Java application that connects to the data source – you still get the benefits of a three-tier application. The Web container you choose will probably support clustering and session management. Apple's WebObjects is a great product if you want to work in a pure Java object environment, but it has minimal support for J2EE.

Another trend that's becoming popular for back-end EAI applications is the concept of e-business messaging. In applications that don't require synchronous real-time responses, JMS messaging is a good alternative to using a full-blown application server environment. Several pure JMS providers are emerging in the market, including Sonic, Fiorano, Talarian, and SpiritSoft. The JMS environments provide the basic capabilities of an application server such as clustering, load balancing, transaction management, and security. And they provide back-end connectivity to legacy systems or Enterprise Information Systems (EIS).

Market leaders in the J2EE application server space such as BEA, IBM, Oracle, HP, and Sun are targeting large corporations that have a need for the complete J2EE offerings, EJBs included. If you look at the product suite offered by these vendors, the functionality is more or less the same: J2EE development and deployment, transaction processing monitors, commerce and portal servers, personalization engines, workflow engines, and more. The smaller players are now following the strategy of sticking to their niche and providing an integration story with the larger players.

The last part of the puzzle is the actual hardware that hosts the products. While it's a noble thought to be cross-hardware portable, this portability comes at a cost and added complexity. Commoditized application servers that run on specific platforms are more affordable since a large part of the installation, training, and development complexity is avoided by companies investing in the solution; case in point – Microsoft.

Hence, in the next few years we should see a consolidation of the hardware and software offerings from the larger vendors. IBM, Sun, and HP are well positioned to achieve this with their application server offerings, which can be easily coupled with proprietary hardware. In fact, with the HP-Compaq merger, HP may be the only company uniquely positioned to provide embedded J2EE capabilities across Unix and Microsoft platforms. How well that integration goes remains to be seen.

I will avoid flying ATA in the future, since they don't suit some of my primary requirements. But similar to the WebObjects community, there are a lot of passengers out there who benefit from the features ATA offers.

ajit@sys-con.com

AUTHOR BIO

Ajit Sagar is the J2EE editor of JDJ and the founding editor and editor-in-chief of XML-Journal. A senior solutions architect with VerticalNet Solutions, based in San Francisco, he's well versed in Java, Web, and XML technologies.

J2EE INDEX

J2EE FAQ

WHAT IS THE CURRENT VERSION OF THE JAVA 2 ENTERPRISE EDITION?

The Java platform versioning is quite confusing. This is even more true since Sun split the original platform into three editions. Although it was a good step for decoupling application development, it has led to more effort in terms of version management.

To answer the question, Sun released Release 1.3 of the enterprise edition (J2EE) on September 24, 2001. This is different from Release 1.3 of the JSE (Java Standard Edition), which was released more than a year ago. Currently the latest version of J2SE is 1.3.1 and the 1.4 version is in the beta release stage.

There's a dependency between the editions. J2EE uses the J2SE libraries, therefore the J2EE released versions need to be compatible with the latest J2SE versions. If you get to the granular APIs, they have their own versioning. For example, the J2EE Connector Architecture is in Release 1.0 and the Servlet API is in Release 2.3

Java 2, the current release of the Java platform, is the umbrella that encompasses the latest versions of all the API editions mentioned above. So the J2ME, the J2SE, and the J2EE APIs are all a part of the Java 2 platform.

CAN AN RMI SERVER RUN ON A JAVA CLIENT?

Yes, it can. The main components of an RMI-based application are an RMI client, RMI server, and RMI registry. The RMI server can run on a machine that hosts a VM that supports the RMI library. Support for RMI is required of all Java VMs.

In the traditional client/server scenario, a Java client application connects to a Java server application; the Java server application does the bulk of data processing and business logic on the server side. If you aren't connecting via the HTTP layer, i.e., your connectivity isn't based on access through a Web server via servlets/JSPs, you'll probably use RMI for connectivity. Typically, the RMI server will run as a part of your Java server application. That is, the implementation of the RMI-based interfaces for your application will be on the Java server application.

However, this isn't necessarily true. If you wanted to use RMI for some processing on the Java client and return the results to the Java client, you can achieve that also. In this case the RMI server will run on the Java client and the RMI client will run on the Java server. An example of this is when you want to push data to the Java client and invoke a method on the client to process that data on the Java client-side after receiving it from the Java server.

IS EJB A SPECIFICATION OR AN ACTUAL JAVA COMPONENT?

Both. The EJB specification defines the design and implementation for Enterprise JavaBeans, which are Java components. An interesting point to note in documents covering EJBs is that both the singular and plural word form is used to refer to EJBs. For example, Enterprise JavaBeans is a specification. Enterprise JavaBeans are Java components.

IS EJBOBJECT A REMOTE OBJECT?

Actually EJBObject (javax.ejb.EJBObject) is an interface that extends java.rmi.Remote interface. Note that an "EJBObject" interface and an "EJB Object" class (notice the space) are two different entities. An "EJB Object" class is an actual class that is autogenerated by the application server. The "EJB Object" object is a delegator object that exposes all the methods of your actual EJB (entity/session/message-driven) implementation.

When developing an EJB component, the first thing you need to do is define an interface (e.g., MyInterface) that extends the javax.ejb.EJBobject interface. In doing so, you'll add your business logic methods to the methods already defined in the EJBObject interface. These methods are used by the container to manipulate your EJB component – methods to find the bean, get a reference to the bean, etc. During deployment the container will generate a class that provides an implementation for all the methods for MyInterface, including delegator classes for the methods that you define. This class is your "EJB Object" class implementation of your "MyInterface" interface.

J2EE ROADMAP

The Java 2 Platform, Enterprise Edition defines the APIs for building enterprise-level applications.

- **J2SE**.....v. 1.2
- **Enterprise JavaBeans API**.....v. 1.1
- **Java Servlets**.....v. 2.2
- **JavaServer Pages Technology**.....v. 1.1
- **JDBC Standard Extension**.....v. 2.0
- **Java Naming and Directory Interface API**.....v. 1.2
- **RMI/IIOP**.....v. 1.0
- **Java Transaction API**..v. 1.0
- **JavaMail API**.....v. 1.1
- **Java Messaging Service**.....v. 1.0

Useful URLs:

Java 2 Platform, Enterprise Edition
www.java.sun.com/j2ee/

J2EE Blueprints
www.java.sun.com/j2ee/blueprints

J2EE Technology Center
http://developer.java.sun.com/developer/products/j2ee/

J2EE Tutorial
http://java.sun.com/j2ee/tutorial/

Security

for J2EE Applications

Implementing a single sign-on

Written by

Donald Levy

&

Abraham Kang

Part 2 of 2

Single sign-on is becoming an important issue for corporations and Java developers. Corporations require applications to be secure. Users demand applications to be easy to use. Usually, the more secure an application is, the more difficult it is to use. For example, users are inundated with remembering multiple logins to different systems.

Part 1 of this article (***JDJ,*** Vol. 6, issue 11) provided readers with a conceptual understanding of single sign-on (SSO). In Part 2 we explain the programming and setup issues related to SSO for Web and standalone applications. With this knowledge, you'll be able to implement SSO solutions for your applications.

Balancing Security with Easy Usage

It was noted that more than 45% of calls to corporate help desks involve user ID or password-related issues. To avoid memorizing multiple logins users put their passwords in insecure locations (like their monitors).

SSO addresses these issues by creating a single-user ID and password that can access multiple applications.

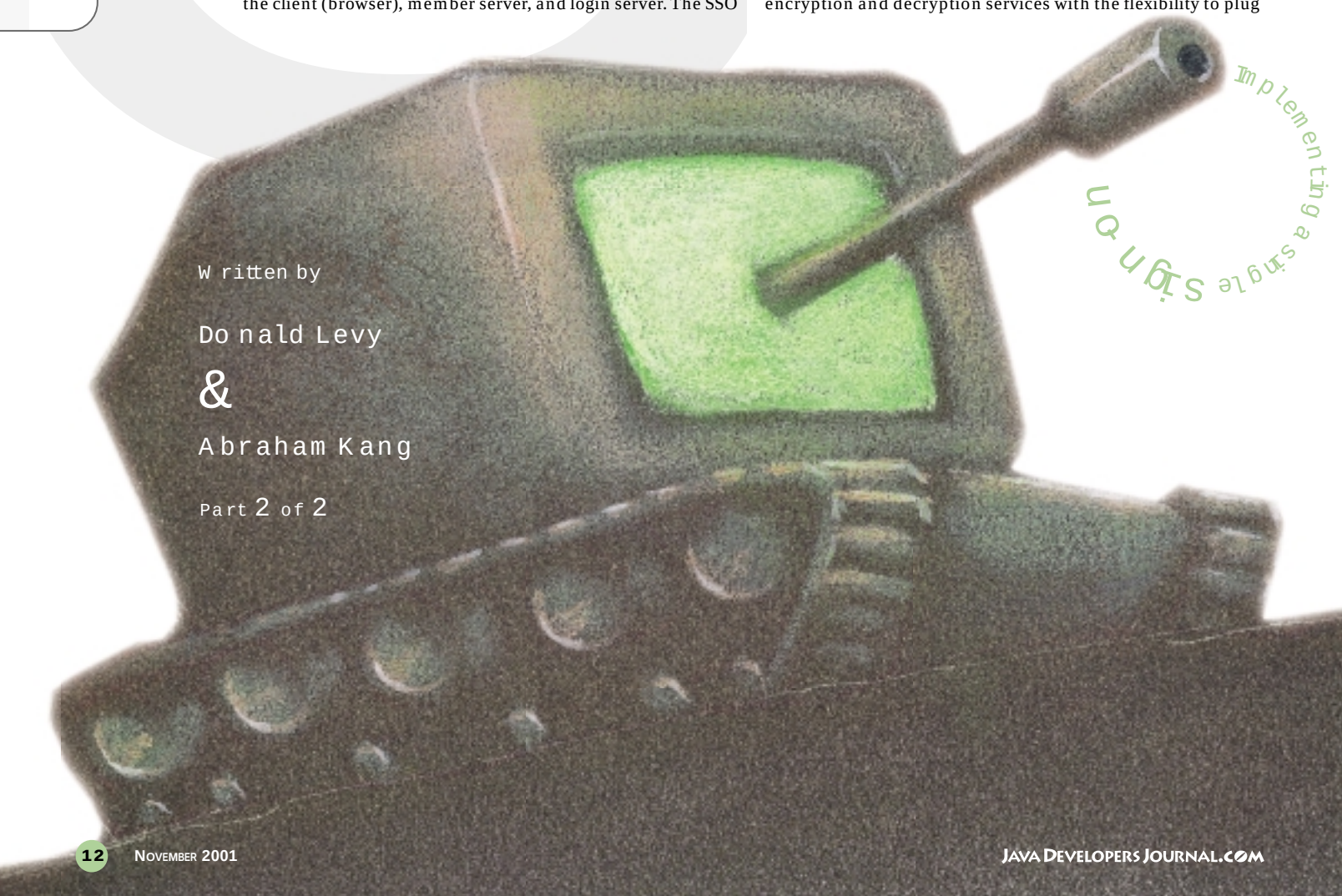
Implementing SSO for Web Applications

Single sign-on for Web applications is a protocol between the client (browser), member server, and login server. The SSO protocol establishes a set of procedures for authentication. All of the SSO procedures utilize encryption to guarantee the confidentiality of data, redirection, and JavaScript to facilitate client communication between the member servers and login server, and cookies to store SSO information. To understand the SSO protocol, we first focus on SSO's components. The first component is encryption.

Encryption

Encryption is used to encrypt URL parameter and cookie values. Sample JCE code is available on the ***JDJ*** Web site (www.sys-con.com/java/sourcec.cfm). Using JCE encryption is straightforward, but make sure you convert raw ciphertext to hex before putting it into the cookie payload or setting it as a request parameter value. The URLEncoder an URLDecoder have problems encoding and decoding ciphertext. By converting the ciphertext to hex, you don't need to URLEncode data.

To facilitate hex encoding and pluggable encryption algorithms, you'll want to create a general encryption and decryption interface, then an abstract class that implements the interface with concrete facade subclasses that utilize different encryption algorithms. This allows you to provide the encryption and decryption services with the flexibility to plug



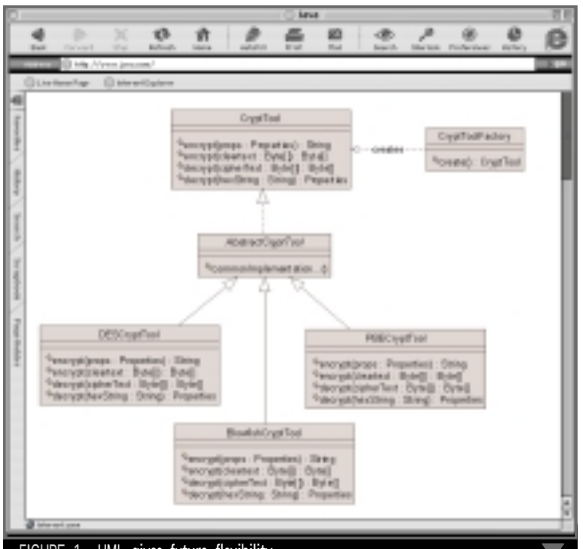


FIGURE 1 UML gives future flexibility

in new algorithms in the future (see Figure 1 for UML's advantage).

Now that we've covered encryption, let's talk about redirection.

A redirect is a type of command sent from the server that sends a browser to another location. It sends users to different servers without the user's intervention. There are two ways of redirecting users in Java. The first is through the HttpServletResponse object's sendRedirect("fullURL"). The second sends HTML and JavaScript to the client, which initiates a POST.

Here is an example of using the HttpServletResponse object:

```
public void service(HttpServletRequest req,
HttpServletResponse res)
        throws ServletException, IOException
  {

 res.sendRedirect
("https://loginserver.yourcorp.com/loginservlet?
   name=val");

…
```

This example tells the client browser to make a GET request to https://loginserver.yourcorp.com/loginservlet?name=val. Because the request is a GET, the user will be able to see the full URL, including the query string in the location bar of the browser. In addition, the amount of information that can be passed using a query string is limited and varies for different Web and app servers. Finally, it's also easier to initiate a replay attack by copying the URL string and pasting it into another browser on a different machine.

A better way of redirecting is by using HTML and JavaScript (see Listing 1).

HTML and JavaScript redirects have strong advantages when compared to response.sendRedirect( ). The first advantage is that the HTML form action is a POST. This means the query string information is not visible in the browser location bar. The second is there's no restriction on the amount of information that can be passed to the alternate server. The only drawback is JavaScript has to be enabled on the browser.

If you don't like the idea of shifting the user from one site to the next, JavaScript or hidden frames can be used to communicate with multiple member servers from a single logical page without physically redirecting the browser.

Listing 2 provides a JavaScript example where a page from the login server tells the member servers to set up a session with the browser receiving the page.

The one drawback of using <SCRIPT> tags is that processing is synchronous. The browser has to wait for partner1 to authenticate the browser before attempting partner2. Hidden frames allow you to work asynchronously, although it will only work with a frames-capable browser. See Listing 3 for an example using hidden frames.

The only frame viewable by the user is "viewable". Listing 4 shows what TempFrame.jsp would look like.

In the preceding example, frames "setPartner1cookie" and "setPartner2cookie" log the client browser into other sites asynchronously. Another advantage of using JavaScript or hidden frames is the ability to receive cookies from multiple sites from a single page.

Now that you understand redirection and its alternatives, let's look at cookies.

The Role of Cookies

Most cookies are made up of six parts: name, value, domain, path, expiration date, and secure flag. The name uniquely identifies a single cookie and is used when retrieving, deleting, and updating cookies. The value stores the payload (text) of the cookie. A cookie can hold up to 4K of information. The domain restricts which machine or DNS domains can read an individual cookie. To share cookies between "members.yourdomain.com" and "stocks.yourdomain.com", set your cookies domain to ".yourdomain.com". If you set the cookies domain to "members.yourdomain.com", then the only computer that can read the cookie is "members.yourdomain.com". The path determines which directory can read the cookie and specifies the directory in which files can manipulate a particular cookie.

If you set the path to "/", then any page on the server can read the cookie. The expiration date determines when a cookie is invalidated. The Cookie class has a method, setMaxAge (), which sets how long a cookie should last in seconds. Finally, the secure flag on a cookie ensures that the cookie can be sent only to the client or server over SSL. It doesn't encrypt the information in the cookie file on the client.

Here are several important cookie guidelines.

1. ***Set the maxAge to –1*** to create in-memory cookies.
2. ***Set the domain to ".yourDomain.com".*** This allows other machines in your domain to share cookie information directly.
3. ***Set the path to "/".*** This allows any of the JSP and servlets on a machine to obtain the cookie, regardless of their location on the machine.
4. ***Use the Properties object*** to pass multiple name value pairs within the value of the cookie or request parameter.
5. ***The secure flag should be set*** when the login server sets cookies. See Listing 5 for sample code.

To change the value of a cookie or delete a cookie, you'll get the cookie, change an attribute, and send it back using response.addCookie (Cookie changedCookie). Debugging cookies can be problematic. To minimize debugging time during development, do the following:

1. ***Set the browser*** to notify you every time a cookie is set. Go to Tools ‡ Internet Options ‡Security and select Prompt in both cookies settings. When you get the Security Alert prompt in the browser, you can see all six attributes of a cookie by clicking the "More Info" button.
2. ***Create a cookie display JSP*** to print out the cookies and their attributes.

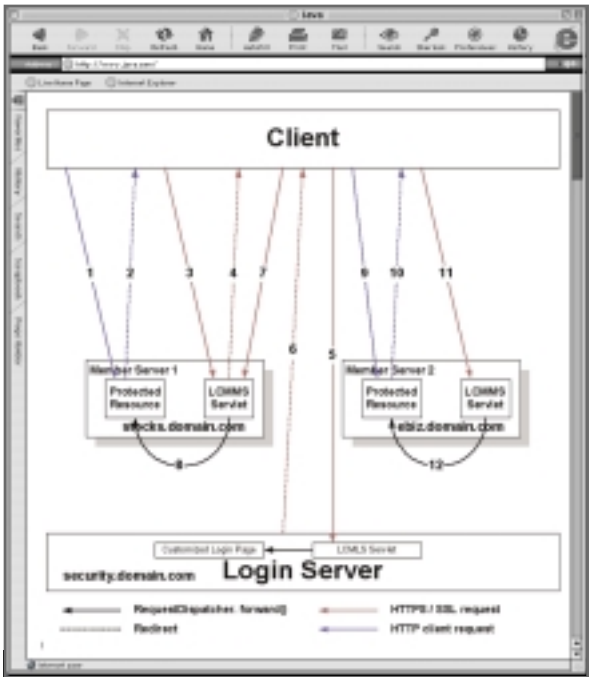


FIGURE 2 Login with shared cookies in a single DNS domain

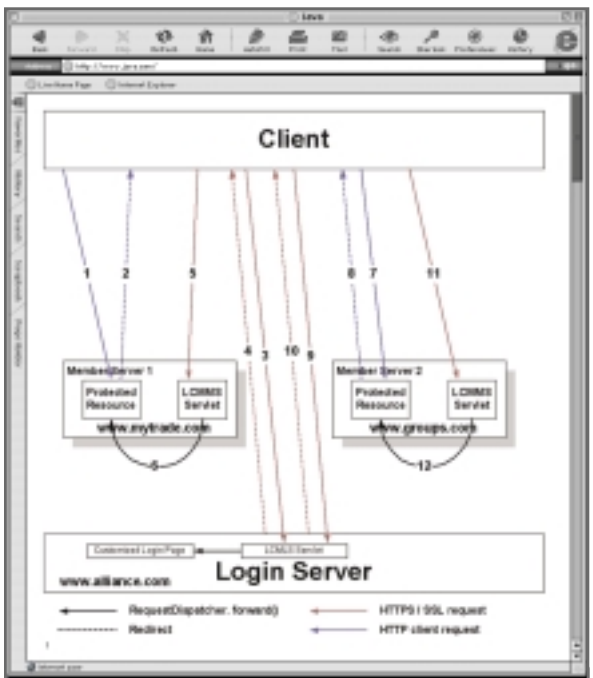


FIGURE 3 Login with unshareable cookies in different domains

3. ***Create a cookie delete JSP*** to delete all of the SSO cookies. We've covered all the components of a Web SSO solution. The last step is the SSO protocol.

SSO Protocol

The SSO protocol describes how browser clients interact with member servers and the login server to enable SSO. The protocol has many variations and you can tailor the protocol presented to suit your needs. The SSO protocol differs slightly, depending on whether all machines participating in SSO reside within the same DNS domain. When machines are under a common domain, cookies set by one machine can be read by another. The LCMServlet on the member server (MS) and login server (LS) serve as the central point of contact for authentication requests and replies. All communication to LCM servlets on MS and LS use HTTPS.

When going from member server to login server, you'll usually have to send the following:

1. ***Action:*** This tells the login server what the member server wants, e.g., authenticate, register, etc.
2. ***Return URL:*** This is the original URL to a protected resource.
3. ***Extra information for associated action:*** This might be a custom login page for the authenticate action.
4. ***Server identifier:*** This identifies which server is making the request.

When going from LS to MS, you'll send some or all of the following:

1. ***User information:*** This will typically be the user ID and identifying information.
2. ***Action:*** Authenticate, register, logout.
3. ***Result code for initiating action:*** Failed, successful, etc.
4. ***Return URL:*** The return URL represents the original URL that the user was trying to obtain.
5. ***Timestamp:*** This is added security to protect against replay attacks. Only requests that have a timestamp less than five-minutes-old will be accepted. Using timestamps requires a clock synchronization service for all servers participating in SSO.

Login Use Case

Remember there are many ways to implement a single sign-on protocol. The actual protocol you'll use depends on your requirements and environment. Figure 2 shows a login use case for servers that share the same DNS domain.

Servers with the Same Domain

1. The user hasn't authenticated and makes a request for a protected resource.
2. The protected resource notices that the user isn't authenticated and redirects the user to the LCMMSServlet passing the requested URL, the server ID, and an action of authenticate.
3. The LCMMSServlet looks for a shared authorization cookie that would have been set by the login server and doesn't find one.
4. The LCMMSServlet takes all the information passed to it and redirects the user to the login server for authentication.
5. The LCMLSServlet receives the request, notices that the user hasn't logged in, and presents a customized login page for the member server that initiated the request.
6. Once the user has successfully authenticated, the login server sets two cookies. The first cookie is an authentication cookie, that stores the user's encrypted authentication information. The second cookie stores the servers that the client has visited (for logout purposes). The login server redirects the user back to the member server's LCMMSServlet with encrypted information in a cookie.
7. The LCMMSServlet decrypts the user information and logs the user in.
8. Once authenticated, the LCMMSServlet forwards the request to a protected resource.
9. If the user tries to access a protected resource on another server in the same domain...
10. The request is redirected to the LCMMSServlet using HTTPS.
11. The LCMMSServlet checks the authenticated cookie and notices that the user has already logged in. The LCMMSServlet sets up a session and writes its server information to the "servers" shared cookie.

