XML DEVCON 2000 FULL PROGRAM DETAILS

XML DevCon 2000 16-PAGE INSERT INSIDE!

JAVA™ DEVELOPERS JOURNAL

The World's Leading Java Resource

Volume: 5 Issue: 5 May 2000

www.javadevelopersjournal.com

Implementing Business Rules in Java

XML for JMS Messaging **in E-Commerce**

SQL Embedded in Java: Part 1 **Starting Out**

Presentation Logic and EJBs: **Session Beans**

NetRexx Programming **for the JVM**

SYS-CON MEDIA

Progress SonicMQ

www.progress.com

Protoview

www.protoview.com

Microsoft

www.microsoft.com

JDJ CONTENTS

JAVA DEVELOPER'S JOURNAL
VOLUME: 5 ISSUE: 5 MAY 2000

No
Magic

www.no-

magic.com

Togethersoft LLC

www.togethersoft.com

JAVA DEVELOPERS JOURNAL

EDITORIAL ADVISORY BOARD

TED COOMBS, BILL DUNLAP, DAVID GEE, MICHEL GERIN, ARTHUR VAN HOFF, JOHN OLSON, GEORGE PAOLINI, KIM POLESE, SEAN RHODY, RICK ROSS, AJIT SAGAR, RICHARD SOLEY, ALAN WILLIAMSON

EDITOR-IN-CHIEF:	SEAN RHODY
EXECUTIVE EDITOR:	M'LOU PINKHAM
ART DIRECTOR:	ALEX BOTERO
PRODUCTION EDITOR:	CHERYL VAN SISE
SENIOR EDITOR:	JEREMY GEELAN
ASSOCIATE EDITOR:	NANCY VALENTINE
EDITORIAL CONSULTANT:	SCOTT DAVISON
TECHNICAL EDITOR:	BAHADIR KARUV
PRODUCT REVIEW EDITOR:	ED ZEBROWSKI
INDUSTRY NEWS EDITOR:	ALAN WILLIAMSON
E-COMMERCE EDITOR:	AJIT SAGAR

WRITERS IN THIS ISSUE

CAROLE ANN BERLIOZ, MATTHEW BROWN, RICK HIGHTOWER, MICHAEL LACY, COLLEEN MCCLINTOCK, JOE MITCHKO, SEAN RHODY, EKKEHARD ROHWEDDER, JON SIEGEL, ARTHUR VAN HOFF, GORDON VAN HUIZEN, JASON WESTRA, ALAN WILLIAMSON, SUSAN YESHIN

SUBSCRIPTIONS

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

SUBSCRIPTION HOTLINE: 800 513-7111

COVER PRICE: $4.99/ISSUE

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

PUBLISHER, PRESIDENT AND CEO:	FUAT A. KIRCAALI
VICE PRESIDENT, PRODUCTION:	JIM MORGAN
VICE PRESIDENT, MARKETING:	CARMEN GONZALEZ
ACCOUNTING MANAGER:	ELI HOROWITZ
ADVERTISING ACCOUNT MANAGERS:	ROBYN FORMA
	MEGAN RING
JDJSTORE.COM:	JACLYN REDMOND
ADVERTISING ASSISTANT:	CHRISTINE RUSSELL
ADVERTISING INTERN:	MATT KREMKAU
GRAPHIC DESIGNERS:	JASON KREMKAU
	ABRAHAM ADDO
GRAPHIC DESIGN INTERN:	AARATHI VENKATARAMAN
WEBMASTER:	ROBERT DIAMOND
WEB SERVICES CONSULTANT:	BRUNO Y. DECAUDIN
WEB DESIGNERS:	MICHAEL MALOOF
	STEVEN KILMURRAY
WEB SERVICES INTERN:	DIGANT B. DAVE
	BRYAN KREMKAU
CUSTOMER SERVICE MANAGER:	CAROL KILDUFF
CUSTOMER SERVICE:	ANN MARIE MILILLO

EDITORIAL OFFICES

SYS-CON PUBLICATIONS, INC.
39 E. CENTRAL AVE., PEARL RIVER, NY 10965
TELEPHONE: 914 735-7300 FAX: 914 735-6547
SUBSCRIBE@SYS-CON.COM

JAVA DEVELOPER'S JOURNAL (ISSN#1087-6944) is published monthly (12 times a year) for $49.00 by SYS-CON Publications, Inc., 39 E. Central Ave., Pearl River, NY 10965-2306. Periodicals Postage rates are paid at Pearl River, NY 10965 and additional mailing offices. POSTMASTER: Send address changes to: JAVA DEVELOPER'S JOURNAL, SYS-CON Publications, Inc., 39 E. Central Ave., Pearl River, NY 10965-2306.