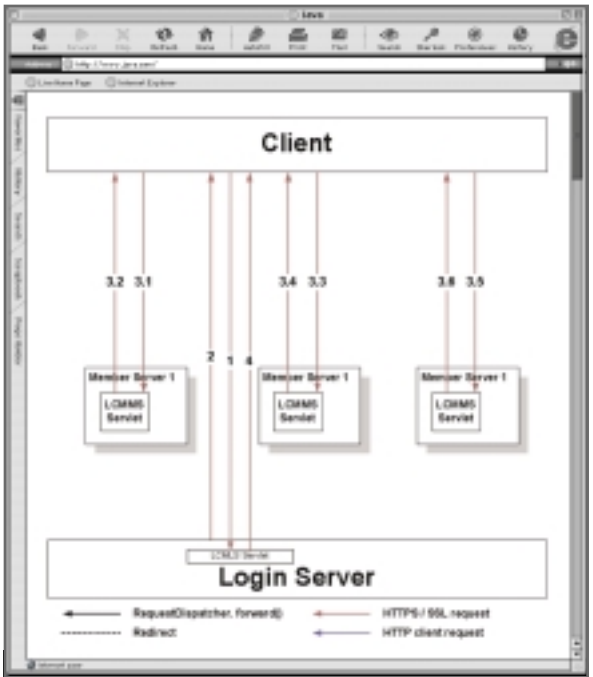


FIGURE 4 Logout using JavaScript SRC or hidden frames

12. The LCMMSServlet then forwards the request to a protected resource.

Things will be slightly different if the participating servers reside in different domains. See Figure 3 for an example.

Different Domains

1. The user has not authenticated and makes a request for a protected resource.
2. The protected resource notices that the user is not authenticated and redirects the user to the login server passing the requested URL, the server ID, and an action of authenticate.
3. The login server receives the request, notices that the user hasn't logged in, and presents a customized login page for the member server that initiated the request.
4. Once the user has successfully authenticated, the login server sets two cookies. The first cookie is an authentication cookie, which stores the user's encrypted authentication information. The second cookie stores the servers that the client has visited (for logout purposes). The login server redirects the user back to the member server's LCMMSServlet with encrypted information in a request parameter.
5. The LCMMSServlet decrypts the user information and logs the user in.
6. Once authenticated the LCMMSServlet forwards the request to a protected resource.
7. If the user tries to access a protected resource on another server in the same domain...
8. The request is redirected to the login server using HTTPS.
9. The login server checks the "authorization" cookie and notices that the user has already logged in. The login server adds the new server to the "servers" cookies.
10. Then login server redirects the user back to the member server's LCMMSServlet with encrypted information in a request parameter.
11. The LCMMSServlet decrypts the user information and logs the user in.
12. The LCMMSServlet then forwards the request to a protected resource.

Things are similar in both cases, except the protected resources now redirect to the login server instead of to their local LCMMSServlet. With shared cookies, the member server LCMMSServlets can read the authentication information directly out of the cookie set by the login server. With unshared cookies, the login server needs to tell each member server of a user's status directly (through request parameters) versus indirectly (through a shared cookie).

If all this redirection is making you dizzy, there's a trick you can use to access multiple sites from a single Web page. Let's look at how it works in the logout use case (see Figure 4).

Steps for Multiple Sites

1. The user clicks a logout icon that is linked to the LCMLSServlet of the login server.
2. The login server sends back a temporary page with <SCRIPT> tags or hidden frames.
3. Each of the <SCRIPT> tags of hidden frames contacts a different member server to logout and delete cookies all at once.
4. Once all of the member servers have been contacted, the temporary page is posted back to the login server with the member server logout responses. The login server then sends the logout response page.

The beauty of using hidden frames or a JavaScript SRC attribute is that you go to the login server only once and can access all of the member servers without redirects.

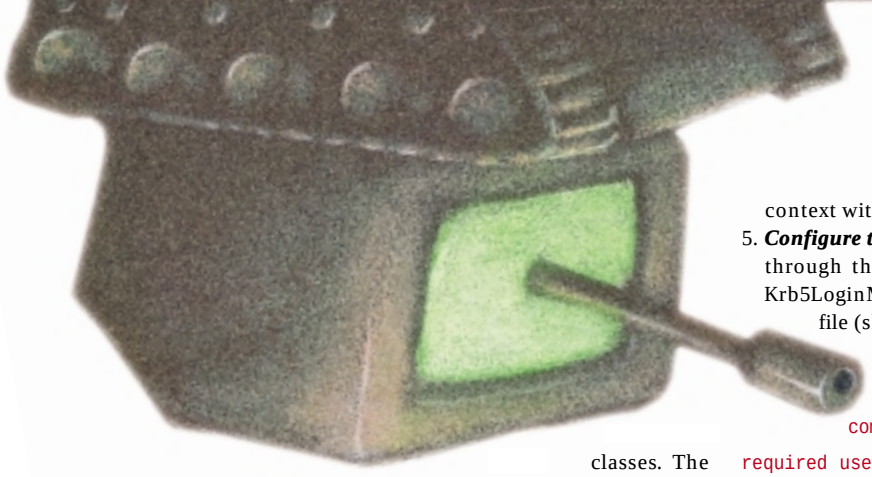
more than 45% of calls to corporate help desks involve user ID or password-related issues

You should have a solid understanding of single sign-on for Web applications. Now we'll discuss single sign-on for standalone applications.

SSO for Standalone Applications

Single sign-on for standalone applications is supported through Kerberos, JAAS, and Sun utility classes. Kerberos provides the single sign-on infrastructure and authentication service. JAAS authenticates users through Kerberos and restricts users based on permissions in a JAAS-specific policy file.

Sun JAAS utility classes (Login and MyAction) help you SSO-enable any Java application. Part 1 discussed Kerberos and JAAS in sufficient detail, so let's look at the Sun utility



classes. The Login class encapsulates all of the JAAS-specific code to load the login modules, prompt the user for credentials if necessary, and authenticate the user. The MyAction run( ) method calls the main method of the application that you're SSO-enabling.

To apply what we have learned, let's SSO-enable a Java client application that makes HTTP requests to a Web server.

How to SSO-Enable Java

Let's look at HttpSocketClient.java (see Listing 6). All the client does is open a socket to a Web server HTTP port and read the requested file.

Here are the steps to SSO-enable your Java applications:

1. ***Get Kerberos set up and running.***
2. ***Use the Login class to serve as an entry point*** for your application and to initiate the JAAS protocol. The Login class does the following:
   - Receives the main( ) class of your application as arguments and any arguments your application needs.
   - Uses the class name of your top-level application class or the –D property java.security.auth.login.config to look up your login configuration file. After finding the LoginContext config file, the Login class will look for the section that has the same name as your top-level application class.
   - Authenticates the user.
   - Associates a subject (with a principal representing the user) with the current access control context.
   - Creates an instance of the MyAction class (also in Login.java) and executes its run method. The MyAction run method uses reflection to load your application's main ( ) class and invokes its public static main method, passing it the application arguments.
3. ***Create a separate JAR*** for your application classes. In our case, we have HttpSocketClient.jar containing HttpSocketClient.java.
4. ***Configure a separate JAAS policy file*** for your application. Look at the HttpSocketClient.policy file (see Listing 7). Notice that we are using Login.jar to carry out the JAAS protocol and run our application. Login.jar contains both the Login and MyAction classes. We physically separated Sun's utility classes from our application classes so the JAAS classes could run unrestricted. The codebase assumes Login.jar and HttpSocketClient.jar are in the current directory from where you are running the program.

   The only client that can run HttpSocketClient is "your_kerb_username". The first permission allows the authenticated user's program to make a socket connection to any host. The second permission allows the client to use the ticket-granting service to get a ticket-granting ticket (TGT). The ticket-granting ticket allows clients to access kerbertized applications without logging in multiple times. The third permission allows the client to initiate a secure context with the specified server_service_principal.
5. ***Configure the LoginContext configuration file*** (as specified through the -Djava.security.auth.login.config) to point to Krb5LoginModule. Here is the LoginContext configuration file (sharedLogin.conf).

```
HttpSocketClient {
        com.sun.security.auth.module.Krb5LoginModule
required useTicketCache=true;
};
```

   The sharedLogin.conf is used to specify the different authentication mechanisms JAAS will use. Here we use the Krb5LoginModule. The Krb5LoginModule interfaces with the Kerberos software client. The "required" flag indicates that this module has to commit successfully for the user to be authenticated. The useTicketCache option tells the LoginModule to look for the ticket-granting ticket in the Kerberos client cache.

   If the TGT doesn't exist, JAAS will prompt users for their user IDs and passwords and log them into Kerberos (which generates a TGT). If the TGT exists, the Krb5LoginModule retrieves the principal associated with the TGT. JAAS then checks the Kerberos principal with the principal specified in the JAAS policy file (HttpSocketClient.policy). If the principals match, the application is run using the permissions in the JAAS policy file (HttpSocketClient.policy). If they don't match, the user isn't allowed to run the JAAS-protected application.
6. ***Run your application*** through the Login class. To start the application execute the following command on a single line:

```
java -classpath Login.jar;HttpSocketClient.jar
 -Djava.security.manager
 -Djava.security.krb5.realm=<your_realm>
 -Djava.security.krb5.kdc=<your_kdc>
 -Djava.security.policy=HttpSocketClient.policy
 -Djava.security.auth.login.config=sharedLogin.conf
 Login HttpSocketClient <host> <port_number> <path>
```

If you haven't logged into Kerberos with kinit, you will be prompted for your Kerberos credentials. The Login utility class will authenticate you against Kerberos and set up your Kerberos client with a ticket-granting ticket. If the client logged into Kerberos using kinit or Leash32, the Krb5LoginModule will look for his or her TGT in the user cache. The TGT will live on your machine for a default of 600 minutes, so remember to delete your TGT using kdestroy or Leash32 before logging out of your computer.

You now have an understanding of single sign-on. You've also seen the issues related to single sign-on – as well as possible solutions. With this practical knowledge in hand, you can now build your own single sign-on solution.

Author Bios

Abraham Kang is an integration architect for Infogain's Enterprise Integration Solutions group. His focus is J2EE, security, and enterprise application integration. Abraham has been developing in Java for five years.

Donald Levy is the director of alumni relations and development information systems at Stanford University. Donald has a BS from MIT and an MS from Columbia University.

abraham_kang@yahoo.com & donald.levy@stanford.edu

Security for J2EE Applications

Listing 1

```
<html>
<head>
</head>
<body onLoad="document.myForm.submit()">
<form action="https://loginserver.yourcorp.com/webapp/login
servlet" name="myForm"
      method="POST">
<input type="hidden" name="key" value="!@#$EncryptedString!@#$">
</form>
</body>
</html>
```

Listing 2

```
<html>
<head>
<title>Hello World</title>
<SCRIPT language="JavaScript"
SRC="https://partner1/servlet/LCMMSServlet/login?data=ALKSDFJQWER…J
LQKWE">
</SCRIPT>
<SCRIPT language="JavaScript"
SRC="https://partner2/servlet/LCMMSServlet/login?data=ALKSDFJQWER…J
LQKWE">
</SCRIPT>
<SCRIPT language="JavaScript">
function postForm() {
   document.myForm.submit( );
}
</SCRIPT>
</head>

<body bgcolor=#FFFFFF onLoad="postForm()">
<form action="https://myserver/servlet/LCMMSServlet/authenticated"
 method="POST" name="myForm">
<input type="hidden" name="data" value="ALKSDFJQWER…JLQKWE">
<input type="hidden" name="url"
value="/requested/url?param1=val1&param2=val2">
</form>
</body>
</html>
```

Listing 3

```
<FRAMESET ROWS="100%,0%,0%" onLoad="submitViewableFrameForm()">
   <FRAME NAME="viewable" SRC="TempFrame.jsp">
   <!--The frame below logs in the browser to partner1 -->
   <FRAME NAME="setPartner1cookie"

SRC="https://partner1:7002/servlet/LCMMSServlet/login?data=ALKSD-
FJQWER…JLQKWE">
   <!--The frame below logs in the browser to partner2 -->
   <FRAME NAME="setPartner2cookie"

SRC="https://partner2:7002/servlet/LCMMSServlet/login?data=ALKSD-
FJQWER…JLQKWE">
</FRAMESET>
```

Listing 4

```
<html>
<head>
</head>
<body bgcolor=#FFFFFF>
Put some text here like "Logging in…Please wait."
<form action="https://myserver/servlet/LCMMSServlet/authenticated"
 method="POST" name="myForm" target="_top">
<input type="hidden" name="data" value="ALKSDFJQWER…JLQKWE">
<input type="hidden" name="url"
value="/requested/url?param1=val1&param2=val2">
</form>
</body>
</html>
```

Listing 5

```
CryptTool ct = CryptToolFactory.getCryptTool( … );
Properties p = new Properties ( );
… //Get user Id
String userId = …;
p.setProperty ("uid",userId);
p.setProperty("anotherProp", someValue);

//The String returned is a hex encoded ciphertext
String encryptedInfo = ct.encrypt(p);
Cookie c = new Cookie ("SSO",encryptedInfo);
c.setMaxAge(-1);
c.setDomain(".yourDomain.com");
c.setPath("/");

//If this is a login server cookie and cookie has to be sent over
SSL
c.setSecure(true);

//Send cookie to client
response.addCookie (c );
```

Listing 6

```
package jdj.sso.test;

import java.net.*;
import java.io.*;

public class HttPSocketClient {

  public static void main(String[] args) throws Exception {
 String host = null;
 int port = -1;
 String path = null;
 for (int i = 0; i < args.length; i++)
     System.out.println(args[i]);

 if (args.length < 3) {
     System.out.println(
  "USAGE: java HttPSocketClient " +
  "host port requestedfilepath");
     System.exit(-1);
 }
 try {
     host = args[0];
     port = Integer.parseInt(args[1]);
     path = args[2];
 } catch (IllegalArgumentException e) {
      System.out.println("USAGE: java HttPSocketClient " +
   "host port requestedfilepath");
      System.exit(-1);
 }

 try {


     Socket socket = new Socket(host,port);

     PrintWriter out = new PrintWriter(
      new BufferedWriter(
      new OutputStreamWriter(
                              socket.getOutputStream())));

     out.println("GET " + path + " HTTP/1.1");
     out.println();
     out.flush();

     BufferedReader in = new BufferedReader(
        new InputStreamReader(
        socket.getInputStream()));

     String inputLine;

     while ((inputLine = in.readLine()) != null)
  System.out.println(inputLine);

     in.close();
     out.close();
     socket.close();

 } catch (Exception e) {
     e.printStackTrace();
 }
  }
}
```

Listing 7

```
grant CodeBase "file:./Login.jar" {
        permission java.security.AllPermission;
};

grant CodeBase "file:./HttpSocketClient.jar",
 Principal javax.security.auth.kerberos.KerberosPrincipal
  "your_kerb_username@your_realm" {

 permission java.net.SocketPermission "*", "connect";

 permission javax.security.auth.kerberos.ServicePermission
  "krbtgt/your_realm@your_realm",
  "initiate";

 permission javax.security.auth.kerberos.ServicePermission

"server_service_principal@your_realm",
  "initiate";
};
```

Centralized Entitlement Engine Blueprints

How to deal with the reality that all users aren't equal

WRITTEN BY
GIORA KATZ-LICHTENSTEIN

Entitlement engines are infrastructure component-enabling software applications that restrict access to data, based on users' privileges. They operate by comparing privileges granted to users (keys) with access permissions that protect resources (locks).

The need for entitlement engines originates from the harsh reality that users are created unequal and different. Access to data is granted based on the business relationship between the data provider and supplier. For example, a Citibank customer should be granted access to data relevant to his or her own account and service level agreement – and only to that data.

Historically, client/server systems relied on RDBMS's built-in security mechanism for privilege checks. To do so, each user had a database logon. The transition to Web-based three-tier systems mandated an entitlement check mechanism that was independent of RDBMS's privilege checks and capable of handling Web-based access. Furthermore, portals and search engines that aggregate data from multiple enterprise legacy systems mandate a centralized control of entitlement checks and security.

This article provides blueprints for building a centralized entitlement engine for an enterprise. It addresses the major design and methodology issues in building such an entitlement engine and provides some real-world experience. The article is geared to architects, developers, and security professionals in need of managing access to multiple enterprise systems.

Additional information about building a centralized entitlement engine is available at http://home.nyc.rr.com/giora/, and includes the following:

- Project plan
- Integration with other systems
- Management API

Evolution of Entitlement Engines

The evolution of entitlement check engines can be roughly broken into the following stages:

- ***Client/server systems*** relied on RDBMS's built-in features to perform entitlement checks. (This mechanism was also used by PERL-CGI Web-based systems.)
- ***Early Web-based systems*** relied on a custom-built entitlement check API servicing a single application.
- ***Newly built enterprise applications*** rely on enterprise entitlement engines capable of centralizing the management of entitlement company-wide.

The transition from RDBMS-based entitlement checks to custom-built entitlement engines was driven by the stateless nature of Web clients using the HTTP protocol. It made little sense to perform a database logon for each HTTP request. Doing so would have been slow and resource-intensive.

In the same way, portals and search engines aggregating data from multiple applications across an enterprise drive the transition from custom-built entitlement engines to a centralized entitlement engine.

A List of Requirements

The following is a list of mandatory requirements for an enterprise entitlement engine:

- One centralized point of management and storage for entitlement data – to make a long story short, this is the mother of all requirements. It mandates a single database containing all entitlement information. By centralizing the data and its management, it makes the life of security administrators manageable.
- Users must be uniquely identified company-wide.
- All enterprise systems supported by the entitlement engine must comply with this previously mentioned user identification mechanism.
- Support integration with multiple systems in multiple languages and environments – a language-independent API.
- Provide high performance.
- Allow dynamic changes of entitlement policies.
- Allow management of entitlement, based on assignment of users into groups. And allow management of entitlement based on assignment of groups into groups.
- Support assignment of unique identifiers to enterprise resources in need of entitlement checks (hyperlinks and APIs).
- Capable of entitling hierarchies of protected resources.
- Hierarchical management model must support the delegation of management tasks to part of the protected resources hierarchy.
- Intuitive manageability via usage of a GUI Console

Build vs Buy

The project plan (found at http://home.nyc.rr.com/giora) demonstrates that building an enterprise entitlement engine can last between three to six months. In an ideal world, an entitlement engine is something we buy from a software vendor (less code for us to write and maintain). I suggest you find a vendor that satisfies the conditions mentioned in the preceding requirement list. Before you write a check, however, read the following warning: *Beware of slow entitlement engines!*

Yes, performance is the number-one requirement. We can't cut any slack when it comes to performance. A delay in response time due to slow entitlement checks is likely to have a negative

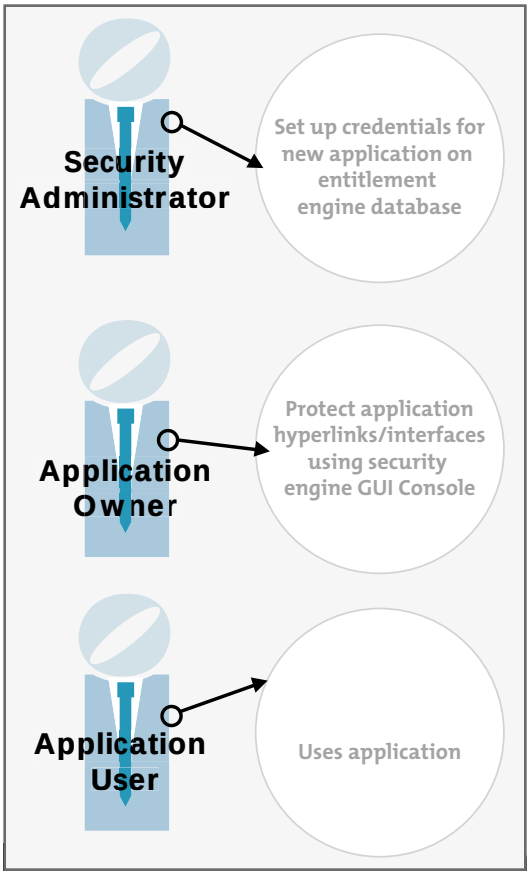


FIGURE 1 Use cases of entitlement engine

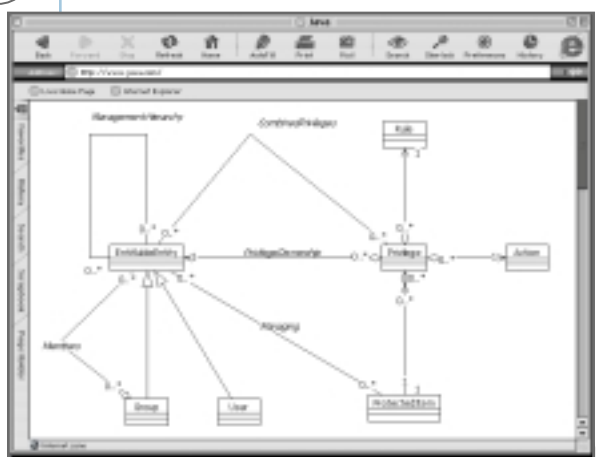


FIGURE 2 Enterprise entitlement engine object model

FIGURE 3 Tree representation

influence on the career of the engine owner.

A common characteristic of the high-performance engines is that they all do all of the runtime entitlement checks in memory. I think an entitlement engine that needs to query databases, move entity beans between pools, or any other type of roundabout configuring in runtime is doomed!

Now you know what to look for when you shop for an entitlement engine. If you find a good one that suits your needs, more power to you. If you don't, read the following for a blueprint of a custom-built entitlement engine.

Use Case: The Actors

- An application owner interested in protecting the hyperlink to his or her applications
- A security administrator in charge of the entitlement engine
- An application user interested in using the application (see Figure 1)

Use Cases by Chronological Order

1. The first use case to take place happens when the owners of an application contact the entitlement engine administrator to notify him or her that the application needs entitlement protection. To get started the application owners must provide the following information about the application:
   - ***List of users or groups*** that will be granted management right for this application
   - ***Name of the application*** in conformance with the corporation naming convention – this name will be used later to fit the application within the ProtectedItem tree (covered later)
   - ***Type of protection needed,*** whether external or internal (external protection entitles access to a URL. Internal protection entitles access to a block of code of an application)

   The security administrator provides the application owner with the necessary rights to manage the application using the entitlement engine management console.
2. The second use case to take place is driven by an application owner who does the following:
   - ***Protects hyperlinks*** via external entitlement check.
   - ***Protects APIs*** via internal entitlement check.
   - ***Grants privileges*** to users and groups to use the hyperlinks and APIs. Privilege/entitlement management is done via the entitlement engine console.
3. The last use case is the actual entitlement/privilege check. A user of the previous protected application attempts to access the application. The call is delegated to the entitlement engine. The entitlement engine performs the entitlement check. If the check result is a success, access is allowed to the protected application (see Figures 4–8).

Class Diagram

The enterprise entitlement engine object model (see Figure 2) consists of the following:

- ***EntitleableEntity:*** Represents an entity capable of granting entitlement and managing entitlements or benefitting from them.
- ***User:*** The user object represents a real-life client/user (actor in UML terms) of an application protected by the entitlement engine. Users have one unique user identifier: GUID.
- ***Group:*** A collection of users or groups; groups can be granted rights to access applications or manage entitlements via their relationship to their super class (EntitleableEntity).
- ***ProtectedItem:*** Represents a hyperlink or an API of an application. The ProtectedItem is organized into a hierarchical tree (covered later).
- ***Rule:*** Represents a rule associated with the privilege. The Rule class holds a user-defined rule written in some rule language. (Rules are optional.)
- ***Action:*** Represents an operation to be performed on the protected item, such as view, execute, modify, delete, manage, deny view, deny execute, or deny manage.
- ***Privilege:*** A single association between protected resource, user or group, action, and rule, it determines the usage rights of users on protected applications.

Relationships

- *PrivilegeOwnership* represents the relationship between the Privilege and the exact EntitleableEntity (User or Group) to which it was granted.
- *CombinedPrivileges* represents the cross influence between a privilege and all of its beneficiaries – users who got the privilege directly or indirectly via group membership. There's a good reason to build this relationship into the model. Privileges do change during the runtime of the entitlement engine. When they change, the ripple effect on each user must be figured out. This relationship makes it possible.
- The *managing* relationship is a sticky issue. Remember, the burden of entitlement management between multiple administrators should be distributed.

  –We need a separate hierarchy to determine who can manage what.

  –All management operations must be entitled.
- The *members* relationship between a group and EntitleableEntity represents a group membership. Notice the potential for recursiveness, where a group can contain another group. Also notice that circular relationships between groups are legal.

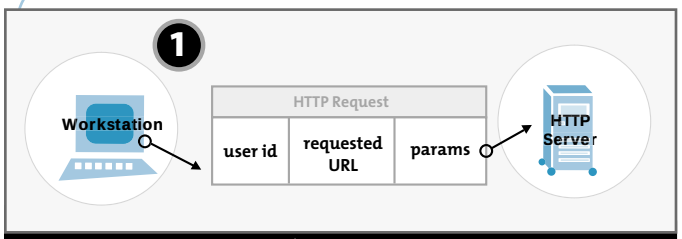


FIGURE 4 Request to access a ProtectedItem

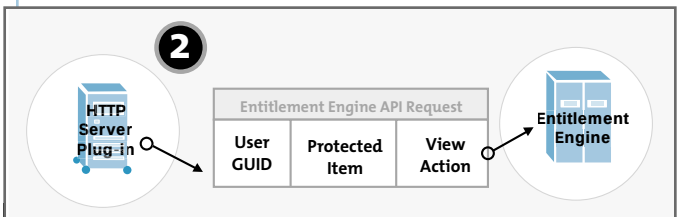


FIGURE 5 Plug-in forwards the request

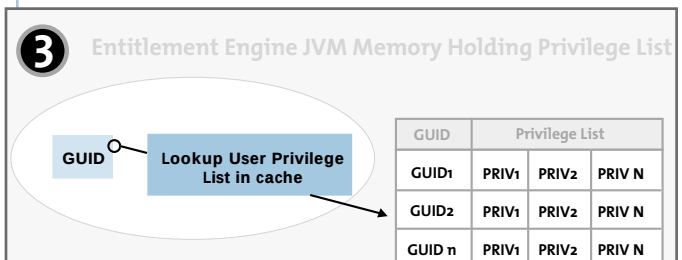


FIGURE 6 ID in user's privilege's cache

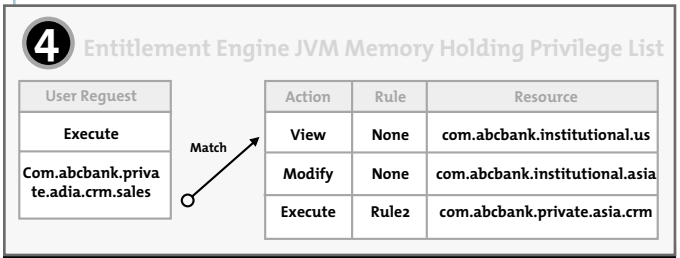


FIGURE 7 Search of privilege list

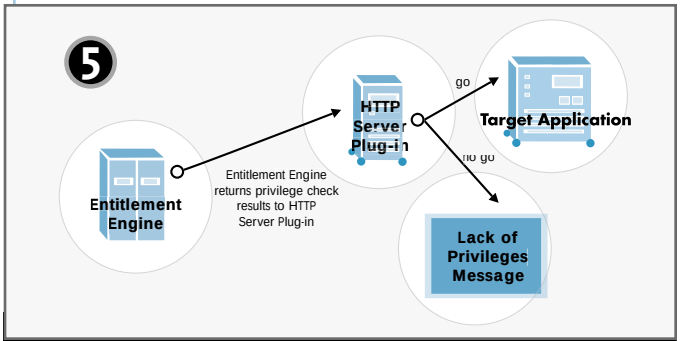


FIGURE 8 Entitlement engine finds privilege

Missing Items in the Object Model

What about supporting a use case where access is revoked for all users of a given company? Our object model doesn't cover that. Instead the combined usage of GUID and a cross-reference table enables integration between an entitlement engine and any CRM system.

ProtectedItem Tree

I'll use a real-life example to demonstrate a fundamental part of our object model: the ProtectedItem tree. The example is entitlement management for a multinational bank operating in the U.S., Europe, and Asia. The bank has the following departments: institutional banking, private banking, and investment management. Each department has several systems. To distribute the burden of security management, each department employs its own security administrator. The centralized entitlement engine owner delegates management rights to the regional security administrators. The delegation is done by assigning regional security administrators with full control over a portion of the entire ProtectedItem tree.

The security administrator employed by the Asian institutional department has full control over the tree branch *com.abcbank.Institutional.Asia*. He's entitled to manage any tree branch below that point (see Figure 3).

In a similar manner a user granted access to com.abcbank.Institutional.Asia is implicitly granted access to *com.abcbank.Institutional.Asia.MoneyMarkets*, *com.abcbank.Institutional.Asia.Swift*, and *com.abcbank.Institutional.Asia.WebService*.

Note: The dot separator is used to break down the names into a hierarchy.

The Runtime API

The following API contains all the necessary signatures for privilege/entitlement checks. The mapping of this API to various languages will be discussed later.

```
public boolean
checkEntitlement(User user,
ProtectedItem item, Action action,
Rule, rule);
public boolean[]
checkEntitlements(User user,
ProtectedItem items[], Action
actions[]Rule rules[]);
```

The runtime operation of the entitlement engine is demonstrated by Figures 4–8.

1. A user clicks on a URL associated with a ProtectedItem in the ProtectedItem tree (see Figure 4).
2. An HTTP server passes a request to a custom-built plug-in designed to package requests for the entitlement engine API. The plug-in forwards the request to the entitlement engine API for an entitlement check (see Figure 5).
3. The entitlement engine looks up the user ID in its user privileges' cache and fetches the collection of privileges for that user (see Figure 6).
4. The privilege list is searched for all privileges referencing the ProtectedItem hierarchy with the relevant action (see Figure 7).
5. In case such privilege is found and access is not denied, the entitlement engine returns success; otherwise, it defaults to a failure (see Figure 8).

Note: We're making an assumption that by the time the HTTP server entitlement engine plug-in receives the HTTP request, it contains the GUID of the user. Implicitly it's assumed the user had already been authenticated and the GUID was fetched from another system.

The Game Plan

Implementing an entitlement engine can be confusing for the newcomer. Have no fear: soon you will understand both the forest and the trees. The entitlement engine code can be broken into the following pieces:

1. Persistent datastore holding configuration information
2. Bootstrap logic loading the users' privilege mapping into memory
3. Runtime logic doing the entitlement check and the multilanguage API support
4. Management API-enabling programmatic update of the entitlement data in the persistent store

The following walks you through the architectural issues encountered when building a high-performance entitlement engine.

Persistent Datastore

The persistent datastore is a fancy name for a relational database that's needed to save the entitlement information. It doesn't really matter which kind of RDBMS, since the runtime performance doesn't depend on it (it's all in the memory).

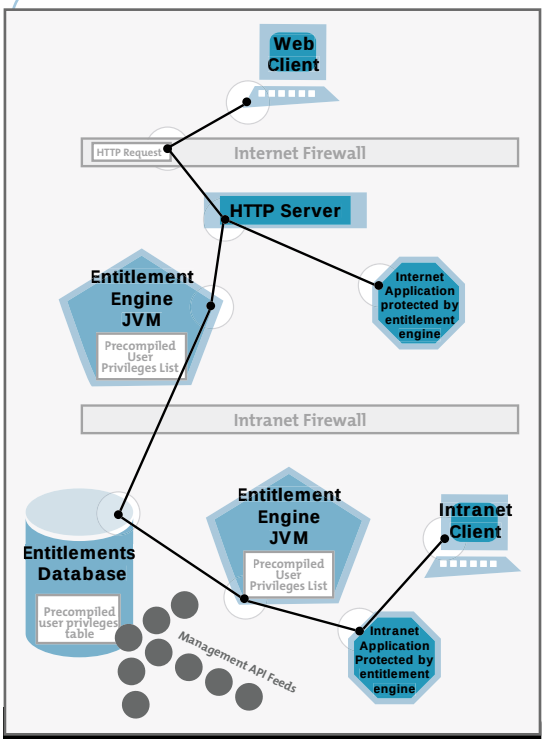


FIGURE 9 Entitlement engine Internet and intranet deployment

Here are some pointers that can really make interaction with the persistent datastore easier:

- Using a database that supports Java triggers can be helpful. We can use those triggers to spawn an update to all the replicas of the user privileges table residing in multiple entitlement engines' JVMs. This is where the Observer design pattern comes in handy for a simple pub/sub interaction between the persistent datastore to multiple entitlement engines (see Figure 9).
- Our object model is not complicated, but mapping it to a relational database isn't trivial. Manually coding the persistence layer is labor-intensive and error-prone. This is where a good object-relational mapping tool is useful. Any object-relational mapping tool eliminating hand-coded JDBC calls will do.
- Our object model contains an OID for every class (OID stands for Object Identifier). An OID is a persistent immutable object identifier. The OID representation on the database is a column in each table. The OID representation in a programming language is a member variable with the following characteristics:
  1. It shouldn't have any business meaning.
  2. It should be a numeric data type.
  3. It should be immutable; once allocated it never changes.
  4. It should be allocated automatically via a mechanism that guarantees uniqueness.

Note: The Interface Police pulled me over several times before for not writing enough abstraction layers in the form of stored procedures. They won't rest before forcing the entire developer community to write thousand of lines in redundant code – adding little or no value from a business perspective. In my opinion there is exactly one place where a database stored procedure might be useful for an entitlement engine. That place is figuring out the content of the user privileges table (covered later). For all the rest, dynamic SQL will do just fine.

Bootstrap Mechanism

Owners of applications protected by the entitlement engine literally expect entitlement checks to be performed in no time. The performance requirements are divided into three distinct steps:

1. Our persistent datastore will always keep an up-to-date precompiled mapping between the users and their privileges.
2. The user-to-privilege mapping is cached in the entitlement engine JVM (java.util.Map.interface). The cache key is the user and the cache value is the list of privileges affecting this user.
3. The cached user to privileges mapping is kept current via a pub/sub mechanism.

The in-memory mapping between users and their privileges practically guarantees a high runtime performance since there isn't much to be done during the entitlement check. We narrowed it down into two simple steps:

1. A single Map lookup returns the collection of all privileges granted to a user.
2. A comparison of the user privileges with the requested action at hand boils down to a single pass of a loop.

From a performance standpoint, this is as good as it gets. Here is an explanation of the overhead omitted by using the precompiled user privileges list.

A user can obtain privileges from three sources:

- By a privilege granted directly to the user
- By a privilege granted to a group the user belongs to
- By a privilege granted to a group containing a group the user belongs to

The precompiled list of user-privileges mapping allows us to bypass the lengthy process of figuring out all the privileges owned by a user.

Keeping Replicas Current

The important thing to keep in mind is that the in-memory, user-privileges list must be synchronized with the persistent precompiled table of user privileges for all the instances of the entitlement engine. It's likely that an enterprise will simultaneously run multiple entitlement engines per one persistent datastore (as described in Figure 9).

- The synchronization interval should be configurable via a property/configuration text file.
- The triggering of the synchronization process should originate from the persistent store. It happens after a management API modification of the entitlement information. A database trigger might be useful for spawning this process.
- In an ideal world the in-memory user privileges' list would be kept current without any disruption to runtime performance. One way to get close to this ideal is by keeping two instances of the user-privileges HashMap cache when only one of the two is active. Doing so enables us to update one copy in the background. (This is somewhat similar to the Copy-on-Write design pattern introduced in *Concurrent Programming in Java*, by Doug Lea.) This approach sacrifices some memory in the form of an additional Map instance.

To Use XML or Not?

The requirements list explicitly mandates support for multiple languages. There are several roads we can take for multilanguage support:

- Using XML
- Using byte stream protocol
- Writing wrappers to the Java API to various languages (C, C++, Perl, etc.)

Another key requirement mentioned in the requirements list is performance. I found that building an XML-DOM tree for each request is significantly slower in comparison to a Java or stream protocol.

Possibly XML will become a valid option due to an increase in processing power, improvements in XML parser technology, or usage of the SAX protocol. For the purposes of this example, however, I don't believe XML is an option for today's high-performance entitlement engines.

A Byte Stream Protocol Alternative

An alternative approach is defining a byte stream protocol using predefined delimiters. A StringTokenizer can construct the ProtectedItem, users/groups, and actions based on the delimiter location within the incoming byte stream. The power of this approach comes from delivering a single API supporting multiple languages with a high performance.

Notice that by using a byte steam protocol more work is delegated to the user community of the entitlement engine.

We could write a wrapper to the Java API to each and every language used within our enterprise , but it inflates our code base.

Privilege Check Algorithm

Before diving in, let's recap privileges. A sample privilege can be verbalized in the following manner: allow user John Smith to perform the action view on the ProtectedItem *com.abcbank.investment*.

Notice several important points:

- The default policy is to return false from a privilege check. Unless a privilege permitting an action is explicitly specified, access is denied.
- Privileges are applied in a downward fashion throughout the ProtectedItem tree. If a privilege is applied to the ProtectedItem*com.abcbank.investment*, then it also applies to the ProtectedItem *com.abcbank.investment.research*.
- After a privilege is applied to some ProtectedItem, the only way to block it from the hierarchy is by another privilege – denying the action in the original privilege.

Pseudo Code

Our inbound goods are the following:

1. User privileges list
2. Requested ProtectedItem
3. Requested action

This is the method body:

- Create a variable holding the most relevant privilege.
- Loop all the privileges in the user-privileges list.
- Check if the privilege action matches the requested action or its denial.
- Check if the privilege ProtectedItem is located above or at the location of the requested ProtectedItem.
- If both checks above are positive, the privilege is relevant. Compare it with the variable holding the most relevant privilege. The one closer to the requested ProtectedItem in the ProtectedItem tree takes precedence. The privilege with precedence is now in the temporary variable holding the most relevant privilege.
- Exit the loop.
- Evaluate the value of the variable holding the most relevant privilege. If this privilege allows access, return true; for anything else, return false.

Acknowledgment

Two people contributed a vast amount of information about this topic: Victor Podvalny and Danny Padwa from Goldman Sachs. Victor and Danny pioneered the technology of enterprise entitlement engines. The entitlement engines they designed and coded deliver a world-class performance and functionality.

Resources

1. Ambler, S.W. (2001). Mapping Objects to Relational Databases. www.ambysoft.com/mapping Objects.html.
2. Brown, W., Malveau, R.C., McCormick, H.W., and Mowbray, T.J. (1998). *AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis*. Wiley.
3. Lea, D. (1999). *Concurrent Programming in Java*. Addison-Wesley.

AUTHOR BIO

Giora Katz-Lichtenstein is an independent software consultant. specializing in Object Relational Mapping, OO A&D,middleware, application servers, and the economy of software projects. He is currently constructing a commercial high performance enterprise entitlement engine.

gkatzlic@yahoo.com

The Death of J2SE? Not Quite

WRITTEN BY ERIC SHAPIRO

Much of Sun's emphasis lately has been on server-side and J2ME technologies, and many industry pundits have predicted death knells for client-side Java.

This makes some sense. Let's look at some J2SE facts:

- The initial versions of Java Virtual Machines (JVMs) were bug-laden, slow, and unreliable, making it nearly impossible for the nascent Java pioneers who had invested heavily in this technology to deliver a reliable, high-quality end-user experience. Developers shied away from widespread adoption, despite rapid improvements to client-side Java.
- Microsoft decided to exclude their version of Java from Windows XP, making client-side Java even more difficult to deploy, and posing new headaches for client-side developers; for example, how will end users run my application if they first have to install Java?
- Web services like SunONE and .NET requires abstraction layers and universal protocols such as SOAP and XML, and pushes most processing and logic to the server-side. This undermines the need for any client-side technology platform such as Java, and provides the option to have all client-side applications delivered through a Web browser, if desired.

So, is anyone still writing Java client software? Will Java ever succeed as a client-side platform?

The answer to both of these questions is "Yes," in increasing numbers, and that there is a strong future for client-side Java.

Here's Why

As the Web services model continues to be adopted throughout the industry, developers have explored the idea of using a Web browser to deliver the client experience. While this does have its place, more and more developers are realizing that it's impossible to deliver a truly customized and application-specific experience with the limitations imposed by Web browsers and HTML. We are, in fact, seeing that more powerful, rich client applications are becoming even more popular, leveraging the power and flexibility of client-side Java.

Furthermore, Java clients work better with Java server applications than do their HTML-based counterparts. J2SE's rich API makes communication with J2EE servers simple, enabling client-side developers to customize the end-user experience with application-specific interfaces. Because of this, as Java continues to succeed on the server-side, developers will continue to write client-side Java applications, leveraging the skills they've already developed without having to hire new HTML design and engineering teams.

We recently conducted an informal poll of Java tools companies and confirmed that, despite rumors to the contrary, client-side Java continues to be a dominant and viable solution for accessing J2EE-based server applications. Developers have experimented with, investigated, and invested in HTML-based client applications – only to find that a Web browser provides a poor user interface and customer experience, and requires the dumbing down of the extensive logic built into their server applications. To add insult to injury, HTML-based applications often require even more resources and time than Java-based client applications, with new design teams and new HTML developers needed to create the same code that existing Java engineers could have created.

The ultimate success of client-side Java is dependent upon solving the thorny issues of deployment, download, and execution. Right now, the average end user has a hard time understanding how to install, configure, and use a Java application. The additional overhead and requirements of installing the JVM discourage the adoption of J2SE applications because of the prohibitive size and complexity of configuration. With Microsoft choosing to exclude their version of the JVM from Windows XP, the success of Java on the client-side will be determined by Sun's ability to successfully market and deliver Java to the platforms that end users use the most.

Sun has addressed this with the introduction of the Java Plug-in and Java Web Start, two technologies that enable the easy installation and execution of Java-based rich client applications. These, along with commercially available Java-based installation solutions, make it possible to deploy Java client applications with little or no end-user configuration. The application downloads and installs, along with a working and tested JVM, and voilà, you're using a client-side Java application.

No, J2SE isn't dead; it's alive and well, and continues to be a cornerstone strategy in the SunONE vision and in successful Java-based application deployment throughout the industry.

eric.shapiro@zerog.com

AUTHOR BIO

Eric Shapiro is CEO and cofounder of Zero G Software, Inc (www.zerog.com), makers of InstallAnywhere and PowerUpdate..

J2SE INDEX

Rich User Interfaces

We're putting our money on GUIs not HTML

Most Java developers intuitively understand the advantages of using a rich Java Swing user interface instead of an HTML interface. The fact that rich user interfaces provide a better experience for the user has often been cited as the primary reason they should be employed in a particular application.

In this article, however, we provide four factors that support the claim that rich user interfaces may be the better choice for many applications – and not just because they provide a better user experience.

We begin by illustrating that rich GUIs can be developed and maintained at a lower cost than HTML-based user interfaces. We then argue that since Web services are typically highly interactive in nature as opposed to being a data delivery system, rich GUIs may be more appropriate than HTML interfaces for Web services. And finally, we describe a new architecture, which we refer to as the *Application Canvas*, for development of Web services–based applications that allow for integration of autonomous Web services on the client. As the application canvas is user-programmable, the user is able to link outputs of one Web service to the inputs of another, thereby, allowing the user to derive functionality not foreseen by systems developers.

Pros and Cons of Rich User Interfaces

We define rich user interfaces to mean Java Swing–based user interfaces. These interfaces improve user experience, but the advantages most overlooked are lower development and maintenance costs – and a high potential for reuse of interface components.

Some believe that rich GUIs are expensive to build. We argue that rich user interfaces can be built at the same cost, if not less expensively, than HTML-based user interfaces. Additionally, until recently, a case could be made that another disadvantage to rich GUIs was deployment. We'll address this issue by detailing the significant progress made in Java deployment technologies to drive deployment costs to a level comparable to the cost of deployment of HTML user interfaces.

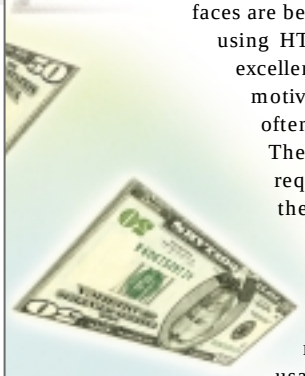
The Case for Rich User Interfaces

Rich user interfaces are composed of high-level user interface components, such as tabbed panes, table widgets, and tree widgets. Compare an HTML tabbed pane with a tabbed pane in a rich user interface.

A tabbed pane is a single component in the Swing component set, and the properties of this component are standardized and well documented. In contrast, an HTML interface can display a tabbed pane; however, the designer must put a tabbed pane together from lower level graphics, HTML, and possibly JavaScript components. The construction of the HTML tabbed pane is likely to be nonstandard and may entail the development of a significant amount of scripting code. Therefore, developing a user interface that requires a tabbed pane in Java Swing will require less code to be written than a user interface with the same functionality developed in HTML. That's why sophisticated HTML user interfaces, especially when combined with JavaScript, may prove to be more expensive to build and maintain than rich user interfaces composed of high-level user interface components.

Maintenance and cost of development of software systems, as well as user interfaces, are often measured by the degree of reuse achieved during development. Reuse is extremely difficult to achieve when the interface is built from low-level HTML and JavaScript components. But when the interface is built from high-level Java Swing components, reuse is much easier to achieve. Composite widgets can be constructed from Swing JavaBeans and later reused throughout the user interface of the application.

Another advantage to using high-level user interfaces is that the skills required to develop an HTML user interface



and the skills required to develop a Java application are often possessed by different sets of professionals. Hypertext interfaces are best designed by graphic artists, who have training using HTML design tools. Many Java programmers are excellent in designing Java user interfaces but are not motivated to use graphics design programs that are often required to develop usable HTML interfaces. Therefore, developing an HTML interface will often require the hiring of a graphic artist in addition to the Java programmer developing the application.

High-level interfaces allow local updates whereas HTML interfaces require a whole page refresh. The fact that the whole page needs to be refreshed introduces additional delays the user must deal with. This fact may be a significant usability concern for those who must use the application every day. Users of mission- and time-critical applications, such as financial trading, supervisory control, and data acquisition systems, may find such delays a serious usability issue.

Highly Interactive Applications

Hypertext-based interfaces have been developed as a mechanism for browsing through large amounts of text. They are, therefore, well adapted for browsing text-based documents, such as electronic newspapers and other data delivery systems. But the hypertext model doesn't work well for many highly interactive software applications, such as financial trading systems, software development systems, control systems, and accounting systems. Many Web services are likely to exhibit more characteristics of interactive applications than data delivery systems. Therefore, rich user interfaces may be a better choice for many Web services–based applications.

The advantages of using high-level user interfaces as discussed so far have focused on usability issues, but the architectural advantages of using them may be of greater importance to systems designers. This may be especially true in the case of systems that are built from autonomous Web services components. Rich user interfaces allow separate Web services to be tied together using object-dependency mechanisms.

For example, a text field that is the output of one Web service can be set up to have dependent text field widgets that are inputs to other Web services. Dependency-directed backtracking algorithms can be used to trigger computations that are performed by a number of cooperating Web services.

The Case Against Rich User Interfaces

It's often argued that rich user interfaces are more difficult to develop than hypertext-based user interfaces. The difficulty most often cited is that Java GUIs require the development of a client application, or applet, that supports the interface. Once the application is developed, additional code needs to be written in order to connect the application with a server. The methods of connecting the client and the server have not been standardized. There are a number of possibilities ranging from Remote Method Invocation (RMI) to HTTP/XML-based schemes. In addition, a server-side portion of the application often needs to be developed in order to provide data marshalling and unmarshalling, as well as code that coordinates the logic contained in the client and the logic contained in the server.

Recently, however, a new architecture has emerged that allows a Java client to be deployed directly from servlets and JavaServer Pages (JSP). This architecture is identical to that used for the development of applications that present an HTML interface. The fact that the same architecture can be used for the development of Java and HTML clients suggests the development costs for each type of user interface should be similar. As we have argued earlier, however, development and maintenance costs may be lower for the rich user interface as it's constructed from higher-level reusable components.