WORLDWIDE DISTRIBUTION BY

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

SEAN RHODY, EDITOR-IN-CHIEF

A Brand New Façade

Some trends you just don't see coming, like the return of bell-bottom pants. They are mysterious and leave you wondering what people can possibly be thinking about. Other trends are much more comprehensible and when they start to manifest themselves, you tend to wonder why on earth you hadn't seen them coming in the first place....

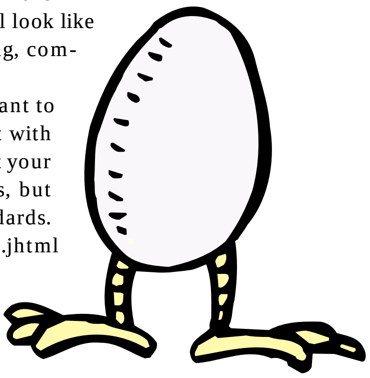
One such development that took me by surprise is the explosion of interest in HTML via Java, either by servlets or JSP. Those of you who did see this coming, feel free to feel superior. For the past year or so I've concentrated on Java's impact on the server space, particularly with Enterprise JavaBeans. But I'm starting to see another trend emerging – the use of Java and HTML to present attractive, customized interfaces to end users.

I attribute this to a chicken-and-egg type of relationship. Up to about a year ago there were only a small number of servlet engines and the JSP specification was still churning. That's the chicken. The egg part was the demand, or market space, for products that would support these technologies. Companies like BEA had servlet support in WebLogic, but you weren't likely to buy WebLogic to serve dynamic HTML – too much like swatting a fly with an atom bomb. Other products that emerged included JRun, ServletExec, ATG Dynamo and 10 BaseJ.

Last month (in editorial time, not calendar time), Persistence purchased 10BaseJ with the intent of integrating JSP into their PowerTier server, and BEA is partnering with Macromedia to add support for JSP into Dreamweaver. The number of files that I see on the Internet that end in .jsp keeps growing.

JSP represents a middle ground where Java programmers and HTML designers can meet. The servlet specification was fine for us Java folks, but it was too cryptic for the HTML people, who wanted to add some additional horsepower to their code but didn't want to understand how to get a Reader or a Writer. The ability to embed Java code in HTML is an advantage for programmers who want to see more of what the file will look like when it gets to the browser, rather than having to imagine it while coding in a traditional IDE.

One problem with JSP, though, is the lack of a good interface. Since I use Dreamweaver to write HTML, my hope is that BEA and Macromedia can go beyond simple support for tags and dive deeper into true understanding of Java in the HTML editor. Ideally, I want an editor that can show me what it will look like when it hits the browser, while still providing syntax checking, compiling and the like.



I'd also like it to be portable. By this I mean that I don't want to have to use a separate dialect to achieve this. I want to do it with standard JSP. Some good alternatives are available if that's not your need: both ATG Dynamo and ColdFusion provide solid IDEs, but each requires some compromises when it comes to the standards. Of the two, I think ATG is closest to the JSP standard with their .jhtml files. Their prebuilt components also reduce the amount of coding that's required – and ATG does support embedded Java in HTML.

But I want to be able to walk away from one system and take my code to another. Not the code per se, but the techniques and the knowledge. I want the same thing for EJB too; maybe I'll get there one day.

So I missed the trend on this one. I should have seen it coming much earlier, but I'll be on the lookout for it now. In the meantime I'm going to dust off my old plaid pants – maybe that'll be the next big thing. (I sure hope not.)

sean@sys-con.com

AUTHOR BIO

Sean Rhody is the editor-in-chief of Java Developer's Journal. *He is also a principal consultant with Computer Sciences Corporation where he specializes in application architecture – particularly distributed systems.*

IMPLEMENTING

The first in a two-part series on how and why to integrate a 'rule' engine into a Java application

PART 1

WRITTEN BY COLLEEN MCCLINTOCK & CAROLE ANN BERLIOZ

Twenty years after it first made waves, rule-based technology is making a comeback. Java developers with an eye on the e-commerce market are becoming aware of how integrating business rules and objects in Java can help expand Java into new niches within Web-based applications. This article discusses how rules fit with Java, the types of rule engines available and how a rule engine is used to execute rules for integration into a Java applet or application.

Business rules are the fundamental policies and procedures that define or constrain a business, guiding how it functions. In modern businesses thousands of these policies and procedures are often embedded in application code, with "rule engines" – intelligent software components – frequently used as the fastest and most effective method to evaluate and execute business rules. Relying on rule engines is usually preferable to embedding rules in application logic, which makes changing and maintaining rules difficult and costly. Developers are increasingly using them, especially to reduce development and maintenance time, increase application performance, and improve application adaptability and flexibility.