Tipping the Scale

Deployment is the single most important consideration that can tip the scales in favor of HTML. HTML interfaces will run without installation on any machine that supports an Internet browser. The penetration of Java Virtual Machine installations, however, is far behind the number of machines that have a browser installed. Java deployment technologies have made significant progress in recent months.

Depoyment technologies such as Sun Microsystems' Java Web Start and Sitraka's Deployment Director can ensure that the client machine is precisely up-to-date. But these technologies require a one-time download of about 5MB and possibly some intervention from the user. That's why HTML may be the better solution for consumer applications that are intended for novice computer users and are deployed over the Internet in an uncontrolled environment. To deploy applications that are used for mission-critical functions over the Internet and intranets, current deployment technologies will prove to be adequate.

Rich Client Systems Architectures

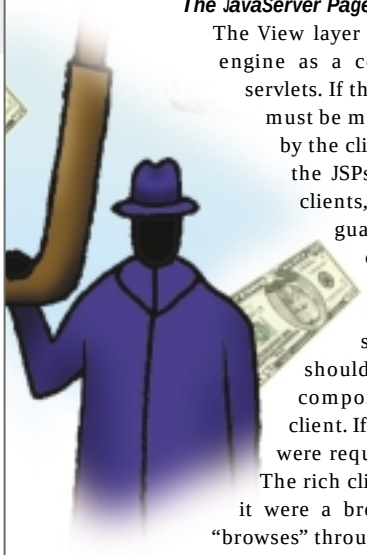
With the advent of Web services, an application architecture based on cooperating Web services has emerged. Modern applications are constructed from Web services that are optimized for a particular purpose. For example, a car rental service can be combined with a flight-booking and hotel-reservation service, a personal calendar service, and a personal wallet service into a single application. We focus on two approaches for connecting cooperating Web services based on the Model View Controller (MVC) pattern in which the cooperating services form the Model layer. In the first approach, the Web services are connected on the server, whereas in the second approach, the services are connected on the client.

Connecting Multiple Application Services

Multiple services can be combined into a single application on the server and presented to the user as a single monolithic application that performs a particular unified service. The user is not aware that the application is actually composed of multiple services and doesn't need to understand the intricacies of each of the services. This architecture is depicted in Figure 1.

The Controller

The integration of multiple cooperating services is the function of the application controller. It's the application controller that must coordinate and tie the services together. The user of the application is then simply presented with a single application interface and isn't aware that the application is composed of several cooperating Web services. The application controller doesn't concern itself with persistence. Persistence is the responsibility of services that reside in the Model layer. The controller can often be programmed as a collection of server-side JavaBeans that connect to the required services and are accessed by the View layer to display a user interface.



The JavaServer Pages View Layer

The View layer can be implemented on top of a servlet engine as a collection of JavaServer Pages (JSPs) or servlets. If the view is composed of JSPs, then the view must be made specific to the output format expected by the client machine. In the case of HTML clients, the JSPs output HTML. In the case of rich Java clients, the JSPs output a program or markup language that can be interpreted by the Java client. In the case of wireless clients, the JSPs output WML. It's important to emphasize here that rich Java clients shouldn't receive special treatment and shouldn't require development of an additional component to handle interaction with the rich client. If development of an additional component were required, additional costs would be incurred. The rich client must therefore be able to behave as if it were a browser and request new JSPs as its user "browses" through the Java Swing user interface.

The XML View Layer

If the View layer is implemented as a collection of servlets, the output from the servlets can be device-specific to the display format used by the client, or it can be made display-format agnostic. If the view servlets are to be made independent of the display device, they must output XML independent of all devices. The XML output must then be transformed into a display-specific markup or script language using an XSLT processor. Again, no special consideration should be given to the rich Java client. The Java client should be able to interpret output from the XSLT processor.

Push-Based Architectures

Certain rich client applications require the ability to push real-time data to the client as opposed to relying on refresh operations. In this case, the client must incorporate a listener component that listens to messages broadcast from the controller. The messages might be encoded in XML or simply contain a script that will then be interpreted by the client to display a user interface.

Remote AWT-Based Architectures

The Remote Abstract Windowing Toolkit (RAWT) architecture is similar to the X Window System pioneered in the Unix operating system. Essentially, RAWT allows an application executing on one machine to display a Java user interface on another. Events trapped on the client machine are transferred to the machine where the program actually executes. This architecture is similar to the thick client architecture, except that the application executes on the server as opposed to a remote client. The difference is, of course, that communication between the presentation layer of the application that displays the user interface and the rest of the application is done using local procedure calls as opposed to remote procedure calls.

The disadvantage of combining multiple application services on the server is that the user is deprived of the ability to combine multiple services in a way not envisioned by the programmer. In certain environments, users require a high level of customization and, often, frequent changes to the user interface and functionality of the application. This is the case when a new application is being developed and its users modify requirements as they learn more about how they could use the application to perform a business function.

The Application Canvas

The application canvas is the software that makes the integration of separate Web services possible on the client. The application canvas is like a modern-day spreadsheet that enables the output from one Web service to be tied to the input of another. The application canvas directs computation using a dependency-directed backtracking algorithm that computes the values of each individual cell – just as a spreadsheet would.

A key advantage of using an application canvas is that each Web service can now maintain its own separate user interface that allows the Web service to be used regardless of the presence of other services. That is, each Web application service can be displayed on the canvas and used independently. This frees system developers from building a server-side controller as the controller actually resides on the client in this case.

The user interface displayed by each Web service should be a rich user interface that allows the programmers and the users to create dependencies between output cells of one service and input cells of another.

Entirely Dynamic, Entirely Independent

To provide the elements of functionality needed to integrate separate application services into a useful application, the application canvas must incorporate most, if not all, of the following components.

The application canvas must provide support for JSP- and servlet-based rich clients. A desktop-like interface akin to that of the Swing JDesktopPane is a good choice for displaying windows that belong to autonomous Web services. It's important the user interface presented by each Web service is entirely dynamic so that the application canvas can remain entirely independent of the Web services it will manage.

Another important component of the application canvas is the dependency database that tracks dependencies for each input cell. The database must provide storage that persists across user sessions. The dependency database must be an embeddable database with a small memory footprint.

The dependency backtracking algorithm allows the application canvas to detect changes in output of one Web service and feed these changes to the input fields of another Web service.

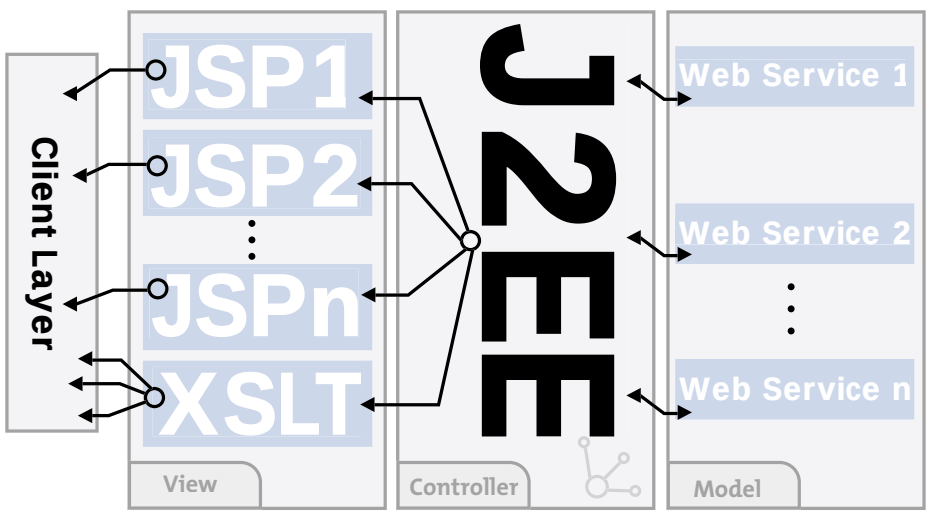
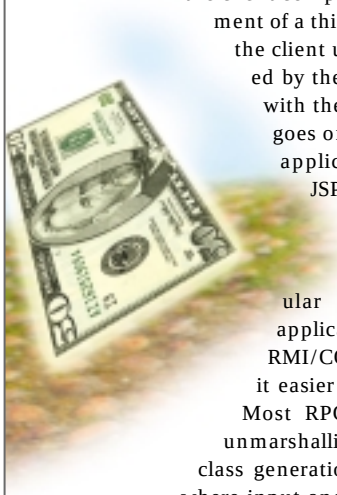


FIGURE 1 Model-View-Controller architecture integrating Web services in the controller

The Thick Client



In cases where the client application must operate when the client computer isn't connected to the network, development of a thick client may be the only option. In this case, the client usually is required to store some data provided by the user. Often, the client will be synchronized with the server or with its peers when the computer goes online again. There are at least three classes of applications in this category: RPC-based, JSP/servlet-based, and P2P.

RPC-Based Clients

Remote procedure calls have been a popular means of developing client/server-based applications. Many RPC-based frameworks, such as RMI/CORBA, provide high-level functions that make it easier for developers to build RPC-based systems. Most RPC systems provide data marshalling and unmarshalling as well as toolkits for automatic proxy class generation. RPC is sometimes combined with XML where input and output parameters are encoded in XML as opposed to a binary format.

An important design consideration that arises when building RPC-based clients is that the client and server will need to contain additional components to interpret RPC messages on the server and on the client. Development and maintenance of these components may prove expensive. Whenever possible, using servlets and JSP communication should be considered.

JSP/Servlet-Based Thick Clients

JSP/servlet-based thick clients distinguish themselves from RPC-based clients in that the client is actually composed of two parts: the static part that's not dependent on the network connection and the dynamic part that is. When the client computer is not connected to the network, the client application executes the static part. When the client is connected, the dynamic part of the application based on the JSP/servlet architecture becomes active. In essence, this architecture is a thick Java client with an embedded thin Java client that's using servlets and JSPs to display a user interface.

The advantage of combining the thick client with a dynamic thin client is that the communication component on the client and server can now be eliminated from the design. But the application may still need to synchronize its data with a server or Web service.

P2P Framework

The most promising data synchronization approach is perhaps Sun Microsystems' JXTA P2P framework. The JXTA framework is an excellent synchronization data mechanism for the parts of an application that act as peers. RPC is an alternative to the P2P approach. In most cases, however, the RPC-based approach is likely to result in the development of more custom network interface code – resulting in higher development and maintenance costs.

Deployment Strategies

Once you have the rich client solution, you need to get it to the end user. Various techniques are available, each with its own pros and cons.

- ***Applet:*** Deploying the Java Swing user interface as an applet requires the end user's browser to have the Java Plug-in installed. The cost of requiring the Plug-in is a one-time download for each user of a 5MB–15MB setup file, where the size varies based primarily on platform. (Microsoft Windows is at the low end.) In addition, using the applet may or may not require changes to the HTML file to load. This depends upon the Plug-in version supported and the user's browser. Using an applet as the delivery mechanism does ensure that the user has the latest version of the client. Each time a user connects up to the back end, the applet can be downloaded. The Java Plug-in supports caching, so it won't have to download every time.
- ***Install Programs:*** InstallAnywhere and InstallShield are two popular solutions for deployment. These tools and others like them offer the ability to provide a complete Java Runtime Environment (JRE), the same 5MB–15MB download as the Java Plug-in, in addition to your rich client application. Installation via these solutions is something users are familiar with, but it can lead to users having multiple JREs on their desktops, one for each installable object. The major drawback of these programs is the coordination of new releases. You as the vendor must actively notify all users of the update and then each user must download the new release. New versions of these products are thankfully adding active update capabilities, so this last issue can be resolved.
- ***JNLP:*** Java Web Start is Sun's latest attempt to solve the incompatible browser runtime environment and deployment issues. It uses a protocol called *Java Network Launching Protocol* (JNLP) to provide for the installation of applications outside the browser but within a secure execution environment. It requires an HTML page for the initial load of the application but, once loaded, permits the browser to be shut down. There is also built-in support for incremental updates. Sitraka's DeployDirector is another JNLP-based deployment alternative that adds additional features.

Commercial and Free Software

A number of commercial and free software products have emerged that make it easier for Web services developers to take advantage of rich GUIs. Several of these are mentioned, however, this list includes only some of the more popular ones and is not intended to be exhaustive.

- ***Altio's flagship product AltioLive*** is a framework for the development of rich clients that execute as applets within a Web browser. Their front end uses an extensive proprietary widget set compatible with Java installations shipped with Internet browsers. The framework also includes a presentation server that drives the user interface. The presentation server uses proprietary protocols to communicate with the client. AltioLive also includes a design tool for developing the graphical user interface.
- ***Bean Markup Language (BML)*** is an instance of an XML-based language customized for the JavaBean component model. The language allows beans to be constructed and wired together. The language is designed to be directly executable. That is, processing a BML script results in the construction of a running Java application as described by the script. The BML language has elements for creating new beans, accessing existing beans, and executing bean methods.
- ***Droplets User Interface Server*** is a presentation server that extends some of the ideas developed by the Remote AWT project at IBM. In the case of the Droplets Server, a C++ API is provided in addition to the Java API. Droplets also offer some prebuild servers that handle a number of applications with rich Java GUIs that include e-mail, customer care management, and content management. Droplets also

include an innovative deployment mechanism that allows application icons to be dragged from a Web page onto a user's desktop.

- ***The Remote Abstract Window Toolkit*** (AWT) for Java is a server-side implementation of AWT that allows any application running on one host to display a Java user interface on another host. The AWT calls made on the server are transmitted to the client for processing. That's why an AWT application running on the server can actually display its user interface on the client. The RAWT toolkit is available from the IBM alphaWorks Web site.
- ***Spidertop*** provides tools for the development of JSP and servlet-based Java user interfaces. The flagship product, called *Bali*, includes an interpreter JavaBean component that interprets a scripting language that can be served from servlets and JSPs to display Java Swing user interfaces. The framework also includes a graphical JSP builder incorporated into Sun Microsystems' Forte for Java that allows developers to build JSPs graphically. The framework also includes a presentation server that simplifies development of server-side controller components. The use of the presentation server is optional as the interpreter JavaBean can communicate with servlets and JSPs without the intervention of the presentation server.

Four Good Reasons

We have identified four factors that will drive the adoption of rich user interfaces for Web services–based applications.

The first factor is the emergence of a new class of architectures that permits the development of rich front ends based on servlet and JSP J2EE back-end systems. When using servlet and JSP-based architectures, development costs of rich GUIs are comparable to development costs of HTML-based front ends, while all the benefits of the Java GUIs are maintained.

The second factor is the interactive nature of Web services–based applications. Many Web services–based applications will exhibit more characteristics of typical computer applications than characteristics of information delivery systems. Highly interactive systems are better presented using rich Java front ends than HTML.

The third factor is the advances in deployment technologies for rich Java GUIs.

The fourth factor is the emergence of the application canvas. The application canvas will give developers a compelling reason for developing rich Java GUIs for Web services, as the application canvas permits the end user to tie separate Web services into composite applications not foreseen by the Web service developers.

We believe these four factors will result in a widespread adoption of rich Java GUIs for Web services–based applications. ✎

Author Bios

John Zukowski conducts strategic Java consulting with JZ Ventures, Inc., and serves as the chief evangelist for Spidertop. His latest books are Java Collections *and* Definitive Guide *to* Swing for Java 2 *(2nd ed) from Apress.*

Vincent Maciejewski is the chief architect at Spidertop, Inc. His research interests include component-based computing and intelligent user interfaces.

jaz@zukowski.net & vincent@spidertop.com

SYS-CON MEDIA

PUBLISHER, PRESIDENT, AND CEO
FUAT A. KIRCAALI fuat@sys-con.com

VICE PRESIDENT, BUSINESS DEVELOPMENT
GRISHA DAVIDA grisha@sys-con.com

ADVERTISING

SENIOR VICE PRESIDENT, SALES AND MARKETING
CARMEN GONZALEZ carmen@sys-con.com

VICE PRESIDENT, SALES AND MARKETING
MILES SILVERMAN miles@sys-con.com

ADVERTISING SALES DIRECTOR
ROBYN FORMA roybn@sys-con.com

ADVERTISING ACCOUNT MANAGER
MEGAN RING megan@sys-con.com

ASSOCIATE SALES MANAGERS
CARRIE GEBERT carrieg@sys-con.com
KRISTIN KUHNLE kristen@sys-con.com
ALISA CATALANO alisa@sys-con.com

EDITORIAL

EXECUTIVE EDITOR
M'LOU PINKHAM mpinkham@sys-con.com

EDITOR
NANCY VALENTINE nancy@sys-con.com

MANAGING EDITOR
CHERYL VAN SISE cheryl@sys-con.com

ASSOCIATE EDITORS
JAMIE MATUSOW jamie@sys-con.com
GAIL SCHULTZ gail@sys-con.com
BRENDA GREENE brenda@sys-con.com

ASSISTANT EDITOR
LIN GOETZ lin@sys-con.com

PRODUCTION

VICE PRESIDENT, PRODUCTION AND DESIGN
JIM MORGAN jim@sys-con.com

ART DIRECTOR
ALEX BOTERO alex@sys-con.com

ASSOCIATE ART DIRECTORS
LOUIS F. CUFFARI louis@sys-con.com
CATHRYN BURAK cathyb@sys-con.com

GRAPHIC DESIGNERS
RICHARD SILVERBERG richards@sys-con.com
AARATHI VENKATARAMAN aarathi@sys-con.com

WEB SERVICES

WEBMASTER
ROBERT DIAMOND robert@sys-con.com

WEB DESIGNERS
STEPHEN KILMURRAY stephen@sys-con.com

CHRISTOPHER CROCE chris@sys-con.com

ACCOUNTING

CHEIF FINANCIAL OFFICER
BRUCE KANNER bruce@sys-con.com

ASSISTANT CONTROLLER
JUDITH CALNAN judith@sys-con.com

ACCOUNTS RECEIVABLE
JAN BRAIDECH jan@sys-con.com

ACCOUNTS PAYABLE
JOAN LAROSE joan@sys-con.com

ACCOUNTING CLERK
PATTI DELVECCHIO patti@sys-con.com

SYS-CON EVENTS

VICE PRESIDENT, SYS-CON EVENTS
CATHY WALTERS cathyw@sys-con.com

CONFERENCE MANAGER
MICHAEL LYNCH mike@sys-con.com

SALES EXECUTIVES, EXHIBITS
MICHAEL PESICK michael@sys-con.com
RICHARD ANDERSON richard@sys-con.com

SHOW ASSISTANT
NIKI PANAGOPOULOS niki@sys-con.com

REGISTRARTION ASSISTANT
JACLYN REDMOND jaclyn@sys-con.com

JDJSTORE.COM
ANTHONY D. SPITZER tony@sys-con.com

Enabling Constant Substitution in Property Values

A useful extension to the properties facility in Java

WRITTEN BY
CHRIS MAIR

XProperties is a simple subclass of java.util.Properties that allows you to treat property values like constants, referring to and embedding them inside other property values. It handles the substitution of the constant value automatically and transparently when you call the getProperty() method.

Using constants in your property files can make them easier to read and maintain. This article describes an extension to the properties facility in Java that greatly enhances its functionality and usefulness.

Like many other Java programmers I use the Properties class extensively for managing configuration information. In an application I recently developed, I had several properties (database URLs) in a configuration property file that shared the same value. I wanted a way to specify a constant in a property file rather than repeating the same value over and over.

My first approach was to add some extra logic where I read in the properties. I designated constant values by an arbitrary prefix ($). As I accessed a property value, I checked if it began with the prefix and, if so, performed the substitution with the named constant value.

I soon realized, however, that I could generalize this capability by moving the logic into a subclass of java.util.Properties. I could also greatly increase its usefulness by not limiting it to a simple, single substitution. Rather, I allowed a constant value to be embedded anywhere within the property value, even allowing multiple constants within the same value. I chose to indicate constant names by enclosing them in braces, abandoning the single prefix character. This makes the constant names obvious and unambiguous, as well as easier to parse.

The following properties illustrate the use of a constant within a property value:

```
SOURCE = SampleApp
ErrorMessage = {SOURCE}: An error
has occurred
```

The SOURCE property defines a constant value that's embedded within the value of the ErrorMessage property. As I mentioned earlier, to indicate that you wish to substitute the value of another property, surround its name with braces. Notice that the SOURCE property is defined just like any other property value. The only thing that makes it a "constant" is that it's referenced by another value. I've used all capital letters for the constant value, as is the convention in Java, but this is by no means a requirement. If the property name within the braces isn't found, then no substitution is performed and the property value is returned without modification.

You can substitute a constant anywhere in the property value, and even have more than one constant within a value, as in the following example:

```
CONST_1 = shoes and ships
CONST_2 = sealing wax
SomeValue = {CONST_1} and {CONST_2}
```

In this example, the "SomeValue" property evaluates to "shoes and ships and sealing wax."

Using constants within property values is especially useful when the same value occurs in multiple places, or when you build up a complex value from several reusable parts (e.g., file pathnames or URLs). Using constants also allows you to enumerate and assign descriptive names to potential property values. You may have to switch frequently between values for a property while developing a system, for instance, the URLs for unit test, system test, and production. One approach I've used is to include a property assignment for each possible value, but comment out the ones not being used. For example:

```
# MyURL = http://localhost:8080/abc
# MyURL = http://testserver.acme.
          com/abc
MyURL= http://prodserver.acme.com
       /abc
```

Without extra comments within the property file, however, it's unclear what each alternative represents. The extra commented lines also clutter the file, making it harder to read. Instead, by defining each alternative as a named constant, you can assign descriptive names to all the values, making the property files easier to understand, and also lessen the chance of making errors when switching between values. For instance, you can define:

```
UNIT_TEST_URL = http://local
  host:8080/abc
SYSTEM_TEST_URL = http://testserv-
  er.acme.com/abc
PRODUCTION_URL=http://prodserver.ac-
   me.com/abc
MyURL = {UNIT_TEST_URL}
```

In the above example, it's self-evident that the "MyURL" property is set to the value appropriate for the unit test environment. This example is included with several others in the sample property file shown in Listing 1. It illustrates some different ways to use constants within property values. Listing 2 contains Java source code to read in and access that property file using the XProperties class. Listing 3 shows the resulting output from running the code in Listing 2.

Note that the syntax for using the XProperties class – creating an XProperties object, loading the properties from a file, and accessing the property values – is exactly the same as for

the java.util.Properties class. This demonstrates the power of inheritance. The XProperties class is designed to be a drop-in replacement for java.util.Properties. No changes or additions are made to the public API of the superclass. The XProperties class handles the constant substitution under the covers when you access a property value using the getProperty() method.

Another point to note is that you can nest the constant substitution to an arbitrary level. The XProperties class recursively evaluates each referenced constant. In Listing 1, the value of the property "MyHtmlFile" includes the value of the property "HTML_DIR". This, in turn, references the value of property "BASE_DIR", which then references "ROOT_DRIVE". You can see in Listing 3 that the value of "MyHtmlFile" evaluates to "c:/base/html/MyFile.html", incorporating all the constants used along the way.

Listing 4 contains the source code for the XProperties class. As you can see, it's not very complicated. The XProperties class overrides the getProperty(String) method. Handling the recursive substitution presented a minor challenge. How do you prevent an infinite recursive loop, as in the following example?

```
A={B}
B={A}
```

One option is to perform a complicated analysis to detect loops in the substitution tree. I chose a simpler approach. The XProperties class keeps track of the current depth of substitution (starts at zero and increments with each recursive substitution), and only allows up to an arbitrary (but easily changed) maximum depth – currently five. This logic is implemented in the private getProperty(String,int) method. This method performs the parsing and constant substitution, handles recursive invocation, if necessary, and checks the maximum substitution depth.

Like its superclass, the XProperties class includes a constructor that takes an existing properties object. These are default properties that are searched if a property key isn't found in the original property list. One use for this capability is to define a separate property file containing common constants that can then be accessed from multiple property files. All you need to do is load the property file containing the constants into a properties (or XProperties) object and then pass this object as the parameter to the XProperties constructor.

Be aware that the Properties.list(..) methods don't use the getProperty() method, and so don't perform any constant substitution before displaying property values.

Property files, like programs, can benefit from defining and using constants. You can specify a value once, and then reuse the same value for multiple properties, or you can have property values that are composed of several (potentially reusable) parts. These capabilities can be especially useful in configuration property files in which you need to specify complex values, like file pathnames or resource URLs. You can also enumerate and assign descriptive names to potential values when you need to frequently switch between alternative values for a property, for example, switching between test and production database URLs. The implementation of the XProperties class is fairly simple, but it provides a very useful extension to the properties facility in Java.

cmair@us.ibm.com

Listing 1: Sample Properties

```
# Sample.properties
#
# These properties illustrate the use of constants
# within property values, accessed using the
# XProperties class.

# Use several levels of recursive substitution to
# define paths for different resource types
ROOT_DRIVE=c:
BASE_DIR={ROOT_DRIVE}/base
HTML_DIR={BASE_DIR}/html
XML_DIR={BASE_DIR}/xml
IMAGE_DIR={BASE_DIR}/image

# Define full pathnames for resource files
MyHtmlFile={HTML_DIR}/MyFile.html
MyXmlFile={XML_DIR}/MyFile.xml
NewImage={IMAGE_DIR}/new.gif
SaveImage={IMAGE_DIR}/save.gif
PrintImage={IMAGE_DIR}/print.gif


# Assemble a property value from several constants
PROTOCOL=https
SERVER=server.acme.com
PORT=80
RESOURCE=index.html
HomePageURL={PROTOCOL}//{SERVER}:{PORT}/{RESOURCE}


# Define a constant for each possible value
UNIT_TEST_URL = http://localhost:8080/abc
SYSTEM_TEST_URL = http://testserver.acme.com/abc
PRODUCTION_URL= http://prodserver.acme.com/abc

# Select from the possible values
MyURL = {UNIT_TEST_URL}
```

Listing 2: Sample Java Code to Use XProperties

```
// import java.io.*;
// Create an XProperties and load the property file
Properties prop = new XProperties();
InputStream in = new FileInputStream("Sample.properties");
prop.load(in);

// List the keys for the property values of interest
String[] keys = {
 "MyHtmlFile", "MyXmlFile",
 "NewImage", "SaveImage", "PrintImage",
 "HomePageURL", "MyURL" };

// Iterate through the keys; display key and value
for (int i=0; i < keys.length; i++) {
 String value = prop.getProperty(keys[i]);
 System.out.println(keys[i]+" = "+value);
}
```

Author Bio

Chris Mair is a senior consultant and software craftsman at IBM with 12 years' experience in the financial sector. He's a certified Java developer and specializes in object-oriented design and development, frameworks, and J2EE.

Listing 3: Resulting Output from Code in Listing 2

```
MyHtmlFile = c:/base/html/MyFile.html
MyXmlFile = c:/base/xml/MyFile.xml
NewImage = c:/base/image/new.gif
SaveImage = c:/base/image/save.gif
PrintImage = c:/base/image/print.gif
HomePageURL = https//server.acme.com:80/index.html
MyURL = http://localhost:8080/abc
```

Listing 4: Source Code for XProperties Class

```
import java.util.Properties;
/**
 * A subclass of Properties that allows recursive
 * references for property values. For example,
 *
 * <pre><code>
 * A=12345678
 * B={A}90
 * C={B} plus more
 * </code></pre>
 *
 * will result in <code>getProperty("C")</code>
 * returning the value "1234567890 plus more".
 *
 * @author: Chris Mair
 */
public class XProperties extends Properties {

   // The prefix and suffix for constant names
   // within property values
   private static final String START_CONST = "{";
   private static final String END_CONST = "}";

   // The maximum depth for recursive substitution
   // of constants within property values
   // (e.g., A={B} .. B={C} .. C={D} .. etc.)
   private static final int MAX_SUBST_DEPTH = 5;

/**
 * Creates an empty property list with no default
 * values.
 */
public XProperties() {
   super();
}

/**
 * Creates an empty property list with the
 * specified defaults.
 * @param defaults java.util.Properties
 */
public XProperties(Properties defaults) {
   super(defaults);
}

/**
 * Searches for the property with the specified
 * key in this property list. If the key is not
 * found in this property list, the default
 * property list, and its defaults, recursively,
 * are then checked. The method returns
 * <code>null</code> if the property is not found.
 *
 * @param   key   the property key.
 * @return  the value in this property list with
 *     the specified key value.
*/
public String getProperty(String key) {

   // Return the property value starting at level 0
   return getProperty(key, 0);
}

/**
 * Searches for the property with the specified
```

```
 * key in this property list. If the key is not
 * found in this property list, the default
 * property list, and its defaults, recursively,
 * are then checked. The method returns
 * <code>null</code> if the property is not found.
 *
 * <p>The level parameter specifies the current
 * level of recursive constant substitution. If
 * the requested property value includes a
 * constant, its value is substituted in place
 * through a recursive call to this method,
 * incrementing the level. Once level exceeds
 * MAX_SUBST_DEPTH, no further constant
 * substitutions are performed within the
 * current requested value.
 *
 * @param   key   the property key.
 * @param level  the level of recursion so far
 * @return  the value in this property list with
 * the specified key value.
 */
private String getProperty(String key, int level) {

   String value = super.getProperty(key);
   if (value != null) {

      // Get the index of the first constant, if any
      int beginIndex = 0;
      int startName = value.indexOf(START_CONST, beginIndex);

      while (startName != -1) {
         if (level+1 > MAX_SUBST_DEPTH) {
            // Exceeded MAX_SUBST_DEPTH
            // Return the value as is
            return value;
         }

         int endName = value.indexOf(END_CONST, startName);
         if (endName == -1) {
            // Terminating symbol not found
            // Return the value as is
            return value;
         }

         String constName = value.substring(startName+1, endName);
         String constValue = getProperty(constName, level+1);

         if (constValue == null) {
            // Property name not found
            // Return the value as is
            return value;
         }

         // Insert the constant value into the
         // original property value
         String newValue = (startName>0)
            ? value.substring(0, startName) : "";
         newValue += constValue;

         // Start checking for constants at this index
         beginIndex = newValue.length();

         // Append the remainder of the value
         newValue += value.substring(endName+1);

         value = newValue;

         // Look for the next constant
         startName = value.indexOf(START_CONST, beginIndex);
      }
   }

   // Return the value as is
   return value;
}
}
```

Download the Code! www.JavaDevelopersJournal.com

JASON BRIGGS J2ME EDITOR

A Dodgy Character

After what seems like months of waiting (and it probably has been), a color mobile phone (as opposed to a PDA/phone combination) has finally appeared on the British market. With a 101x80 pixel 256-color STN display, an e-mail application, calendar, games, and PC synchronization, it has everything you might want in a mobile...hang on a minute...where's the Java?

Gah!!

The manufacturer of this electronic travesty will remain anonymous. Suffice it to say I have purchased a voodoo doll in the shape of the son of the Viking called Eric and I'm stabbing it with large – blunted – pins every time I feel particularly aggrieved.

Thankfully, Motorola has also released a new phone in the UK – the Accompli 008. Unfortunately, it only has a monochrome screen, but does include unhealthy levels of caffeine. Look for a review of the A008 in a future issue of ***JDJ***. (If you're an American I'm sure you'll be pleased to know Motorola has also released two new phones in the U.S. – the i90c and i80s.)

It has been a busy month in terms of Java and J2ME press releases. Nextel and Motorola have announced programs for the purchase, management, and (wireless) download of applications for their Java-enabled mobile phones. Also Motorola announced the imminent availability of Cybiko games, again for their Java-enabled handsets. For those of you who haven't come across Cybiko before (www.cybiko.com), the company manufactures and markets consumer electronics, and operating system and software products, including a wireless handheld computer (also called Cybiko) aimed at the youth market.

From Compaq came the news that the iPAQ H3800 series Pocket PC will ship with Insignia Solutions' Jeode PersonalJava Virtual Machine. Up until now, users had to download Jeode (reviewed in ***JDJ***, Vol. 6, issue 8) from a Web site after their purchase, so it's exciting to see it available with the shipping unit – removing an obvious impediment to installing Java on the devices. Hopefully we'll see more PDA manufacturers following suit.

Although I haven't seen any official word about it, I came across an excellent site awhile ago for anyone hunting for free MIDlets: http://midlet.org has categories for games, graphics, Internet, business, general utilities, and developer MIDlets. At the moment there are just under 100 applications available for download, and it looks as if a new application is uploaded every couple of days. Well worth a look.

On November 1, Aligo, Inc., announced the first annual Wireless Application Development contest. If you haven't heard about it already, the contest runs until January 15, 2002, and the first prize is $5,000. Entrants must submit an application written in Java or JSP, and deploy it on Aligo's M-1 Mobile Application Server; a panel of judges will select the three best applications. It's worth noting that one of these judges is ***JDJ***'s own editor-in-chief Alan Williamson. He is, of course, a thoroughly dodgy character, and I'm sure open to all sorts of bribery; anything from large quantities of money (in small, unmarked bills) to compliments on his hair. Don't be surprised if your generous bribe doesn't affect the outcome of the contest, however. It will undoubtedly go into the refurbishment of his bolt-hole in the Azores.

In this edition of ***JDJ***, we look at Zucotto's Bluetooth SDK, and we review the book *Wireless Java: Developing with Java 2, Micro Edition* by Jonathan Knudsen. James Caple provides an insight into his experiences building a Palm Pilot–powered robot based upon a Carnegie Mellon University design; for those experiencing a profound sense of déjà vu, James' article provides an interesting alternative to the Lego Mindstorm robot introduced by David Hardin and Mike Frerking back in October (***JDJ***, Vol. 6, issue 10). In addition, Glen Cordrey presents a treatise on his top 10 limitations in the Mobile Information Device Profile API and discusses some of the alternatives (when they exist). ✎

jasonbriggs@sys-con.com

AUTHOR BIO

Jason Briggs works as a Java analyst programmer in London. He's been officially developing in Java for three years – unofficially for just over four.

J2ME INDEX

A Waba-Powered Palm Pilot Robot

Someday they might say it all began with the Mexican Hat Dance

Written by James Caple

Back when Java was called *Oak*, it was thought that this new language would be ideal for developing embedded applications, such as those that would run on set-top boxes. The developers of this new language were well ahead of their time. Java's momentum began to build not from its large set-top developer community but from developers wishing to enhance their Web sites using Java applets. Thank goodness that was short-lived!

Java then became popular in server-side applications, but it has only recently begun to gain popularity in the embedded devices it was originally intended for. Java is now running on, and in, everything from big SMP servers to portable devices, such as PDAs, phones, and even smart cards.

Like many Java enthusiasts, I'm interested in exploring the capabilities of Java programs on small-footprint devices, especially my own Palm Pilot. Sometimes the best way to learn more about a new technology, or to develop a new skill, is to just mess around with stuff. Some refer to this as hacking (not to be confused with cracking, which is breaking into computer systems illegally).

Not too long ago I discovered plans for a Palm Pilot Robot at Carnegie Mellon University's School of Computer Science. I downloaded the plans, bought all the necessary parts (except for the Palm Pilot, which I already had), and started putting it together. After I got the robot assembled and sandwiched together, I immediately wanted to see if I could control it with a Java program.

I learned a great deal while hacking a few Java programs to control various parts of the Palm Pilot Robot Kit (PPRK) hardware, and much of what I learned can be applied directly to embedded applications.

For example, I decided to construct a simple framework that attempts to make programming robot software, for robots similar to the PPRK, much easier.

This article discusses the lessons learned in writing Java software to make this robot come alive – well, sort of alive.

Java programs running on your favorite handheld device can serve as the brains for your robot creation.

Waba

Today there are several Java, or Java-like, Virtual Machines (VMs) for running embedded Java applications. Waba is one such VM. The Waba SDK provides a Java-like development environment for small-footprint devices and is available under the terms of the GNU Public License (GPL). Because it's an open source VM implementation, a large group of developers continue to enhance and improve the Waba API on a project called *SuperWaba*.

I didn't do extensive research on what VM I wanted to use and run on my PPRK. I had some experience with Waba already and have found it easy to understand, download, and install. The fact that it's distributed under the GPL is also in its favor. Furthermore, the SuperWaba newsgroup is very active – and developers, such as Guihlerme Campos Hazan (lead SuperWaba developer), are friendly and quick to answer newbie developer's questions. These types of development communities make it easier to get up to speed on new and emerging technologies.

After assembling my PPRK (see Resources section) and choosing a VM for my Palm V that could run Java-like programs, I was ready to start writing some Java code.

It's All About the Serial Port

The key to writing a Waba application to control your Palm

Pilot Robot is a good understanding of the following:

1. The protocol the Pontech SV203 servo controller board "speaks"
2. How to use the Waba SerialPort class

The Palm Pilot Robot, designed by Grigoriy Reshko at Carnegie Mellon University, uses the Pontech SV203 servo motor controller board to control the three onboard servo motors and to gather data from the three infrared sensors located by each motor. The communication protocol of the Pontech SV203 servo motor controller board is straightforward. When building a PPRK from scratch, you must modify one of your extra HostSync cradles by lopping off the end of the cable and fitting it with a male DB-9 connector. This connector plugs into the female end of the SV203 board. Thus, your modified HostSync cradle is the means by which your Palm device can communicate with the SV203 board.

As an aside, if you decide to buy the PPRK kit from Acroname, which I recommend, this modified HotSync cable is provided for you in the kit. (*Note:* You can download a PDF version of the SV203 manual from www.pontech.com to learn more about the SV203 design and protocol specifications.)

Understanding how to first send commands from your PC to your SV203 board using a serial cable (at your local Radio Shack) and a terminal emulation program is important. I have documented some notes and procedures on how to do this, as well as notes for debugging your PPRK hardware and software, at www.trexlabs.com/pprk.

Communicating with the SV203 is quite simple. You need to tell the board what servo motor you wish to control and in what direction and speed you wish to move it. You do this by generating a string of ASCII characters that represent a command and the device number you wish to control, terminating the string with a "\r".

For example, the command SV1 M55 turns servo 1 counterclockwise (counterclockwise movement is any number between 0 and 127, and clockwise movement is any number between 128 and 255. Also note that the motors turn slower the farther away from 0 up to 127 you get, and faster the farther from 128 up to 255 you get).

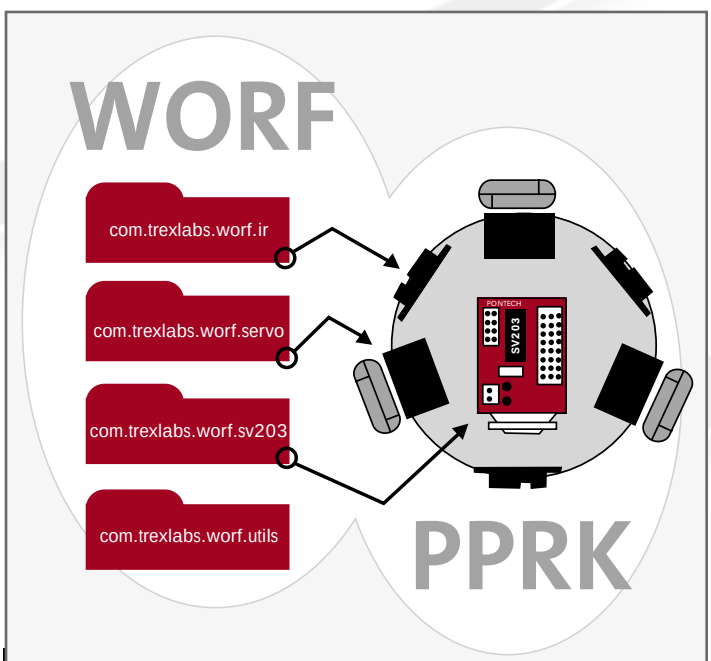


FIGURE 1 The WORF packages and how each component relates to the hardware composing the PPRK

Once you establish a connection between your SV203 and your PC's terminal emulation program (on Linux I use minicom), you can type the command above, hit return (enter), and servo motor 1 should begin a clockwise rotation. To stop the motor, enter the command SV1 M0 and hit return (enter).

The PPRK has three motors, so you can control all three by making the appropriate substitutions in the command sent to the SV203 board. Now all you need to figure out is how to write these same commands to the serial port in our Waba program.

Waba makes this easy to do. As mentioned earlier, Waba has a class in the waba.io package called *SerialPort*, which you'll use to open a connection to the SV203 controller board serial port. This class can be constructed in two ways:

```
SerialPort(int port_number, int baud_rate)
SerialPort(int port_number, int baud_rate, int bits,
boolean parity, int stopbits)
```

The first constructor constructs the SerialPort object using 8 bits, no parity, and 1 stop bit as the default, so you can use the following to establish a serial connection with the SV203 controller board:

```
SerialPort sp = new SerialPort(0, 9600);
sp.setFlowControl(false);
```

Now, you need to populate an array of bytes containing the appropriate commands, and then write that array to the serial port. If you want to write the commands SV1M55, you can accomplish it like so:

```
byte[] buff = new byte[7];
buff[0] = (byte)'S';
buff[1] = (byte)'V';
buff[2] = 49;                // ASCII equivalent of 1.
Very important!
buff[3] = (byte)'M';
buff[4] = 53;                // ASCII equivalent of 5.
buff[5] = 53;
buff[6] = (byte)'\r';    // End of command
```

Now you're ready to write this array of bytes to the serial port object instance, sp, which was created earlier. The following code snippet uses our SerialPort instance to write the 7-byte array that contains the command to the Palm's serial port and out to the SV203 board:

```
sp.writeBytes(buff, 0, 7)
```

After your Waba application has sent this command in the manner described previously, your servo motor 1 on your PPRK should begin spinning lazily along in a counterclockwise direction. Again, you can stop it by issuing the SV1 M0 command in a similar manner.

Give the Gift of Sight

The PPRK uses three Sharp GP2D12 Infrared Object Detectors, which are also connected to your SV203 board. These little infrared sensors are used to detect nearby objects. The GP2D12 IR sensor will change the voltage on a given SV203 port, based on the distance, in centimeters, a given object is from the sensor. You communicate with these sensors the same way you communicate with the servos, except this time you must also read a 4-byte response from the board. The SV203 protocol for sending a request to the IR sensor is simply to send the following ASCII command: AD1\r. This tells the SV203 board to take a reading from the first IR sensor

and to send 4 bytes back to our Palm device indicating the range, in centimeters, of the closest object, if any. You can create a 4-byte array and populate it with the command like so:

```
byte buff[] = new byte[4];
buff[0] = (byte)'A';
buff[1] = (byte)'D';
buff[2] = 49; // ASCII equivalent of 1.
buff[3] = (byte)'\r';
```

You then send the 4-byte command using the SerialPort instance created earlier. You need to be aware of one subtle nuance before doing this. Your Waba program can write and read to the serial port faster than the information can be processed by the SV203 board, so you need to give the board time to digest its contents when reading data back from the board. The SV203 User's Manual suggests a sleep time of approximately 3 milliseconds, but give it a little more just to be sure. You can do this with Waba using the waba .sys.Vm.sleep() static method. The parameters to this method indicate the number of milliseconds the VM should sleep.

```
sp.writeBytes(buff, 0, 4);
waba.sys.Vm.sleep(15);
```

You should expect the SV203 board to send some information back at this point, hopefully in the form of 4 bytes. These 4 bytes will contain a single-digit, two-digit, or three-digit number followed by a \r. *Note:* The largest value you should get back from the SV203 is 255 and the smallest is 0. Here's one way you can read the results and set the value in a waba.ui.Label object:

> "Sometimes the best way to learn more about a new technology, or to develop a new skill, is to just mess around with stuff"

```
Label lblIRVal = new Label("");
sp.readBytes(buff, 0, 4);
StringBuffer sbsensorin = new StringBuffer();
for (int i=0;i<4; i++) {
if ( ( buff[i] != 0x0 ) && ( buff[i] != (byte)13 ) &&
(buff[i]  != (byte)10 ) )
sbsensorin.append( (char) buff[i] );
}
lblIRVal.setText( sbsensorin.toString());
```

The Waba Open Robot Framework (WORF)

It's a useful exercise to take the code written thus far to create an API Framework that would allow other hackers to quickly and easily develop software for their own robot creations. The idea behind the Waba Open Robot Framework is to create Java components for each major hardware component composing the PPRK, while at the same time trying to keep it flexible and extensible enough to accommodate other robot configurations. WORF is an open source project and is governed under the terms of the LGPL. You can grab a snapshot of WORF from http://worf.sourceforge.net.

One of the truly neat things about Waba is the availability of the Waba VM on several different embedded platforms, such as Palm, iPaq, Apple's Newton, and even DOS. While I have not tested the portability of WORF across these platforms, it's certainly conceivable that robots can be built using any of these devices and powered with Waba programs built on top of WORF.

The PPRK consists of three servo motors, three GP2D12 infrared sensors, and a Pontech SV203 Controller Board to coordinate and control these devices. WORF, therefore, consists of Java components for each, and provides a burgeoning API set for easily sending messages to the SV203 board on behalf of a given component. The utils package provides miscellaneous support functions (see Figure 1).

With WORF, developers of embedded robot software applications can focus more on the creative aspects of what they want their robot to do and less on the details of how to communicate with this specific type of hardware. This is accomplished by encapsulating the details of sending the specific commands described earlier to the SV203 board and making this functionality available vis-à-vis an easy-to-use set of APIs.

SV203 Component

The SV203 Waba component extends, or subclasses, the waba.io.SerialPort class and provides three constructors, each of which provides the developer with a different means of initializing the parent SerialPort class. The SV203 Waba component maps to the Pontech SV203 controller board piece of hardware. The best reason to morph the waba.io.SerialPort class into the SV203 class, rather than just use the SerialPort object provided by Waba, was the need to wait at least 3 milliseconds for the SV203 board to process commands that require a response. While this sleep could just as easily have been put in the IRSensor class, there was also the aesthetic reason to have an SV203 Waba component that could easily be paired with the SV203 hardware component. Using this Waba component as a fundamental building block of the other two, the servo and IR components can easily wrap convenient API calls around the lower-level serial port writes and reads employed by the SV203 Waba component.

Servo Component

The PPRKServo component takes an instance of the SV203 object as an argument to its constructor. As a result, the PPRK Servo component implements easy-to-use APIs, like turnRight, turnLeft, stop, moveServo, and moveHolonomic. These APIs, in turn, stuff a byte array with the appropriate ASCII commands and call the appropriate write method on the SV203 instance.

IR Component

The IRSensor component also takes an instance of SV203 as an argument to its single constructor. As stated earlier, when communicating with the GP2D12, a response is expected from the SV203 controller board with each command sent to it requesting object location information. The IRSensor component implements methods to poll a particular IR sensor for a data reading and also includes methods getStringPortValue and getIntPortValue, which return the IR sensor readings as either an int or a string.

Utilities

Finally, WORF currently uses a simple class, Utils, to pro-

vide static functions for byte-to-ASCII conversions, and vice versa. It's meant to be a catch-all class for methods that don't really fit anywhere else.

Putting It All Together

Okay, so here you are with our PPRK hardware all assembled and functioning (see Resources for WORF diagnostic utilities and hardware troubleshooting tips), and you have a basic knowledge of how to develop for the robot. But what can you do with it?

Well, at the risk of using a cliché, you're truly limited only by your imagination, creativity, and attention span to your project.

My first idea was an attempt to make my PPRK do a Mexican Hat Dance by writing a short program using WORF. Listing 1 is the complete source code for my WORF Mexican Hat Dance application, which you can run on your own PPRK (see Figure 2).