In many ways Java is an ideal language for business rules because platform-independent, robust, maintainable, object-oriented Java applications more accurately reflect the structure of a business. Even though the Java language doesn't currently support rules, new tools – including freeware – are helping to overcome Java's limitations in mak-

BUSINESS RULES IN JAVA

ing changes or updates. For developers, being aware of rule engines and the available tools for Java is highly applicable to the burgeoning e-commerce market. Having an approach for integrating business rules and objects in the Java language can be a boon to expanding Java into new niches within Web-based applications – particularly for fast-growing applications like online lending and investment management, and for the many Web personalization solutions ideally suited to business rules and rule engines.

In this article we'll also address design/implementation strategies for business rules and the changing requirements driven by the Web and by e-commerce that mean nonprogrammers must have access to the technology. It's important to note that there are other rule-based software systems, but this article will discuss only Java implementations

Rule-Based Systems: A Brief History

Rules are declarative statements that drive activity in a software application by describing the action or actions to take when a specified set of conditions is met. Rules consist of a left side – an "IF/WHEN" condition statement – and a right side – a "THEN" action statement.

The earliest rules technologies had their roots in artificial intelligence (AI) methodologies. MYCIN (written in LISP), developed at Stanford University in the 1970s, was the first system to use rule-based knowledge representation. Used to diagnose blood diseases and recommend treatments, MYCIN's core concept was the separation of knowledge and control. Knowledge, represented in rules, was separated from the control logic responsible for evaluating and executing the rules.

Rule-based programming reached a pinnacle in the 1980s during the AI boom. At this time many vendors marketed, and many organizations purchased, commercial rule engines. However, the combination of poor performance, an inability to integrate the technology into mainstream computer architectures and the realization that rule-based declarative programming was not the panacea it was purported to be left many organizations disenchanted with the technology.

Although the hype surrounding rule-based programming and expert systems died in the late '80s, the technology continued to be used effectively and quietly for certain types of applications. The current trend toward developing business rule applications is once again leading many mainstream organizations to evaluate the use of rule-based technology more than 20 years after its introduction.

Among the first users of the rules technology was the telecommunications industry, particularly for network management. System and network management applications are typically developed using rules. The following are examples of rules in a telecom network management application:

- ***Related alarms:*** When a switch raises an equipment failure alarm, ignore all communication failure alarms on the associated port.
- ***Maintenance activity scheduled:*** When maintenance activity has been scheduled on a network element, ignore all alarms raised on this piece of hardware between the maintenance start and end time.
- ***Critical alarms on sensitive devices:*** When more than 15 alarms are detected on a network element, notify the network operator by pager. When more than five alarms are detected on a switch, notify the network operator by pager.

The class diagram associated with these rules is shown in Figure 1. If you were to develop an application to implement these rules, how would you do it? Chances are you'd code them into the methods of one or more objects. Listing 1 contains the Java code to implement these rules using a correlateAlarm() method on the Network class. The Network generates the alarms and events, which queue and are processed by the AlarmCorrelator class. The AlarmCorrelator class executes the correlateAlarm() method each time an alarm is processed.

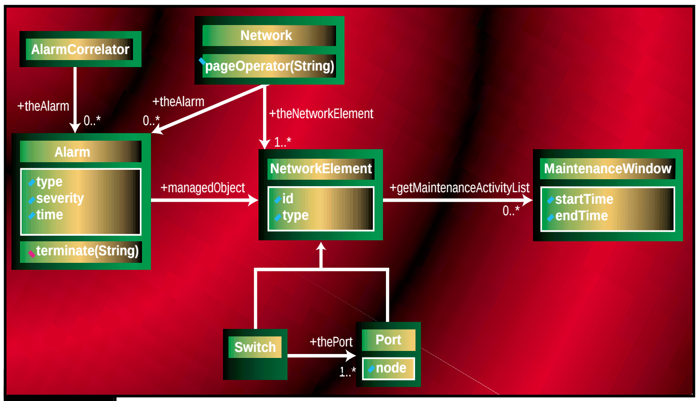


FIGURE 1 A telecommunications network management class diagram

The "correlate Alarm" method contains complex logic that becomes difficult to maintain as the number of rules increases. Typically, hundreds of such rules are implemented in a network management application. The processing time to handle the generated alarms becomes significant since all tests must be evaluated each time an alarm is received. To reduce processing time, rules are often combined into the same "IF" or "SWITCH" statement. This further obscures the code since the one-to-one relationship between the rule specification and the implementation is lost.

This type of application is difficult to develop and maintain using Java or any other procedural programming language. Java objects consist of methods for executing procedures and data members to hold the inputs and outputs of these procedures. They lack a mechanism for capturing rules that determine when and how procedures should be executed.

Enter Rule Engines

A rule engine is a software component designed to evaluate and execute rules. Since it implements all of the logic necessary to perform rule evaluation and execution, the rules can be coded as stand-alone atomic units, separate from and independent of the rest of the application logic. This makes them easier to develop and maintain. Rule engines have a proprietary rule language for writing rules. To implement an application using a rule engine, you write the rules in the rule language and embed the rule engine into your application. The rule engine integrates with the application through an API that controls the loading of rules into the rule engine, the monitoring of application objects referenced by rules and the execution of rules.

As shown in Figure 2, the rule engine as a software component is an object with attributes and methods. An application consists of the application objects and the rule engine object. The rules are loaded into the rule engine by invoking a method on the rule engine object and passing the rules – in a file, stream, XML representation or some other form – to the rule engine. To evaluate the rules, the rule engine must have visibility to the application objects referenced by rules. To provide this type of visibility, the object data members and methods referenced by rules are usually specified as public. Through a process called *asserting*, the rule engine's assert method introduces the objects referenced by the rules. Once the objects have been asserted to the rule engine, the rule engine maintains a set of references to the objects. The rule engine's fire rules method is then invoked to cause the rule engine to evaluate all of the loaded rules. Following evaluation, when the conditions of rules are met, the rules are executed or fired, causing their action statements to execute. The action statements of rules can modify the objects referenced by the rule engine, thereby causing more rules to become eligible to fire. The process continues until no more rules are eligible to fire, at which point control returns to the application statement following the invocation of the fire rules method.

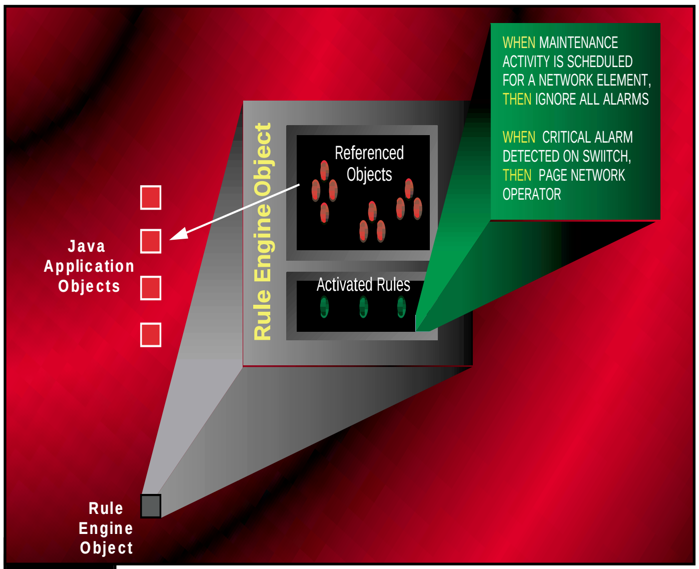

WHEN MAINTENANCE ACTIVITY IS SCHEDULED FOR A NETWORK ELEMENT, THEN IGNORE ALL ALARMS
WHEN CRITICAL ALARM DETECTED ON SWITCH, THEN PAGE NETWORK OPERATOR
Rule Engine Object
Referenced Objects
Activated Rules
Java Application Objects
Rule Engine Object

FIGURE 2 The rule engine in a Java application showing sample rules indicative of a network management application

Let's go back to the network management example. The network management rules would be implemented using a rule engine by coding the rules shown in Listing 2. These rules are shown using the Java-like syntax of JRules, a rule engine from ILOG Inc. Notice that the syntax of the rule language is declarative and has a one-to-one correspondence with the English-like rule specifications. The rule engine would be integrated into the application as an object and the rules would be loaded, objects asserted and rules fired as described previously.

The rule engine implementation results in a flexible and maintainable application design since the rules are separated from the application logic and can be maintained in a single place. The rule engine implementation is straightforward, with the evaluation of rules automatically performed by the rule engine, eliminating the need for the complex conditional logic required in the application to monitor the network objects. The use of a rule engine improves application performance since the rule engine implements an efficient pattern-matching algorithm to optimize rule evaluation. In this algorithm, known as the RETE algorithm, rules are compiled into a discrimination network that enables conditions shared by multiple rules to be evaluated just once. This enables the processing of thousands of network alarms per second.