This is a simple demonstration of how easy it is to write a small robot program for your PPRK using WORF. The WORFHatDance application (see Resources) causes the robot to alternately turn right and left at different stages of the musical ditty it plays when you tap the Push Me button displayed on your Palm (see Figure 3). The static waba.fx.Sound.tone() method tells your Waba application to play a tone at the given frequency for the given duration. The tones specified in Listing 1 play some semblance of the Mexican Hat Dance, even if it is a little off key at times.

I'm sure many reading this article are asking themselves, "What's the real value in building a PPRK and writing software for it? Could I enter my PPRK in the BattleBots competition? More important, how can I make money with it?" The real value in these types of exercises, however, is fun and education. Granted, the PPRK, as it is, probably couldn't easily be modified to do your laundry or sweep your floor, and wouldn't last two seconds in the BattleBots arena. The servo motors aren't strong and the GP2D12 IR sensors are somewhat delicate.

But I have to admit to a certain childlike fascination in writing a few lines of code capable of controlling hardware external to my Palm device. The PPRK provides a good platform for investigating possibilities in controlling and interacting with robotic hardware from a small footprint device, such as a Palm or iPaq. As far as I'm concerned, building robots and controlling them with my Java programs is just good fun!

The money will come, hopefully, from the knowledge you gain from basic projects such as this, strengthening your skills in embedded device programming, and contributing code – and lessons you've learned – to the Java development community at large.

FIGURE 2 Screenshot of WORF Hat Dance program

FIGURE 3 The PPRK performing the Mexican Hat Dance

Resources

1. *Informal notes on building and troubleshooting the PPRK:* www.trexlabs.com/pprk
2. *Official Carnegie Mellon University PPRK:* www.cs.cmu.edu/~pprk
3. *The Waba VM:* www.wabasoft.com
4. *The SuperWaba VM:* www.superwaba.org
5. *Waba Newsgroup:* news://pilot.programmer.waba
6. *Source repository for WORF, the WORF Diagnostic Utility, and the WORFHatDance program:* http://worf.sourceforge.net

Author Bio

James Caple is an independent programmer and author. He has more than eight years of industry experience, including building mobile device synchronization software and systems in Java. He is a Sun Java 2 certified programmer and developer.

jcaple@patriot.net

Listing 1

```
package com.trexlabs.worf.robot;

import com.trexlabs.worf.sv203.*;
import com.trexlabs.worf.ir.*;
import com.trexlabs.worf.servo.*;

import waba.ui.*;
import waba.fx.*;
import waba.io.*;

/**
 * An example program using WORF to make your robot dance
   while it
 * plays a catchy tune.
 *
 * @author James Caple Copyright (C) 2001, All Rights
   Reserved
 */

public class WORFHatDance extends MainWindow {

    private Button begin;
    private SV203 board;
    private PPRKServo servos;

    public WORFHatDance() {
begin = new Button("Let's Dance!");
begin.setRect(50, 85, 59, 20);
add(begin);

// WORF Setup
board = new SV203();
servos = new PPRKServo(board);
    }

    /**
     * Draw about Strings.
     * @param Graphics
     * @return void
     */
    public void onPaint( Graphics g ) {
g.setColor(0, 0, 0);
g.drawText("WORF Hat Dance Example", 0, 0);
g.drawText("Copyright (C) 2001, James Caple", 0, 10);
g.drawText("http://worf.sourceforge.net", 0, 20);
    }


    public void onEvent(Event evt) {
 if (evt.type == ControlEvent.PRESSED && evt.target ==
begin) {
     getYourGrooveOn();
 }
    }

    private void getYourGrooveOn() {
 // Swing to your right
 servos.turnRight((byte)3);

 Sound.tone(3000, 500);
 Sound.tone(2900, 200);
 Sound.tone(3000, 200);
 Sound.tone(2600, 200);
 Sound.tone(2500, 200);
 Sound.tone(2600, 200);
 Sound.tone(2100, 200);
 Sound.tone(2000, 200);
 Sound.tone(2100, 200);
 Sound.tone(1500, 500);

 // Swing to your left
 servos.turnLeft((byte)3);

 Sound.tone(1300, 200);
 Sound.tone(1400, 200);
 Sound.tone(1500, 200);
 Sound.tone(1700, 200);
 Sound.tone(1900, 200);
 Sound.tone(2000, 200);
 Sound.tone(2300, 200);
 Sound.tone(2500, 200);
 Sound.tone(2700, 200);
 Sound.tone(2300, 500);

 // Swing to your right
 servos.turnRight((byte)3);

 Sound.tone(2700, 200);
 Sound.tone(2600, 200);
 Sound.tone(2700, 200);
 Sound.tone(2300, 200);
 Sound.tone(2200, 200);
 Sound.tone(2300, 200);
 Sound.tone(1900, 200);
 Sound.tone(1800, 200);
 Sound.tone(1900, 200);
 Sound.tone(1500, 500);

 // Swing to your right
 servos.turnLeft((byte)3);

 Sound.tone(3000, 200);
 Sound.tone(3000, 200);
 Sound.tone(3000, 200);
 Sound.tone(3300, 200);
 Sound.tone(3000, 200);
 Sound.tone(2700, 200);
 Sound.tone(2500, 200);
 Sound.tone(2200, 200);
 Sound.tone(2000, 500);

 Sound.tone(4000, 100);

 // Cha cha cha!
 servos.stop();
 servos.turnRight((byte)3);
 servos.turnLeft((byte)3);
 servos.turnRight((byte)3);
 servos.stop();
    }

}
```

Wireless Java: Developing with Java 2 Micro Edition

REVIEWED BY JASON BRIGGS
jasonbriggs@sys-con.com

I don't usually buy technical books. Most of the time I manage to find what I'm looking for after a bit of hunting on the Web, virtual elbow-grease, so to speak. However, occasionally a tree-killing madness does come over me and Amazon gets the chance to clear the electronic cobwebs out of my bank account.

So I'm sure much to Amazon's complete disgust – as it was almost time to satisfy that craving to handle dried wood mulch – *Wireless Java: Developing with Java 2, Micro Edition*, by Jonathan Knudsen, arrived on my desk from Apress.

This 226-page book is aimed at intermediate to advanced developers – by which I think they mean experienced Java developers who are new to the Micro Edition platform. It has 12 chapters that cover the basics of developing for MIDP up to more advanced subjects, such as parsing XML and cryptography.

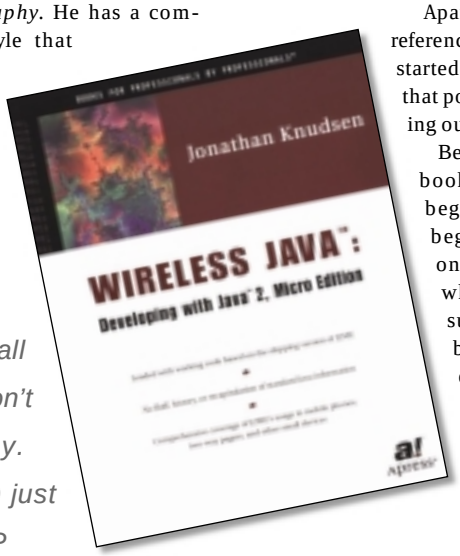
Jonathan Knudsen has penned a number of other books, including *The Unofficial Guide to Lego Mindstorms Robots*, *Learning Java*, *Java 2D Graphics*, and *Java Cryptography*. He has a comfortable writing style that makes for an easy read. Here's an excerpt from Chapter 9 – "Programming a Game Interface":

How exactly does repaint() work? When you call repaint(), paint() won't be called right away. The call to repaint() just signals to the MIDP implementation that you want the screen to be painted. Some time later, the implementation services the repaint request, which results in an actual call to the paint() method of the Canvas. The MIDP implementation may even combine several repaint requests, particularly if their repaint regions overlap.

There are some useful tips hidden in the text for MIDP beginners:

Bandwidth is expensive in terms of both time and money. Today's wireless networks are relatively slow, so less data passing through the air means less waiting time for your users. Also, wireless service tends to be pricey, so less data passing through the air means smaller bills for your users.

If I have any criticism, it's the obligatory inclusion of an API summary at the end of the book – which seems to be de rigueur for any technical tome but usually not necessary, in my opinion (however, it's nowhere near as large as some API references I've seen included, so kudos to Jonathan and Apress for keeping it to a minimum).



Apart from this, it's a useful reference if you're just getting started with MIDP and, from that point of view, worth dishing out the $34.95.

Bear in mind that the book is aimed at J2ME beginners, not Java beginners, so if you're one of the few people who are new to both, I suggest investing in a beginners Java reference as well (and no, I'm not aiming to receive Amazon's "Literary Pimp of the Year" award for 2001).

info

Wireless Java: Developing with Java 2, Micro Edition
by Jonathan B. Knudsen
Published by Apress

Chapter 1:	An introduction to configurations, profiles and MIDP vendors
Chapter 2:	Getting started building MIDlets (compilation, preverifying, etc.)
Chapter 3:	The MIDlet life cycle, packaging, and descriptors
Chapter 4:	What's included and what's missing from the API set
Chapter 5:	Creating a user interface
Chapter 6:	Using Lists and Forms
Chapter 7:	Managing record stores and performing queries
Chapter 8:	The Connection framework
Chapter 9:	Low-level graphics used for programming a game interface
Chapter 10:	Performance tuning
Chapter 11:	Parsing XML
Chapter 12:	Protecting network data (in other words, cryptography)

Pushing the Limits

The top 10 MIDP limitations

WRITTEN BY
GLEN CORDREY

Developers new to the Mobile Information Device Profile (MIDP) are often surprised by some of the limitations and their consequences, because they seem to be omissions of obviously needed functionality.

In this article I discuss my personal favorites – the rationale (to the degree I can discern it) for them; the impact of their omission; workarounds and alternatives; and the efforts that exist to address these limitations. These efforts often involve Java Specification Requests (JSRs), which I'll reference when applicable.

Some of the limitations are in fact limitations of the Connected Limited Device Configuration (CLDC). As I write this article the MIDP is the only finalized profile under the CLDC, so I'll discuss CLDC limitations in the context of the MIDP.

URLs for all referenced JSRs and software products are provided in the Resources section at the end of the article.

1 No Floating Point Support

The CLDC doesn't require floating-point support – neither the float and double primitives, nor the Float and Double classes, are present in the CLDC. The CLDC specification states this is because "...the majority of CLDC target devices don't have hardware floating-point support, and...the cost of supporting floating-point in software was considered too high." However, there are obviously many valuable applications that you simply can't implement without floating-point calculations.

Floating-point support is being considered for inclusion in the next version of the CLDC specification (JSR139), but for now if your application requires floating-point calculations you have the following options:

- The targeted platforms' OEMs might provide APIs that support floating-point.
- You can use publicly available software for floating-point emulation.
- You can implement your own floating-point emulation software.

The CLDC doesn't preclude OEMs from supplying their own floating-point APIs. If OEM floating-point APIs are available for your targeted platforms, you should use them because most likely they'll execute native code, giving better performance and accuracy than floating-point emulation software.

Without access to OEM APIs you can either use publicly available floating-point emulation software such as MathFP (see Resources) or create your own. If you implement your own you must address issues such as the accumulation of truncation and rounding errors, and rigorously test your software.

Regardless of whether you use OEM-supplied APIs or floating-point emulation, consider creating your own classes to wrap all calls to the solution available to you. This will make it easier to debug your applications and to port them to other platforms, because your changes will be centralized within the wrapper classes. If you model your wrappers as closely as possible after the J2SE Float, Double, and Math classes, you may also reduce your effort in transitioning to the next version of the CLDC. It's reasonable to assume that if it does support floating-point it will provide Float, Double, and Math classes that are subsets of their J2SE counterparts.

2–4 No JNI, No Telephony APIs, No Sound APIs

I group these together because the absence of telephony and sound APIs are good examples of the impact of the lack of support for the Java Native Interface, which the CLDC specification says is omitted because:

1. JNI would allow applications to compromise the CLDC security model.
2. A full JNI implementation would be "too expensive given the strict memory constraints of CLDC…devices."

The lack of JNI eliminates the possibility of third-party or custom workarounds for providing missing functionality; the only way MIDP applications can access native functionality is if the OEMs provide APIs for it. For example, on an MIDP-enabled cell phone, a phonebook MIDlet could only initiate a phone call if the cell phone's manufacturer provides Java APIs to support it. Likewise, if a MIDlet game wishes to use snazzy sound effects that are available on the phone via native

libraries, it can do so only if the phone's OEM provides Java APIs for sound.

It doesn't appear that any JSR is considering the incorporation of JNI support into CLDC or MIDP. Optional J2ME telephony APIs are being defined under JSR120, and a base sound API is listed for investigation under JSR118 (the next generation of the MIDP specification, aka MIDP_NG).

5 No RMI

RMI is not supported due to the lack of support for reflection. The CLDC doesn't support reflection because the ability to dynamically examine an object could lead to the subversion of features of that object, thereby compromising security. Reflection is used in serialization to discover class attributes that need to be serialized, and an object must be serialized to be transported via RMI.

Without RMI an MIDP client can't directly interface with existing or new servers that provide only RMI interfaces. kSOAP provides an MIDP-implementation of SOAP that can be used as an alternative to an RMI architecture (kSOAP includes an interface and classes for implementing a partial replacement for serialization). Some applications may also consider Softwired's iBus, which implements an MIDP JMS solution.

The inclusion of reflection and serialization doesn't appear to be under consideration in any JSRs. A proposed final draft of a J2ME RMI profile has been developed under JSR66.

6 The Only Required Protocol Is HTTP

The CLDC provides a connection framework but doesn't mandate the support of any network protocols. The MIDP mandates the support of a subset of HTTP 1.1; OEMs can provide support for other protocols, but are not required to. The CLDC and MIDP both state that a larger set of protocols is not mandated because of memory limitations and environmental conditions (e.g., whether a cell phone uses a circuit- or packet-switched network) specific to the various devices.

This means your MIDlets can't use datagrams (UDP), serial ports, HTTPS, sockets, or other protocols unless the OEMs of the platforms you deploy to support them. Support for these protocols would provide some of the following benefits:

- Since UDP is more efficient than HTTP, datagrams would provide a transport mechanism that could use the limited bandwidth and speed of wireless connections more cheaply than HTTP (at the cost of UDP's lack of reliable delivery).
- Serial port support could provide a simple, efficient connection for hot-synching MIDlet's data stores, such as address books, with their PC-based counterparts.
- Support for HTTPS would promote the security of sensitive data transactions, which are necessary for mobile commerce.
- Support for sockets would allow the development of client/server applications with significantly better speed than HTTP provides.

Sun's Web page for JSR118 states that support for sockets, datagrams, and HTTPS are "functional areas to be investigated"; no mention is made of serial port connectivity. Until and unless the next version of the spec addresses the issue, we're dependent upon the good graces of the OEMs to support the desired protocols.

As a partial alternative to HTTPS, you can encrypt your data using J2ME crypto APIs available from Bouncy Castle.

7 No Shared Libraries

MIDlets are deployed in a MIDlet suite, which is a JAD file plus a JAR file containing one or more MIDlets. MIDlets in the same suite can share classes within the suite, but the MIDP specification makes no provision for sharing classes among MIDlet suites unless those libraries are provided by the OEM and reside on the platform. This limitation stems from concerns including security and versioning issues.

This limitation can cause the size of your deployed applications to be larger than they could be with custom shared libraries, which is a significant concern because of the limited storage capacities of Mobile Information Devices (MIDs). If you think the limitation can be avoided simply by packaging all your MIDlets into a single suite, consider the situation in which you wish to sell separate suites that use common functionality provided by floating-point emulation code. Not only will each suite incur the size cost of the floating-point class files, but each suite needs to be repackaged and redeployed whenever you wish to incorporate floating-point bug fixes.

None of the JSR descriptions at Sun's Web site mention efforts to address this issue. It's unclear whether OEMs could provide their own solutions without breaking conformance to the MIDP specification.

8 No Custom Display Widgets for the High-Level UI

You can't create custom visual components (aka widgets) that can be used with the MIDP's high-level UI APIs. Class javax.microedition.lcdui.Form is the only screen that can contain disparate widgets, and it only allows components that extend class javax.microedition.lcdui.Item. However, Item's paint method has protected access, so a custom subclass of Item can't display itself to the screen (if you try to get around this limitation by placing your subclass in package javax.microedition.lcdui, there's still no guarantee that the widget will display properly).

This constraint results from a focus of the MIDP specification upon interoperability, a write-once, run-anywhere goal. By preventing the inclusion of custom widgets, the high-level UI eliminates the possibility of a custom widget displaying satisfactorily on one MIDP device but not another. However, it also limits the ability of developers to provide visually distinctive displays, and to explore new ways of interacting with the user in the small real estate of a wireless device's screen.

JSR118 is investigating UI issues that may include support for the creation of custom widgets. For the present, both the Motorola Lightweight Windowing Toolkit (LWT) and Nextel's Open Source Windowing Toolkit (OWT) provide UI frameworks that support the creation of custom widgets.

"By preventing the inclusion of custom widgets, the high-level UI eliminates the possibility of a custom widget displaying satisfactorily on one MIDP device but not another"

9 No Control Over High-Level UI Layout

The high-level UI API mentioned earlier also doesn't provide control over the layout of visual components. The layout is vertical, with focusable widgets always added to a new line. You also can't specify the horizontal (left, centered, right) placement of items. For example, you can't place three text fields horizontally adjacent, separated by hyphens, to create a social security field, nor can you center a text label above a widget.

The rationale for this is the same as for preventing the creation of custom widgets – to promote interoperability. Similarly, this limitation may be addressed by JSR118, and both Motorola's LWT and Nextel's OWT provide alternative approaches.

10 No Shared Data

The MIDP's persistent data mechanism, the Record Management System, doesn't allow MIDlet suites to share data. Any MIDlet in a suite can access the persistent data stored by other MIDlets in the same suite, but not the persistent data stored by MIDlets in any other suite.

This is a security precaution to prevent malicious applications from accessing sensitive data created by other applications. However, it also precludes the sharing of data that might be reasonably shared, and requires that the entry of that data be duplicated, which may be particularly annoying to the user given the limited keyboards of many MIDs. Consider the deployment on a cell phone with a numeric keypad only (letters are entered by multiple presses of a number) of multiple m-commerce MIDlet suites that require credit card information, which the user will have to enter for each suite.

Many security issues are under consideration in JSR118, although it's unclear whether there will be any changes in the security policy for data sharing.

Last Thoughts

As with many nascent technologies, the MIDP has limitations that you may need to address to produce your applications. In some cases you may be able to find or develop custom software to provide the functionality you need. In other cases the functionality may only be available if the OEMs of your targeted platforms provide it. In either case the solutions may compromise the portability of your applications. You can reduce the impact on portability, or at least reduce the effort to port to different platforms, by providing wrappers for code that's dependent upon custom or OEM solutions, and by keeping abreast of, and conforming to, candidate standards under consideration as JSRs.

Resources

1. *Bouncy Castle:* www.bouncycastle.org
2. *JSR 66 – J2ME RMI Profile:* www.jcp.org/jsr/detail/66.jsp
3. *JSR 118 – MIDP Next Generation:* www.jcp.org/jsr/detail/118.jsp
4. *JSR 120 – Wireless Telephony Communication APIs:* www.jcp.org/jsr/detail/120.jsp
5. *JSR 139 – CLDC, Next Generation:* www.jcp.org/jsr/detail/139.jsp
6. *kSOAP:* http://ksoap.enhydra.org/index.html
7. *Motorola Lightweight Toolkit:* http://developers.motorola.com/developers/wireless/tools
8. *Nextel Open Windowing Toolkit:* http://nextel.sourceforge.net
9. *OTA User Initiated Provisioning Recommend Practice:* http://java.sun.com/products/midp/OTAProvisioning-1.0.pdf
10. *Softwired/iBus:* www.softwired-inc.com/products/mobile/mobile.html

Author Bio

Glen Cordrey is a developer of i2ME and i2EE solutions for DigitalFocus in Herndon, Virginia. He is the architect of Nextel's i2ME toolkits, the principal author of Nextel's i2ME Developer's Guide, and a member of the MIDP_NG Expert Group.

glen@oojava.com

WHITEboard SDK, Bluetooth Edition

by Zucotto Wireless Inc.

REVIEWED BY JASON BRIGGS

jasonbriggs@sys-con.com

info:

Zucotto Wireless Inc.
4225 Executive Square,
Suite 400
La Jolla, CA 92037
Web: www.zucotto.com
Phone: 877-ZUCOTTO
Fax: 858 777-1323
E-mail: sales@zucotto.com

A contagious disease that results in oddly colored teeth? A South Seas pirate with a penchant for eating toxic sea food? Or perhaps a superhero with really unhelpful superpowers?

If you answered yes to any of these questions, chances are you've had your Java blinkers on and haven't been paying attention to the rest of the industry. Bluetooth, despite its interesting name (Bluetooth being another name for the Danish king Harald Blåtand, who unified his kingdoms in Norway and Denmark), is a specification intended to provide low-cost, wireless connections between all sorts of devices separated by about 10 meters. A number of high-profile companies are involved in the specification including 3Com, Ericsson, IBM, Agere, Intel, Motorola, Nokia, Microsoft, and Toshiba. The members of the Bluetooth Special Interest Group (SIG) are, at the time of this writing, the nine aforementioned promoter companies, 187 associate companies, and 1,305 early adopter companies of version 1.2.

Here's where it gets interesting. If you've been reading the technology news recently, you may have noticed that the general manager of Intel's Communications Group rather vocally announced that Bluetooth had "already lost the battle to become the wireless network standard" to IEEE 802.11. 802.11b is a full LAN connectivity solution utilizing the same frequency range as Bluetooth, but using different types of spread spectrum technology that's designed to provide network services at Ethernet data rates (whereas Bluetooth is limited to 1Mbps).

So, depending upon whom you decide to believe, either 802.11 has eliminated Bluetooth's chances for world domination, or the two technologies have enough differences that they will eventually find their own place in separate markets.

FIGURE 1 Hardware component

Kit

Let's put aside speculation as to whether Bluetooth or 802.11 will be the choice of the new generation, and imagine a device with enough broadcasting power to beam messages to Mars and change channels on the TV...from the other side of the moon, all without using more than a AAA battery.

Zucotto's Bluetooth Edition of its WHITEboard SDK, with XJB Bluetooth Host Protocol Stack, does none of those things. However, Zucotto's Bluetooth Stack is written entirely in Java; it implements the host portion of the Bluetooth specification and supports the serial port, generic access, and service discovery application profiles. Which doesn't really compare to being able to signal the mothership from across the solar system, but if you're developing Bluetooth-based applications, you might not be too bothered by this.

Exactly what is the Bluetooth WHITEboard SDK? Figure 1 shows the hardware component of the package – two small Class 2 Bluetooth communications boards with radio, baseband, and antenna (and with a power supply and serial cable for each one). According to the Zucotto documentation, each board complies with version 1.1 of the Bluetooth spec and supports one active connection at a time. Also supplied is a Developer's Guide, a Reference Guide, and a CD with the WHITEboard software, which is essentially a Device Editor (used to create simulated device skins), a Device Emulator (with a Motorola phone "skin"), a PNG Painter, and finally the WHITEboard Integrated Development Environment (based upon the NetBeans IDE). Figure 2 shows the IDE in action.