Table 1 lists several commercial and shareware rule engines that integrate with the Java language. The rule languages implemented by the rule engines and their mechanisms of integration differ slightly.

Rule engines have been used for event-driven applications such as system and network management and workflow for some time. As we mentioned before, these applications benefit from the RETE pattern-matching algorithm implemented by the rule engine to efficiently evaluate rules against the set of referenced objects. However, rule engines are rapidly gaining acceptance as the preferred approach for implementing a new class of applications known as business rule applications.

Business Rules

The term *business rule* has been used to define business policies and procedures that describe or constrain the way an organization conducts business. Organizations have thousands of such business rules that

Computer Jobs

www.computerjobs.com

RULE ENGINE	VENDOR	COMMENTS	FOR MORE INFORMATION:
Advisor	Blaze Software 150 Almaden Boulevard San Jose, CA 95113 408-275-6900	100% Pure Java Certified	www.blazesoft.com
JES	Ernest J. Friedman-Hill Distributed Computing Systems Sandia National Laboratories Livermore, CA	100% Pure Java Certified, freeware	http://herzberg.ca.sandia.gov/jess/
JRules	ILOG Inc. 1080 Linda Vista Ave. Mountain View, CA 650-567-8000	100% Pure Java Certified	www.ilog.com

TABLE 1 Some commercial and shareware rule engines

define and direct how they manufacture, sell, buy and otherwise transact business. Business rules are everywhere – in your corporate charter, your marketing strategies, pricing policies, product and service offerings, and customer relationship management practices as well as in the legal documents that regulate your business and industry.

Business rules can be differentiated from the executable rules coded in the rule language of the rule engine since they directly represent business policy. Other types of rules, such as rules to perform data manipulation or control application-processing flow, can be coded in the rule engine. Furthermore, business rules don't require a rule engine for implementation. They can be implemented in computer systems using a variety of different approaches: for example, they can be implemented in middle-tier application servers in the methods of objects or in database stored procedures. In fact, business rules are implemented throughout computer systems.

To express and model business rules, it's desirable to use an English-like, implementation-independent representation. Such a representation allows nontechnical business people, those responsible for defining and maintaining the business rules, to understand and use them. Minimally, an English-like representation for business rules can help in creating better requirements specifications. However, some applications require that the business rules be represented and externalized in a way that permits business people to create and modify them at execution time. In other words, the application should allow the business rules to be added and changed dynamically by nontechnical users.

For the purposes of this article, we'll refer to applications with these requirements as "business rule applications." This could be clarified further as follows: in order to be considered a business rule application, the application must allow business people to define business rules using an English-like representation. Thus a claims-processing application that allows the user to modify the business rules specifying the handling of auto insurance claims and a portfolio management application that allows the user to create guidelines for buying and selling investments are both business rule applications.

Implementing Business Rules

One method of allowing business rules to be managed by business people at execution time is to create a set of parameter tables that can be stored in databases and edited via a GUI. This method is often used for applications implementing product pricing and bundling. The flexibility gained by this method is very restricted in scope, and when business policy changes fall outside the original application boundaries, substantial programming changes to the application's front end and back end are required.

As stated, a more adaptable approach is to implement the business rules using a rule engine. Since the rule languages offered by the commercial and shareware rule engines are extensive and powerful, this approach provides a more flexible architecture that is unlikely to require change when the scope of the business rules change. Some of the rule engines have an English-like, readable rule language and a rule editor that can be embedded in the application's GUI to provide runtime access to business rules. However, this approach is feasible only for technically savvy business people because the rule languages provided by rule engines are complex and extensive whether they have an English-like or a codelike representation. When users modify business rules, it's difficult for them to determine whether the rule will execute as expected. Using the full syntax of a rule language, users can create a rule that will cause the application to loop endlessly or crash.

To avoid these problems, the application must be designed to restrict the syntax and features of the rule language to a safe subset of the rule language provided by the rule engine. The restricted rule language, known as the business rule language, is usually specific to the application's business domain and uses business terminology particular to that domain. Developing a custom rule editor GUI to guide the user through the specification of rules using the business rule syntax normally provides the capability to define and modify business rules. Two types of rule editors are commonly created for business rule applications. In the first, the business rules are represented as predefined templates that allow the user to specify only the parameters of the business rule. Also used are more sophisticated rule editors that allow the user to define a rule using a business rule syntax and step the user through the rule creation process based on previous choices.

Using either approach, business rules created through the custom business rule editor can then be represented in objects or stored in a parsed format, a text string or XML representation. Executable business rules in the language of the rule engine are then generated from this intermediate representation.