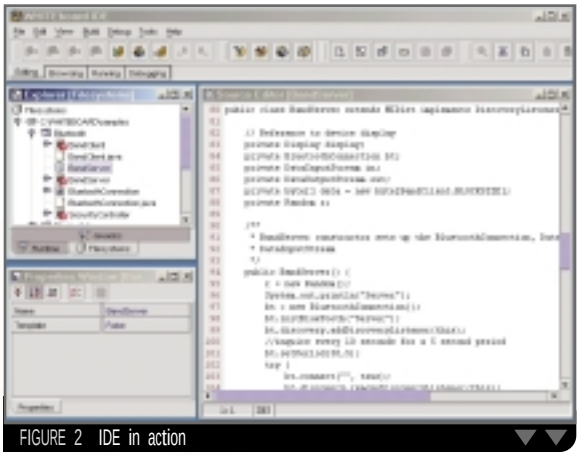


FIGURE 2 IDE in action

Demo

Luckily for your average "not-enough-hours-in-the-day ***JDJ*** editor," the software kit also comes with various demonstration MIDlets. Which means it's not too painful to plug the bits and pieces together and get something up and running. Figures 3–5 show screenshots of the ChatServer and ChatClient

MIDlets in action. In practical terms, copies of the Device Emulator are running on two laptops, with the communication boards plugged into the serial port of each machine.

Figure 3 shows the ChatClient MIDlet running on one laptop as it connects to the server MIDlet. Figure 4 shows the server after it has received a message and has replied, and Figure 5 shows the receipt of that message.

Problems and Successes

The only problems I experienced with the dev kit were a difficulty running the IDE (after installing JDK 1.4, go figure) and a temporary glitch with one of the comms boards (which fixed itself after being powered off, and never reoccurred). Apart from that, the demo programs started up first time without any hitches, and without any need to diagnose obscure faults in equipment that you don't really understand the intricacies of – sighs of relief all around! The code for the sample applications is included (and fairly well commented), which is exceedingly useful if you're a beginner to the world of funnily colored teeth.

Conclusion

The SDK comes with a fairly hefty price tag, $2,995. Not for your average bedroom coder on a budget. But if you have a product you're going to aim at the Bluetooth market, then a kit like this – to support your development efforts – is probably essential. An educational version of the kit is available at a lower price.

The experience for me, coming from a background that admittedly hasn't generally included Bluetooth devices, was a positive one.

For an overview of the kit visit www.zucotto.com/whiteboard/blue_index.html. For more details on the Developer's Guide visit www.zucotto.com/whiteboard/downloads/WB_SDK_2-0BT.pdf, or simply contact sales@zucotto.com.

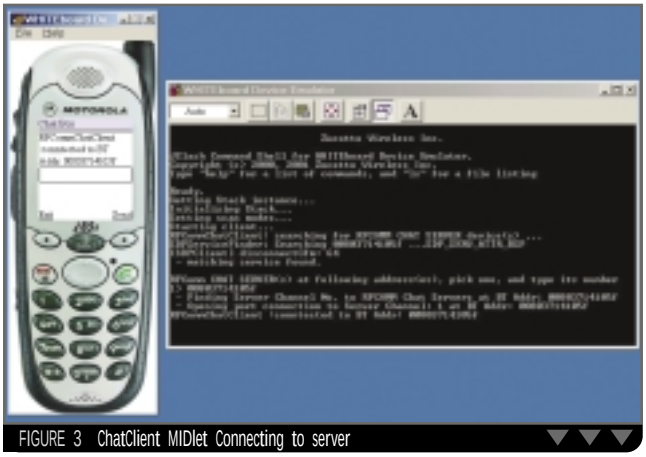


FIGURE 3 ChatClient MIDlet Connecting to server

FIGURE 4 The server replies to a message

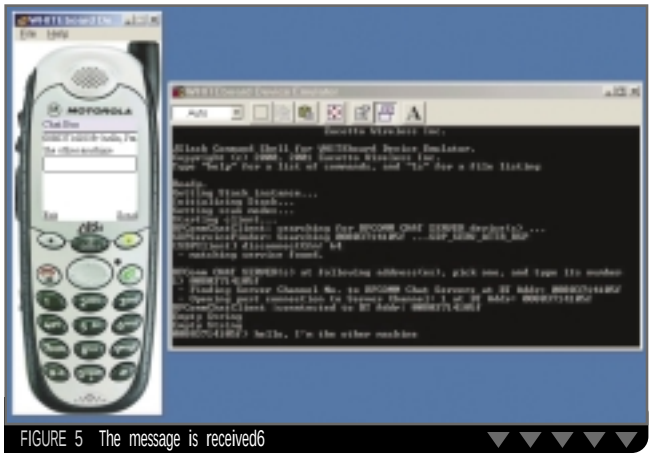


FIGURE 5 The message is received6

Web Services Edge West & XMLEdge 2001

Show Wrap-up

WRITTEN BY
JEREMY GEELAN & SEAN RHODY

Steve Benfield CTO, SilverStream Software

Web Services Edge West and XMLEdge 2001 opened for business October 22 with a series of sessions and tutorials on the hottest topics in the industry. Five tracks were maintained throughout the conference.

The conference began on Tuesday, October 23, with a keynote presentation by Dr. Charles Goldfarb, "the father of XML technology." Dr. Goldfarb's presentation was an intriguing exploration of the dichotomy that exists between application developers and document developers as they approach the concept of XML from two completely different perspectives. XML "is a precision tool...being misused by people who think it's a shotgun," he asserted.

Santa Clara Convention Center

He expressed this dichotomy with a number of examples that led to the development of a concept he calls HARP – Human, Abstraction, Rendering, and Presentation – and made a strong case for extracting and encapsulating data in XML format.

Goldfarb took the audience through the entire history of data systems in order to reinforce his point about how revolutionary XML actually is. But the interest level of delegates increased markedly as he asserted that many so-called XML "gurus" in fact misuse or manipulate some of the basic terminology associated with XML in order to sell XML-based products of one variety or another.

Goldfarb's vision for XML is pure and based on freedom, not proprietary solutions. "It frees developers from having to be locked into a single processing paradigm," he explained.

"XML is based on free and open standards. So every time I see that phrase MSXML, I get a little nervous," he quipped, in a reference to the initials of a parser that bear a wholly coincidental similarity to those of a certain company in Seattle with a well-known reputation for keeping things proprietary.

Goldfarb also told the audience at his keynote that he expected to see large companies like SAP adopting XML wholesale "very soon" since XML removes the need for monolithic data systems. XML, he observed in rounding off his presentation, "is, like SGL before it, designed as a standard for coping with the unstandardized – this is the most important aspect of the XML revolution."

The second keynote of the day was given by Oracle Corporation's vice president of product development for Oracle9*i*, Thomas Kurian.

Kurian explained to an attentive audience of developers and *i*-technology professionals how greatly they would benefit "if the J2EE and Web services programming models were rendered analogous so that you didn't have to learn a new programming model."

Kurian outlined Oracle's vision for a world of Java-driven Web services and presented in detail how the architecture of, for example, Oracle's own 9*i*AS application server

CEO Panel participants: (back row) Annrai O'Toole, chairman, Cape Clear; Sean Rhody, panel moderator and editor-in-chief, *Web Services Journal*; Dirk Slama, CEO, Shinka Technologies; Greg O'Connor, president, Sonic Software; (front row) Ali Kutay, former CEO of WebLogic and President and CEO of AltoWeb; Eileen Richardson, president and CEO, Infravio; and Barry Morris, CEO, Iona Technologies.

with its unified model lends itself to building, deploying, and managing Web services.

"There's a lot of hype going round," he observed, "about Web services as nirvana." Oracle's view, according to Kurian, was more grounded, and based on a solid belief

"As the Java print media market consolidates, *Java Developer's Journal* will most likely be the only long-term player..."

– Scott Clark, Editor of *Java Boutique*/MecklerMedia

Now that our competitor is no longer in print...

BPA Web site: November 15, 2001

Title/Topic/Home Page	Current Average Qualified Circulation	Prior History Statements
Java Developer's Journal Topic: COMPUTERS (Consumer) http://www.sys-con.com	97,325 June-2001 View Report	View History
Java Developer's Journal (E-mail Newsletter) Topic: COMPUTERS (Business)	121,462 November-2000 View Report	View History
Java Report Topic: COMPUTERS (Consumer) http://www.javareport.com	36,223 June-2001 View Report	View History
JavaPro Topic: COMPUTERS (Consumer) http://www.fawcette.com/jp	35,715 December-2000 View Report	View History

Media Owners | Media Buyers | Products | Reports Library | BPA University | Resources
About Us | Site Map | Contact Us | Home

BPA © Copyright 2000, 1999, 1998 BPA International. All rights reserved. 1-212-779-3200 • info@bpai.com

designed and hosted by gcnpublishing

JAVA DEVELOPER'S JOURNAL

Rate Base	Actual Circulation
60,000	97,325

JDJ has always underpromised and overdelivered. In June 2001 we exceeded our rate base by 37,000 copies!

...*JDJ* is here to serve you as your exclusive media partner for many years to come!

Historical Facts: June 2000

	Rate Base	Actual Circulation
JDJ	60,000	67,618
Java Report	30,000	23,777
JavaPro	30,000	24,179

carmen@sys-con.com

miles@sys-con.com

For accurate information on the circulation of Java publications, and the hands-down leadership position of ***Java Developer's Journal*** and how it relates to your advertisement cost per thousand copies (CPM), please contact your account manager today.

"*Java Developer's Journal* gives us the highest number of leads on any advertising we do..."
Larry Humphries, President of Sitraka
LIVE on SYS-CON Radio

SYS-CON MEDIA
www.JavaDevelopersJournal.com
(201) 802-3020

that open standards and interoperability can take e-business to the "next level," as he termed it. By that he meant the next level of operational efficiency, and therefore of profitability.

He explained that, in his view, UDDI, SOAP, and WSDL alone will not be enough, and how the true success of more complex Web services will require ebXML, RosettaNet, and the Java JAX standards.

"Web services is a great way to allow applications to be built. It's a set of technologies to let one app talk to another using open standards." Kurian continued, "Oracle believes in open standards and interoperability, including .NET, which is why our goal is to unify the J2EE and Web services models."

Ajit Sagar interviews Ben Brauer, product marketing manager, H-P (top), and Dirk Slama, CEO, Shinka Technologies (bottom) for SYS-CON Radio

On Wednesday, October 24, the exhibition floor opened and a large show crowd explored the wares of more than 40 vendors, including SilverStream, Borland, and IONA. Several vendors unveiled new products, including Systinet (formerly IDOOX) and Borland.

In Wednesday morning's opening keynote, Steve Benfield, CTO of SilverStream Software, addressed the overall question of Web services and emerging application developer techniques.

Well known in the industry for his view that "Web services will change the world," Benfield nonetheless took care to explain that in many ways "there is nothing magic about Web services." Their impending success is based on changes in the overall business environment as much as on any quantum leap in technology. Benfield asked, "What is all the fuss about?"

The answer, he asserted, is that while developers have been doing XML over HTTP for awhile, Web services is composed of far more than that. A Web service, he said, can best be defined as, "A readily shareable business function," and that is what makes the move toward Web services revolutionary, and constitutes a business opportunity for software vendors and business alike since it will help customers extend their business to the Web.

Exhibitors demonstrate the latest technologies

Flexibility is another advantage. "A few years ago, when you built an application," he told the many developers in the audience, "you knew who the users were going to be and what they'd use, but today you don't. Web services creation allows you to architect for reuse, which can be done without regard to the actual user."

The most important difference between Web services and previous attempts, like CORBA, to ensure interoperability between businesses, is the fact that every major vendor is working to support it: "With Microsoft and IBM both behind it 100%," Benfield said, "Web services is uniquely positioned to succeed – plus, it works using existing Internet and XML standards."

Steve explained that "mere mortals" won't be able to build successful Web services without help, and that a new product set will sweep through the industry, what he

We don't mind if they say we publish "geek" magazines, as long as we're known as...

The World's Leading *i*-Technology Publisher

June 2001
136,000 copies in print

June 2001 Preview
250,000 copies in print

June 2001
82,000 copies in print

June 2001
72,000 copies in print

June 2001
48,000 copies in print

Premiering December 2001
20,000 copies in print

Premiering January 2002
20,000 copies in print

Premiering February 2002
150,000 copies in print

and Producer of the World's Leading *i*-Technology Developer Conferences

JDJ EDGE conference&expo | XML EDGE conference&expo | web services EDGE conference&expo | ColdFusion EDGE fasttrack | wireless EDGE conference&expo

Inc. 500 1999 WINNER

SYS-CON™ MEDIA SYS-CON™ EVENTS

www.SYS-CON.com

America's fastest-growing privately owned publishing company

Inc. 500 2001 WINNER

termed "Integrated Services Environments" – in other words, an entire framework for building, deploying, and managing Web services.

"If you want to spend your time building applications versus understanding the semantics of Web Services Description Language and Simple Open Access Protocol, then you need an ISE," he said, emphasizing the ease of creation, the flexibility of the interface, and the strong cross-platform implementation of Web services as key to its success. His overall point was that the ease of creation will be driven not so much by the underlying simplicity of the paradigm, but more by the availability of powerful tools and ISEs for abstracting the details to a simple interface for creation.

Wednesday afternoon the West Coast Web Services panel discussion took place, another of SYS-CON's energetic CEO panels, a high-octane formula already much enjoyed at the Web Services Edge East event in New York in September, and now equally successful at Web Services Edge West.

> "There has never been anything like Web services before ...it's a complete and utter revolution"

Moderated by ***Web Services Journal*** editor-in-chief Sean Rhody, the panel was made up of Barry Morris, CEO of IONA Technologies; Greg O'Connor, president of Sonic Software; Dirk Slama, CEO of Shinka Technologies; Eileen Richardson, CEO of Infravio; Ali Kutay, CEO of AltoWeb; and Annrai O'Toole, executive chairman of Cape Clear Software.

Rhody opened the discussion with the same question that opened the East coast panel: "What is a Web service?" and the panel soon moved on to discuss the role of Web services in the industry.

Discussion was animated from the first, fueled by the panel's collective ability to both enthuse and dispute about the subject at hand: the role of Web services.

Steve Benfield, CTO of SilverStream Software delivering his keynote speech

"You're going to see the power structures in our industry change," asserted Morris, a viewpoint endorsed by O'Toole, who was also adamant that Web services represents a "fundamentally different architecture" than anything we have seen to date in the world of computing.

The point that created the most controversy was the question of whether Web services represented an evolution or a revolution. Slama's endorsement of the "evolutionary" view was strongly contested by Morris, who time and again stressed to his fellow panelists and to the entire audience of developers and *i*-technology professionals that there has never been anything like Web services before...it's a complete and utter revolution.

Morris spoke of Web services as a "flattening" of information infrastructures, such that "more people are able to get at your enterprise value" than ever before. Exposing that enterprise value as Web services is going to be the key to the next phase

of e-business, he explained. Just as people writing in Word or doing a presentation in PowerPoint are in effect programming, they are empowered by technology to do something previously doable only by programmers. So, continued Morris, Web services is poised to enable a mass adoption of distributed computing techniques previously available only to software engineers.

AltoWeb's Kutay took issue with this point of view. He couldn't see how Web services would ever supersede the need for, say, J2EE since at the enterprise level there were issues of transaction management and security that Web services simply didn't address adequately. In Kutay's view, "Web services are complementary to what J2EE provides," but they don't, he felt, represent the "revolution" that Morris was arguing they do.

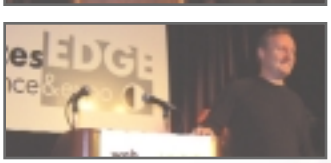
At one point it was suggested that a Web service should use UDDI and WSDL over HTTP, but this caused a fairly significant uproar among the panel

CEO panel discussion engrosses audience

members. Morris dismissed UDDI and WSDL at one point, insisting that focusing on the "plumbing" of Web services was missing the point of Web services entirely: Web services is a paradigm that enables corporate developers to finally unite their applications.

For me, probably the most interesting aspect of the discussion was the completely divergent opinions on the target audience for Web services. O'Toole and several other members clearly felt that Web services should be ubiquitous and purposefully easy to use for everyone – basically the next HTML. Morris and O'Connor disagreed completely, consigning Web services to the realm of professional developers and programmers.

On Thursday, October 25, O'Toole gave a keynote in which he set out to prove to delegates that Web services "transforms the whole economics and technology of application integration." O'Toole outlined to the software engineers, *i*-technology professionals, and senior IS managers in the audience how Web services is encouraging a welcome shift in the competitive strategies being used in the computer industry.



Annrai O'Toole, executive chairman of Cape Clear, presenting his keynote

"Throughout the past 20 years, the industry has been divided by a series of technological hardware and software platform battles," he explained. "Now, with Web services, the industry has for the first time agreed on ubiquitous interoperability standards, thanks to industry standards such as XML, SOAP, WSDL, and UDDI, offering a solution to the bickering and posturing that has created 20 years of IT incompatibility."

"The arrival of widely accepted standards such as XML and SOAP provides a common base platform," he continued, "that supersedes arguments over operating systems, languages, tools, and applications. It prepares the ground for a new era of cooperation – a third way, if you will."

For O'Toole the joy of Web services is that there's nothing actually new in them, "but they're repackaged so that the mainstream developer is reenfranchised," he explained,

"putting power back in the hands of the type of developer who would use, say, Visual Basic rather than J2EE."

Known for two decades in the industry for his forthright views, O'Toole was not afraid to expand on this view of how, with the move toward a Web services paradigm, developers and users are back in charge, instead of vendors.

The business impact of Web services on application integration would follow the 80/20 rule, O'Toole made clear. "Eighty percent of the integration that end users will want to do can be delivered by Web services, while 20% will continue to be high-end integration solutions that can't be supplied by Web services." That 80% figure, he said, means that "many businesses that don't do integration at all today, the small and medium enterprises, are soon going to find that with Web services integration is possible – for under $100,000 rather than the $500,000 associated with high-end integration solutions, which often need an additional $1–2 million of consultancy fees on top of that."

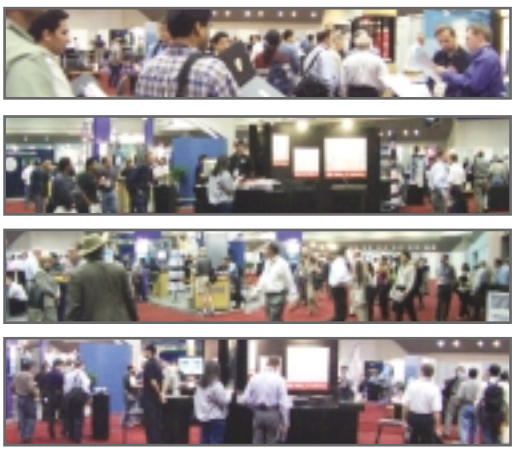


Exhibit booths drew well-informed attendees

O'Toole closed his keynote address by giving a demo of a simple Web service, including publishing it into UDDI. He successfully created it in seven minutes instead of the week that he claimed would have been necessary before the introduction of the open standards that are at the heart of the new Web services paradigm

Later that day, software engineers and business applications developers enjoyed the benefit of a keynote speaker's 18 years of industry experience in building software tools when they heard Dave Chappell, Sonic Software's VP and chief technology evangelist, deliver the afternoon keynote.

His keynote, "Web Services Meets Reliability," was well received by the audience, whose own industry experience enabled them to easily follow the technical details of Chappell's contention that messaging in general, and asynchronous messaging in particular, is critically important.

Chappell opened with a strong assertion: "In distributed application communications among enterprises and across business entities, the use of SOAP over vanilla HTTP just doesn't cut it. Parties are often unreachable." This gap, Chappell predicted, will be filled in the future by more powerful JAXM messaging providers based on protocols that provide a simple, robust way of addressing those issues – such as Java Message Service (JMS).

After explaining how JAXM – the Java API for XML Messaging – can be used for constructing and deconstructing SOAP messages intended to be sent over the wire, Chappell focused on XML messaging and talked the audience through a generic Web services usage model. He actually gave a live demo showing Apache SOAP, Tomcat, and SonicXQ all working together in a Web services interaction, then demonstrated the reliability of the system by shutting down parts of the system and bringing them back up.

He then discussed Web services from several viewpoints, leveraging the views of industry experts to prove the need for Web services.

By the end of the keynote the audience was left in no doubt that bringing reliability to the Web services infrastructure was an important goal and that messaging will play a key role since, as Chappell put it, "interfaces will always need data transformation and end-to-end guaranteed delivery, along with all the other enterprise needs such as scalability and security." And XML messaging – whether through SOAP or through more rigid sets of rules such as ebXML, JAXM, or JMS – is, within a Web services architecture, the answer.

The Week that Was

Web services goes mainstream

WRITTEN BY
ROBERT MCGARVEY

If you weren't there, a couple of years from now you'll misremember and say you were. That's how big – how momentous – Web Services Edge 2001 West and XMLEdge were.

Call that late October conference in Santa Clara the Woodstock of Web services because this is the one everyone wishes they has attended. The reason is simple: Web services has gone mainstream and suddenly, a concept that even proponents were admitting as recently as six months ago was spacey verging on vaporous, is now emerging as the next must-have by enterprise IT groups. Bottom line: Web services got sexy and it all happened in late October.

Hold on, because lots more happened in that same week. The Redmond Goliath weighed in, for instance, with an October 23 announcement laying out its view of XML architecture for Web services – at a sprawling developers conference in Los Angeles, where chairman Bill Gates dramatically demonstrated Microsoft's .NET, the Web services platform Microsoft is betting on heavily. But the bigger point is that, unmistakably, Microsoft sees Web serices as a central part of computing's future.

Meantime, in the very same week, Sun Microsystems and its CEO Scott McNealy gave developers a look at the Sun ONE Web services strategy, and this wasn't just an opportunity for McNealy to take jabs at his .NET nemesis (although, of course, there was plenty of that, too). The Sun leader unveiled new and significant tools, including the Sun ONE Starter Kit, a package that promises to give developers what they need to start doing meaningful work in Web services.

The amazing fact: all that happened in the same few days – the Web Services Edge West conference, Microsoft's PDC, Sun unveiling its Web services strategy – and when all that activity is added up, there's no doubt: "Web services is at an inflection point," proclaims Nathaniel Palmer, Delphi Group chief analyst.

Web services, adds Steve Chazin, director of marketing for Bowstreet, an enterprise software developer, has become today's IT cover story and "2002 is poised to become 'The year of Web services.'"

Lico Talamantes, a senior consultant with SEI Information Technology, puts it even more directly: "Web services technology is ready to rock the marketplace."

More proof is that early in November 2001, a parade of IT heavyweights lined up to issue their own Web services announcements. SAP, for instance, debuted mySAP; BEA announced Liquid Data; Borland announced new Java-based tools; Oracle unveiled Oracle9*i* JDeveloper; and IONA unveiled Orbix E2A, a platform built from the ground up with Web services in mind and which,according to IONA, bridge the .NET – J2EE divide.

Whew, it's been hectic, but know this: "In the last 45 days we've seen a major shift in customer interest," says Steve Benfield, CTO of SilverStream Software and a keynote speaker at Web Services Edge West. A message he hammers home: "Customers now believe that Web services can help solve their problems. Web services," he adds, "will change the world."

That is a catchphrase that emerged from the conference and while Web services leaders may – and do – debate key points about how to deploy Web services, there's an emerging consensus that customers will line up to put in orders because they've now realized that Web services is the fast-track way to economically solve key problems.

"Companies – potential customers – couldn't quite get their arms around Web services, but that definitely is changing," observes Eileen Richardson, CEO of Infravio, a Web services developer, and a conference panelist. "Customers," says Richardson, "are seeing in Web services a route to increased productivity coupled with lower costs."

"What customers can expect from Web services, even today, is a real solution that can lower the cost of integration," adds Victoria Schmidt, a product marketer with IONA.

"Customers have spent an incredible amount on IT in the last decade. They want to leverage their investments to achieve new values," says Peter Graf, vice president, marketing, for SAP. "They want to find ways to use their IT infrastructure to reach outside the enterprise. That's why Web services is so significant. Web services lets you keep the systems you've invested in but use them to drive new values."