For example, in a customer resource management (CRM), the business user from the marketing department may define a special new promotion for a certain group of customers. The business user creates the rule in an English-like representation using the business rule editor. An executable business rule, in the language of the rule engine, is generated from the business rule language and introduced to the rule engine. The rule engine executes in a CRM Web server-based application, as illustrated in Figure 3.

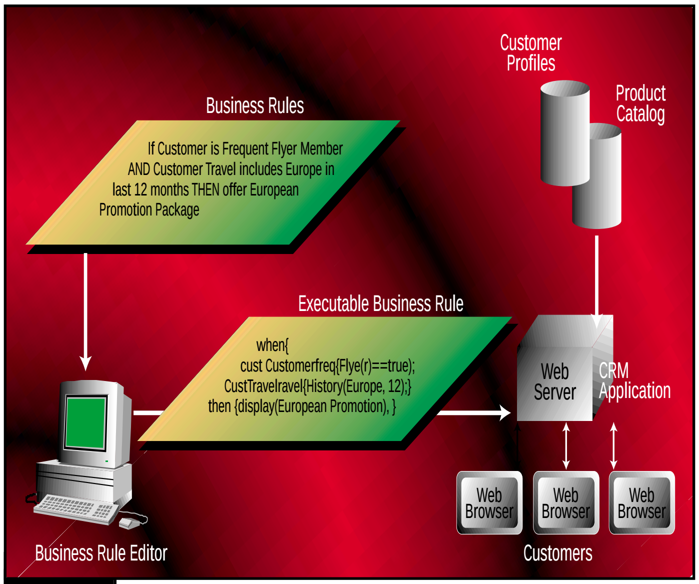
This approach, although challenging to implement, results in a powerful and flexible application in which the business rules can be changed dynamically while the application executes to immediately implement new sales policies or pricing strategies or to respond to changing regulations imposed by government or regulatory agencies.

Applications Requiring Business Rules

In today's highly competitive business environment, it's becoming increasingly necessary to externalize business rules and put them in the

Insignia Solutions

www.insignia.com



If Customer is Frequent Flyer Member AND Customer Travel includes Europe in last 12 months THEN offer European Promotion Package

```
when{
  cust Customerfreq{Flye(r)==true);
 CustTravelravel{History(Europe, 12);}
then {display(European Promotion), }
```

FIGURE 3 Network management rules implemented in the ILOG JRules rule language

hands of business people. Organizations can no longer afford to wait for IT to implement changes in business policy. To gain and maintain a competitive advantage, companies must be able to immediately respond to the requirements of their customers and partners and to the initiatives and actions of their competitors.

Business rule applications exist in virtually every business domain. Let's look at a selection of typical business rule applications in the areas of e-commerce personalization, loan underwriting and customer relationship management.

E-COMMERCE PERSONALIZATION

As more companies extend their sales and marketing functions from traditional channels to the Web, the capability of personalizing each customer's experience becomes essential in order to attract, satisfy and retain customers. With the Web comes the ability to apply the concepts of one-to-one marketing and to replace the old "one size fits all" approach with individually tailored offers.

Rule-based personalization uses specific information about individual Web site visitors to precisely tailor the content displayed. By capturing the needs, interests, preferences and motivations of the individuals that visit a Web site, and applying business rules against that information, visitors can be provided with information and recommendations that will be relevant to them. Web merchants can make visitors aware of interesting promotions, offer them discounts based on their purchasing history and take advantage of opportunities to cross-sell and upsell without offending them. Here are some typical rules:

- If Shopping Cart contains more than $100 worth of CDs, give $5 off the next purchase.
- If Customer purchased a Pilot III last week, show all Pilot accessories.
- If Customer traveled to Europe within the last year, send notification of European travel promotions.

Some larger merchants are developing their own personalization solutions using business rules. By externalizing the business rules, they create a more flexible solution that can be changed by their marketing personnel, enabling them to immediately implement new promotions or offer new products. Vendors such as BEA and IBM are offering flexible personalization solutions that allow the merchant to define and modify these types of rules in their respective WebLogic Commerce Server and WebSphere Commerce Suite. Using these personalization solutions, merchants can create Web sites that intelligently interact with customers.

MORTGAGE LOAN UNDERWRITING

Lenders have been developing rule-based underwriting applications for the last decade. In these systems, rules are used to simulate the decision-making process of a human underwriter. As competitive pressures have increased in the mortgage industry, lenders have developed more loan products to offer flexible alternatives to homebuyers. At the same time, the credit policy organizations within lending institutions are defining new rules to ensure that risk is managed at an acceptable level for these new loan products. Additionally, lenders must frequently comply with regulations imposed by the federal government and the policies of the secondary mortgage institutions.

To implement and maintain the large number of rules required in these underwriting applications, many lenders are developing business rule applications that allow the marketing department to define new loan products and credit policy to define new risk management rules. For example:

- If borrower is a first-time homebuyer, offer the no-down-payment option.
- If the lien type is a second mortgage, the borrower's occupancy status must be principal residence.
- If the transaction is a cash-out refinance, the loan-to-value ratio must be less than or equal to 85%.

TELECOM CUSTOMER CARE AND BILLING

Competition and deregulation have combined to make the customer all-important to the telecom industry. In order to excel, service providers must continually offer the customer new services at competitive prices. Moreover, they must be able to react immediately to market changes from competition, customer demand or technological advances. These market pressures force the companies to constantly search for new ways to stimulate business through service packages and pricing schemes, and to run aggressive sales campaigns with attractive promotions and discounts.

The most innovative applications for customer care and billing are business rule applications because they offer time-saving, cost-effective ways to impart outstanding flexibility and customization for managing ever-changing rates and promotions. Business rules help companies secure customer loyalty by allowing them to target individual needs.

The long distance market is extremely competitive. To attract and keep customers, companies are offering special rates and discounts. For example:

- If destination is a preferred number, give a 50% discount on call.
- If customer is a member of nickel nights plan and call time is after 5 p.m., billing rate is $0.05 per minute.

To be effective, these promotions must be incorporated into the billing system. Externalizing the business rules in the billing system and allowing the marketing department to define and modify them provides this flexibility.

Summary

Implementing business rules using a rule engine is an increasingly popular approach due to their ability to create a flexible, adaptable, high-performance application. In this article we discussed rules and rule engines and how to integrate the rule engine into a Java application. We also examined how business rules applications are developed using a rule engine. In the second installment of this series we'll examine in more depth the implemention of business rules using Java rule engines.

AUTHOR BIOS

Colleen McClintock, product manager for JRules at ILOG, Inc., has more than 15 years of industry experience specializing in the planning, design and implementation of rule-based systems and the integration of rules and objects.

Carole Ann Berlioz, a field sales engineer for the Telecommunications Business Unit at ILOG, Inc., has over five years of international experience in design, recommendation and development of data visualization, intelligent agents and integrated architecture components using OO technologies.

cmcclintock@ilog.com cberlioz@ilog.com

Compuware

www.compuware.com

Listing 1

```
public void correlateAlarm ( Alarm alarm )
{
    //Related Alarms
    //When a switch raises an equipment failure alarm ignore all
    //communication failure alarms on the associated port.


    if ((alarm.type == Alarm.Type.CommunicationFailure)
     && (alarm.managedObject.type ==
         NetworkElement.Type.Port))
    {
        // search for Equip failure on Switch
        AlarmList alarms = GetHistoryAlarmList();
        for (Alarm a = alarms.firstElement();
             alarms.hasMoreElements(); a = alarms.nextElement())
            if ((a.type == Alarm.Type.EquipmentFailure)
             && (alarm.managedObject.type == NetworkEle-
                 ment.Type.Switch)
             && (a.managedObject.node == alarm.managedObject)) {
                alarm.terminate(
                 "Terminated Communication Failure on Port
                 due to Equipment Failure on the Switch " +
                 alarm.managedObject.id);
                return;
            }
    }

    //Maintenance Activity Scheduled
    //When maintenance activity has been scheduled on a net-
    //work element ignore all alarms raised on this piece
    //of hardware between the maintenance start and end time.

    MaintenanceActivityList activities = GetMaintenanceActiv-
     ityList(alarm.managedObject);
    for (MaintenanceActivity maintenance =
         activities.firstElement();
         activities.hasMoreElements(); maintenance = activi-
          ties.nextElement())
        if ((alarm.time >= maintenance.startTime)
            && (alarm.time <= maintenance.endTime) {
            alarm.terminate("Terminated due to a maintenance
                            activity on "
                + alarm.managedObject.id);
            return;
        }

    // Critical Alarms on Sensitive Devices
    // When more than 15 alarms are detected on a Network Element,
    // notify the Network Operator by Pager.
    // When more than 5 alarms are detected on a Switch,
    // notify the Network Operator by Pager.
    AlarmList alarms = GetHistoryAlarmList();
    int count = 0;
    int countOnSwitch = 0;
    for (Alarm a = alarms.firstElement();
         alarms.hasMoreElements(); a = alarms.nextElement())
{
        if (alarm.managedObject == a.managedObject) {
            count++;
            if (alarm.managedObject.type ==
                NetworkElement.Type.Switch)
            countOnSwitch++;
        }
    }
    if (count > 15)
    network.pageOperator("More than 15 alarms on the NE " +
                        alarm.managedObject.id);
    else if (countOnSwitch > 5)
        network.pageOperator("More than 5 alarms on the
Switch " + alarm.managedObject.id);

} // public void correlateAlarm
```