Why? How? Answer those questions by stepping back to the key themes put forth at Web Services Edge West and a big one is this: "Web services is the thing that lets the three last big things work," says David Chappell, vice president and chief technology evangelist at Sonic Software and another conference keynote speaker, citing Daryl Plummer of the Gartner Group. He adds that a key factor in driving enthusiasm about the future of Web services is that never before has there been so much agreement in the IT community. "For the first time in our lives, every major player in IT is behind one initiative," and with that much computing and marketing muscle behind Web services, there's ample reason for everybody – developers and customers alike – to stay optimistic, says Chappell.

A related fact: "When major vendors collaborated to push Web services standards, the industry began to notice," said Ben Brauer, product marketing manager for Hewlett-Packard's Web services operation.

A second crucial idea: "XML is based on free and open standards," said Charles Goldfarb, the father of XML, who delivered the opening keynote at the conference. What that means is simple: by using XML, various enterprise apps will be able to easily talk with each

other, inside and outside the enterprise. Can they now? Not quite. Bits of enabling technology need to be standardized, but don't fasten on that because another big message from the conference is this: "If you are focusing only on technical details, you are missing the point," says Sean Rhody, editor-in-chief of ***Web Services Journal*** and a conference panel moderator. Exciting as the technology may be, and certainly there remain details to be worked out – this is something of a work in progress. But no matter, insists Rhody, because, more importantly, "Web services is a paradigm shift. With Web services, companies can do new, very interesting things that transform the enterprise."

Dissonance

Is all copacetic with Web services proponents? Don't think that because – as vividly demonstrated at the conference and also by the dueling Sun and Microsoft announcements in the same week – there remain large points of disagreement. Is this cause for fretting that the Web services bubble is about to burst? Don't believe that, not at all. The disagreements, in fact, underline the tumultuous vibrancy that characterizes a community that recognizes it is center stage in IT.

Just where do proponents disagree? Conference attendees heard many points of dispute and concern.

- ***Where are the customers hiding?***

One overarching fact is that as much as there is buzz now about Web services, firm orders are still lagging. But hold on since, predicts Rhody, that will change, and fast, even in the current torpid IT marketplace. "Within six to twelve months, this market will mature," he predicts.

- ***All new, or old wine in a new bottle?***

A key point of debate: Are Web services a new, new thing? Proponents of the viewpoint that Web services is revolutionary include Annrai O'Toole, chairman of Cape Clear Software, and IONA CEO Barry Morris. Or is Web services an evolutionary extension of proven IT capabilities – a viewpoint firmly held by David Chappell, for instance: "Web services has evolved out of existing technologies. SOAP has been around since the mid-90s." Guess what? Just maybe this fight, one of the liveliest at the conference, will be mainly of interest to IT historians because in the trenches, where customers are making decisions, Web services looks to be a wholly new way of enabling applications to work together in ways that previously seemed unimaginable. Is that revolution or evolution? Probably the customer doesn't care, not if Web services deliver on the promise of easier, low-cost integration.

- ***Will the IT power structure change?***

Another hot button at the conference: Barry Morris' assertion that, with Web services, "you're going to see the power structure of our industry change." While Morris sees the new platform as opening tremendous opportunities for players that are comparatively small today, other conference attendees – while seeing potential for smaller players to earn big profits with Web services – nonetheless expressed belief that today's IT leaders, from BEA to SAP, Oracle, and IBM, will likely continue to reap the greatest rewards as they aggressively move into Web services. Who's right? Time will tell but perhaps the key take-away thought is that, huge profits will be earned by companies that articulate an early but coherent strategy for helping enterprises put Web services to work. This isn't just about .NET versus J2EE (and as Annrai O'Toole told conference attendees, "Everyone is bored with Java versus .NET.").

- ***Are Web services overpromised?***

Even some of Web services' biggest fans openly wonder if the technology's capabilities are overpromised: "Web services are expected to be integration nirvana," says Hitesh Seth, chief technology evangelist at Silverline Technologies, an e-business consulting firm. That, as Seth points out, is probably not realistic. Web services will bring useful results but this is no silver bullet, say those who want the industry to adopt a go-slower stance. Others are more optimistic; a case in point is Benfield, who readily admits that Web services once were overhyped, "but companies now are finding ways to deploy Web services that produce real results and the reality is that Web services let one machine talk to other machines. That's important." The take-away thought of conference attendees here is simple: promise only what Web services already can deliver because that will be plenty to stoke interest on the part of the large enterprises that are the first-wave target customers.

- ***B2B on steroids?***

Call this the most closely watched point of contention. Will Web services allow enterprises to easily use commonplace tools (XML and UDDI for service lookup) to find and expand B2B relationships – that is, will Web services help companies to effortlessly find new suppliers and even customers? Some Web services advocates think this is the next wave; others are more skeptical. Either way, Benfield, an ardent advocate of the benefits of the UDDI lookup capabilities, concedes that first deployments likely will be "inside the organization" – that is, Web services will let companies soup up their intranets. But as those test-beds show powerful results, watch out, suggests Benfield. A revived B2B marketplace may be a byproduct of today's Web services technologies.

- ***Just how easy is easy?***

A last, intriguing point of debate: How easy will and should Web services get? "Are Web services tools aimed at developers, or is the analogy to HTML, where the tools got so easy, everybody can use them?" asks Rhody. Just that is a central line of argument. Everybody at the conference agreed that, for the most part, existing tool sets remain too complicated for most users and, the next wave of tools will bring dramatically improved usability. But how usable should tools get? For now, leave that as an open question because the one fact is that Web services' future is still being defined.

And Finally...

Add it up, however, and even with the debates – maybe even because there remain points of contention amid wide agreement about foundation points that enable Web services – nobody left the October conference thinking anything but upbeat thoughts. "This is simple technology, but it will change how business does business," says Benfield, and massive as his prediction is, conference attendees could only applaud his viewpoint that this is the IT solution customers have been waiting for.

Now, don't you wish you had been at Web Services Edge West/XML Edge conference? But even nonattendees can take heart in the good news offered by Rhody: "Web services will continue to grow – it solves real problems, for real customers, cost effectively. This industry is now just beginning." ✎

AUTHOR BIO

Robert McGarvey has covered the Web since 1994 for magazines ranging from Technology Review *to* Upside. *He is the author of the best-selling book* How To DotCom *and a contributing writer for several SYS-CON publications, including* Wireless Business & Technology *and* Web Services Journal.

mcgarvey@sys-con.com

Where Are the Listings?

First off, good magazine! I'm relatively new to Java, after working for many years with (and finally giving up on) Microsoft technologies. ***JDJ*** has a good mix of beginner and expert articles so I can quickly get up to speed on the wide range of technologies out there. I hope you keep an overall expert/cutting-edge level as well as some occasional beginners material in there.

My only gripe and major complaint is that most listings are available online only. I understand that you do this to save space, but I tend to read magazines like this on the bus or underground on the way home from work. As a professional developer in a finance company, I don't have time to sit beside a PC at work and read the articles (unless it's to develop a specific bit of code); when I come home after staring at a screen for 10–12 hours, I don't want to log on and read online or have to print the article.

More important, I bought the magazine in Australia (where perhaps the publishing schedule is out of date as I'm only reading Vol. 6, issue 6 now). I'm now online to print the code of an otherwise unreadable article, "The Impact of EJB 2.0" by Tyler Jewell, which continuously refers to the listings. I have gone online to ultimately print these out and it's available only to subscribers! First, it was hard to find this out as the listings were not in the source code section of Vol. 6, issue 6. I only found it by an author search, which told me it was for subscribers only, then referred me to the magazine subscription page.

The fact that I purchased ***JDJ*** and am unable to read the listings for the articles is an unacceptable situation.

Mark Lamont
lamontmark@hotmail.com

From the editor:

Thanks for your note. We are working on a new system to make the source code more accessible. Watch this space.

Alan Williamson

Why Java?

This is my fifth year as a software developer. What I hate most is rewriting the same routines again and again – one time you're writing it in VB on Win32, another time you're working in C on an embedded system, and so on!

I like to do new things, to investigate new solutions. At this time I'm developing embedded software and am very interested in J2ME. I'd love to develop my applications on a workstation and then see it run on my embedded system regardless of the underlying hardware!

Yes, I know Java is sometimes slow... Well you can compile it! There is a GCC front-end for doing that.

Mirco Bova
mirco.bova@milmail.com

Visual J#.NET

I guess Microsoft is trying to jump on the Java train again since Java is now one of the major players in the software development process. I've been working in the insurance industry for quite a while and noticed that Java is used most of the time in new application development. There are exceptions, for example, if new applications are pure mainframe and are developed in Cobol or PL/1. It's the same in the banking industry. Since both the insurance and the banking industries are the *big* customers for every software vendor, it's obvious that Microsoft has to support Java, but doing it the way it is now – running only on the .NET platform, not supporting the Java standards defined in the Community Process – I think Microsoft is shooting itself in the foot.

Gregor Schneider
gregor@2smart4u.de

Comments on Vince Bonfanti's Editorial

I think Vince Bonfanti's editorial, "J2EE Without EJBs?" (***JDJ***, Vol. 6, issue 11), is dead-on. So many times projects get warped by marketing bull. Web development is already expensive and swaying customers toward an "enterprise" server that they don't need is ridiculous. I agree that there are great reasons to use EJB solutions, but a lot of solutions can be handled without them and, yes, they are J2EE compliant.

Richard Dean
richarddean@yahoo.com

Maybe EJB was a marketing tool exploited by the vendors. The problem is that system architectures and designers also fell for the trap and instead of giving their customers the best value for their money by using JSP/Servlets and JDBC, they went for EJBs, which resulted in the loss of time, effort, etc. Not to say that EJB shouldn't be used; however, it has its own place and purpose.

Lalit Popli
lpopli@consultant.com

We've built large applications without a single EJB – we used JSPs/Servlets with logic encapsulated in JavaBeans. I think the most important factors that make EJBs almost mandatory from a design perspective are the transactional nature of the application and the handling of distributed enterprise resources (multiple databases, messaging, ERPs, etc.). I believe the use of EJBs are justified even if these two factors are a medium to long-term likelihood for a project.

Chandika Mendis
chandika@java-shop.com

InstallShield Introduces MultiPlatform 4.5

(Schaumburg, IL) – InstallShield Software Corp. announced the release of InstallShield MultiPlatform 4.5. Developed with the assistance of Hewlett-Packard Company, MultiPlatform 4.5 creates graphical installations for multiple platforms including the HP-UX operating environment.

MultiPlatform 4.5 (formerly known as InstallShield Java Edition) offers over a hundred new features and enhancements designed to simplify the installation development process. It also has Software Update and Patching capabilities, allowing developers to create installations that upgrade installed products to their latest version.
www.installshield.com

Tower Offers New Service for Migration to Java 1.4

(Austin, TX) – Tower Technology has launched a new service to help companies that are considering the impact of Java 1.4 on their applications. This service allows managers to assess the overall costs and benefits of the transition and, without disrupting internal projects, helps them decide which applications to target and when to make the transition for maximum cost-benefit.

For more information visit www.towerj.com/java1.4migration.htm.

Sonic Software Ships SonicMQ 4.0

(Bedford, MA) – Sonic Software Corporation is shipping SonicMQ 4.0, the next generation of messaging software for the transport of business-critical information throughout the extended enterprise. The release offers enhancements in reliability, large message support, and wireless messaging capability.
www.sonicsoftware.com

Zero G Releases InstallAnywhere 4.5

(San Francisco) – InstallAnywhere, a software deployment solution, offers hundreds of new enhancements including increased platform support, improved installer optimization, and integration with the company's complete software updating solution, PowerUpdate.
www.zerog.com

Ashnasoft Shipping AshnaMQ

(Fremont, CA) – Ashnasoft Corporation, a messaging infrastructure company, is now shipping AshnaMQ. Offered on a wide range of platforms including mobile devices, AshnaMQ features complete implementation of JMS 1.0.2, EJB server integration, and a bundled proprietary message store for high performance and throughput. A free evaluation copy is available at www.ashnasoft.com.

MQSoftware Expands Training to Include Java

(Minneapolis) – MQSoftware, Inc., in partnership with Kenetiks, has expanded its service offerings to include Java programming.

MQSoftware is well known for its experience in delivering MQSeries training; Kenetiks is a worldwide provider of Java training and Java project mentoring featuring IBM VisualAge for Java, WebSphere Application Server, and VisualAge Micro Edition and Sun J2SE, J2EE, and J2ME. Classes will be offered at the MQSoftware training facility in Minneapolis.
www.kenetiks.com
www.mqsoftware.com

Passport's Wireless App Receives Honors in Java Contest

(Paramus, NJ) – Passport Corporation's InstiTrade prototype was honored with the Best Overall Application Award at the recently held "***Java Developer's Journal*** J2SE for Handhelds Developer Contest."

InstiTrade enables users to interactively perform stock trades with each other using small wireless handheld devices, a requirement of the Java Developers' competition. Developers were challenged with building J2SE applications specifically for the Compaq iPAQ and powered by the new SavaJe XE operating system.
www.passportcorp.com

SchlumbergerSema Markets FIPS 140-1 Level 2 Certified Java-Based Smart Card

(*Austin, TX*) – SchlumbergerSema announced landmark FIPS 140-1 Level 2 certification for its Cyberflex Access 32K smart card. The card is the first of its kind to receive this certification by the National Institute of Science and Technology (NIST), and is the only card available that fully meets the stringent Department of Defense (DoD) Common Access Card (CAC) specifications for Java-based smart cards with cryptographic applications.

The DoD will use the secure, robust Cyberflex Access smart cards for physical identification, and building and network access in a multitiered program.
www.slb.com

Aligo Announces Wireless App Development Contest

(San Francisco) – Aligo, Inc., announced its first annual Wireless Application Development Contest beginning November 1 and running through January 15, 2002. The grand prize, $5,000 and public recognition, will be awarded by judges from ***Java Developer's Journal***, Sun Microsystems, and Aligo to the developer who designs the most compelling mobile application that solves a real-world problem.

The second prize is $1,000 and the third prize is a Sony Clié PDA worth $500. The judging panel will announce the winners on January 31, 2002.
www.aligo.com/developer/contest1.htm.

Computer Associates Released COOL:Plex 5.0

(Islandia, NY) – Computer Associates International, Inc., introduced COOL:Plex Release 5.0, an innovative development tool that reduces the cost of designing and delivering Java-based e-business applications for the Microsoft Windows NT/2000/XP and IBM iSeries 400 platforms.

The newest version of COOL:Plex also offers an environment for building new applications from scratch, since its Software Patterns – reusable designs that solve common development productivity challenges – allow developers to make more effective use of their existing skill sets to design and deliver high-value Java, wireless, and XML applications.
http://ca.com

Microsoft Signs Consent Decree with Government

(Redmond, WA) – Microsoft Corp. signed a consent decree with the U.S. government to settle a three-year antitrust case, accepting restrictions on the company for five years.

The settlement calls for the company to give computer makers more flexibility on software, and requires Microsoft to share the inner workings of its Windows operating system with other software firms.

The pact, which must be approved by a federal judge and is not yet endorsed by the states that joined the antitrust case, is a far cry from splitting the company in two, a remedy sought by the administration of President Clinton.
www.microsoft.com

Next Month

J2ME BENCHMARKING
Evaluating an application's performance objectively
by Carl Barratt and Glenn Coates

J2EE APPLICATION SECURITY
Container- versus application-managed security
by Timothy Fisher

J2EE TESTING: WHAT'S IT TO YOU?
An overview of the CTS for J2EE component developers
by Jonathan Maron

GOT A PROBLEM WITH J2EE?
Doctor Java has a cure
by James McGovern

THE KEY TO SUPERIOR EJB DESIGN
The specifics of EJB and how they influence the design of EJB systems
by Boris Lublinsky

JAVA DEVELOPER'S JOURNAL

Where We've Been and Where We're Going

Highlights from 2001

WRITTEN BY
BILL BALOGLU & BILLY PALMIERI

This past year has been a tumultuous one, initially for our high-tech industry, and then for our nation and the world at large. We don't presume to know what the coming year will bring.

But in the 12 months we've been writing this column, we've covered many topics and offered some professional advice to Java engineers that we still think is pretty sound. So here are some highlights of the subjects that we've touched upon over the course of a challenging year.

Types of Engineers

In several columns we focused on defining the types of Java engineers working in the industry. We initially categorized them into two groups: the Knows-Java engineer and the Understands-Java engineer.

The Knows-Java engineer may have a strong background in networking or databases with skills in SQL, Oracle, or PowerBuilder. Aware of the strong demand for Java skills, these engineers pick up a Java book, take a Java course, or "teach" themselves Java.

While these skills may be sufficient for basic Java projects, on mission-critical positions the Knows-Java engineer often discovers the limits of his or her Java skills quickly, resulting in bad blood and burned bridges with the hiring manager and client company.

The Understands-Java engineers typically have a background in C++ programming and a strong grasp of object-oriented methodologies. They have Java programming experience (more than two or three years) and have worked with J2EE (EJBs), among other skills.

The Understands-Java engineer has experience with large, scalable, multithreaded applications. He or she knows what can be done and what can't be done in a Java environment. If there's a problem or a bug, he or she knows how to fix it.

A senior engineer who's on top of the industry and in highest demand has experience and skills in the hottest technologies, such as J2EE, EJBs, XML, XSLT, and application servers (like WebLogic, WebSphere, iPlanet, or Tomcat), and experience with servlets.

J2EE has a high learning curve, so a lot of experience is required of an engineer who claims to be at the senior level. J2ME (Java 2 Micro Edition) and WML are important for writing applications for small and wireless devices.

Books and courses on these technologies are a good place to start. But on-the-job training, either full time or contract, is still the fastest, most effective way to pick up real-world experience with a variety of new technologies.

The Hiring Process

We've covered many of the steps in the hiring process, from the résumé to the interview.

The most common problem with résumés is that they either misrepresent candidates as being more experienced than they really are or misrepresent seasoned candidates as less experienced than they really are.

A good résumé details who you are, what you've done, and what you want to do. It highlights how your skills and experience are relevant to the position you're applying for, and provides lots of specific detail – especially on skills you've used, what you've done with them, and where.

The goal of the résumé is to get you a phone screen interview. The goal of the phone screen is to get you a face-to-face interview.

Throughout the process, be prepared to answer questions about your technical expertise and work experience, as well as career-focused questions about why this position is a good fit for you right now. Be prepared to explain why your last job ended, or, if you're currently employed, why you want to leave.

The single most important element that turns a job applicant into a "new hire" is the level of preparation before the interview. Research the company's Web site, print a copy of the job description, and tailor your "interview pitch" to the specific needs of the position and the company.

For a face-to-face interview, know the names of the people you'll be interviewing with, arrive a few minutes early, and always look sharp (a little sharper than you might dress for a typical day at work).

Working with Agencies

There are many good, reputable agencies that provide valuable services and opportunities for engineers. But there are many more agencies eager to make a "quick sale" without much regard for solid business ethics or an understanding of who you are and what you do.

Since you're known by the company you keep, run the following qualifications by any agency that wants to represent you:

- Does the agency ask detailed questions about your skills and background and how well they fit the position?
- Does it ask to meet you in person or speak with people who can verify your expertise?
- Is it interested in building a long-term working relationship with you?
- How long has it been providing candidates to that client?
- How many of its consultants are currently working with that client?
- Ask to speak to one of the engineers who currently works with the agency. You can get valuable insight from someone else who walks in your shoes.

What's in Store for the New Year

As we look to the new year, we'd like this column to address your specific questions and problems as a Java professional. Please e-mail us at jdjcolumn@objectfocus.com with your questions and concerns, so we can have a collaborative, productive new year together.

Happy holidays and our best wishes to all of you.

jdjcolumn@ObjectFocus.com

AUTHOR BIOS

Bill Baloglu is a principal at ObjectFocus (www.ObjectFocus.com), a Java staffing firm in Silicon Valley. Previously he was a software engineer for 16 years. Bill has extensive OO experience and has held software development and senior technical management positions at several Silicon Valley firms.

Billy Palmieri is a seasoned staffing industry executive and a principal at ObjectFocus. His prior position was at Renaissance Worldwide, a multimillion-dollar global IT consulting firm, where he held several senior management positions in the firm's Silicon Valley operations.

sys-con media

www.sys-con.com

Ah, Youth...

WRITTEN BY
BLAIR WYMAN

Lately I seem to be getting younger. Oh, don't get me wrong – I'm not complaining – it's just that puberty was bad enough in the forward direction; the prospect of going through it backwards leaves me a little unsettled.

Fortunately, if the rate at which I'm getting younger matches the rate I had previously been aging, I've got many years to prepare.

Why do I think I'm getting younger? Well, there are actually several indicators. I'm feeling a little more fit, and getting back some muscle tone in my upper body. I've actually got a great physique, but unfortunately it is currently encased in a lumpy sheath of unrefined lard. I guess you might say that I'm hiding my "six-pack" behind a quart of chocolate ice cream.

Now, part of the credit for my reverse metamorphosis certainly goes to my three beautiful children. Hmmm….Now that I think about it, I sort of wonder whether I might actually just be tapping into their voluminous reservoirs of youth, which is in turn causing them to age prematurely. (Heaven knows, my 14-year-old talks like she's 30-something.) Since parents usually get blamed for everything anyway, I might as well admit to my little subterfuge now. (Just don't tell Oprah.)

And, to be fair, I guess that part of the credit – at least that part that involves my muscle tone – has to go to my rediscovery of something I had always thought of as a long-vanished part of my misspent youth: foosball.

No, I didn't say *football*. And no, I don't mean *soccer*. By *foosball* I mean that table game you used to find in (only the nicest) bowling alleys and biker bars: the game with the plastic figures attached to steel rods that you twist violently to kick a little ball around. Sometimes it's called "table soccer," but it's always been "foosball" – or just "foos" – to me.

Now, as ridiculous as it seems, foosball was a big part of my youth. In my early teens it was a nightly ritual to beg a quarter from my folks just so I could walk a couple of miles to the bowling alley and play some foos.

There was always a group of "regulars" that would assemble every night to play, and some of them were quite good. Early on, I'd put my lone quarter on the table, pick a partner, and be soundly and humiliatingly defeated.

Well, I'm nothing if not obsessive, so foosball became a big part of my life. At one point in high school – just about the time it became clear that I had utterly sacrificed any passing grade in physics upon the altar of foos – it had been literally months since I had been defeated. In my tiny little east-side-of-town ocean, I was one big foosball fish.

Then I decided to travel to the other side of town – to the "other" bowling alley – determined to show them a thing or two about foosball. That's when I ran into a buzzsaw we'll call "Steve," who had the fastest and most accurate series of mesmerizing mystery shots I'd ever seen. I was never the same. The relative importance of foos promptly passed its zenith, and although I continued to play, I knew I would never be "the best."

Gradually, with the nascence of video games, foosball tables were replaced with *Pong* and *Lunar Lander* and *Tempest*. Video games took less room, made more money, and were effectively "zero maintenance" (at least, when compared to foosball tables, which constantly needed adjustment or lubrication). Foosball slowly evaporated from my life, leaving a sort of dried psychic scum in its place.

Well, when I started at IBM, I heard about a fellow programmer who had played foosball professionally in the late '70s, making literally hundreds of dollars per week. I sought out this kindred spirit, we shared some foos stories, and visions of foos again danced in my head.

Several months ago, the powers that be installed foosball tables in a couple of the newly remodeled break rooms. On a lark I started playing again, and started enjoying my little "breaks." Most of my old skills came rushing back, along with some new ones, and the sheer joy of utterly decimating the puppy dog–eyed 20-somethings – especially at what they presumed was "their game" – is what we in Minnesota politely call a "hoot." (It should be obvious by now, Dear Reader, that I'll take any respect I can get.)

So, anyone for some foos? I think I still have a quarter here somewhere...

AUTHOR BIO
Blair Wyman is a software engineer working for IBM in Rochester, Minnesota, home of the IBM iSeries.

blair@blairwyman.com

Java Dudes