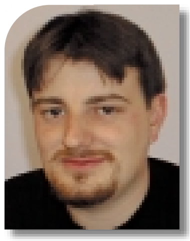
Listing 2

```
//Related Alarms
//When a switch raises an equipment failure alarm ignore all
//communication failure alarms on the associated port.

rule Port_CommunicationFailure_due_to_Switch_EquipmentFailure
{
  priority = maximum;
when
{
  ?alarm: Alarm(type == Alarm.Type.EquipmentFailure);
  ?switch: Switch() from ?alarm.managedObject;
  ?comm: Alarm(type == Alarm.Type.CommunicationFailure);
  Port(node == ?switch) from ?comm.managedObject;
}
then
{
  modify ?comm
  {
    terminate("Terminated Communication Failure on Port due
              to Equipment Failure on a Switch");
  }
}
};

//Maintenance Activity Scheduled
//When maintenance activity has been scheduled on a network
//element ignore all alarms raised on this piece of hardware
//between the maintenance start and end time.

rule Terminate_Alarms_During_Maintenance
{
  priority = high;
when
{
  ?maintenance: MaintenanceActivity();
  ?alarm: Alarm(managedObject == ?maintenance.managedObject ;
time >= ?maintenance.startTime ; time <= ?maintenance.endTime);
}
then
{
  modify ?alarm
  {
    terminate("Terminated due to a maintenance window on " +
?maintenance.managedObject.id);
  }
}
};


// Critical Alarms on Sensitive Devices
// When more than 15 alarms are detected on a Network Element,
// notify the Network Operator by Pager.
// When more than 5 alarms are detected on a Switch,
// notify the Network Operator by Pager.

rule Page_Operator_When_Critical_Alarm_On_Network_Element
{
when
{
  ?n: Network();
  ?networkElement: NetworkElement();
  collect Alarm(managedObject == ?networkElement)
      where (size() >15);
}
then
{
  ?n.pageOperator("More that 15 alarms on the NE " + ?net
                  workElement.id);
}
};

rule Page_Operator_When_Critical_Alarm_On_Switch
{
when
{
  ?n: Network();
  ?switch: Switch();
  collect Alarm(managedObject == ?switch)
      where (size() > 5);
}
then
{
  ?n.pageOperator("More that 5 alarms on the Switch" +
?switch.id);
}
};
```

Gemstone

www.gemstone.com

'You Certified?'



WRITTEN BY
ALAN WILLIAMSON

Today marks the beginning of British summer time. Hurrah! I prefer the winter myself, but having experienced the extreme temperatures of New York on a January weekend, I have to reevaluate my climate preferences. Now that the office heating has been turned down (I wonder if they'll notice?), let's get on with this month's rant and rave.

Java certification – What are your thoughts on this particular hot potato? It's been making the rounds on the mailing list again, and is interesting to listen to. I think – and the veterans of the list will correct me if I'm wrong – it was discussed about six months ago, and the attitude of the list as a whole still seems to be divided on the subject.

"Should I Get Java Certified?"

Good question, and since this is one I get asked an awful lot, let me go through some of the pros and cons of the process so you can make up your own mind. I'll include many of the views of those that are active on the **Straight Talking** mailing list, including a number of my own.

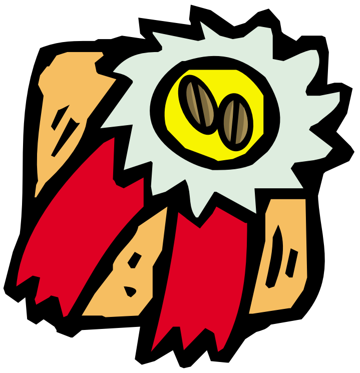
First of all, let's have a look at what getting certified actually is. There's a series of exams you can take that will test your knowledge over the whole API. Sun offers a number of different types of exams, depending on the JVM. There are two classes: Programmers Certification and a Java Architects Certification. The difference between the two is that the former takes you more into the API side of the equation while the Java Architect looks at using the components and designing systems with them.

Generally speaking, becoming Java certified isn't as expensive as it used to be, but is still beyond the budgets of many individuals, especially those just starting out. Thus they have to rely on employers to sponsor the process. This is where things become a little difficult – but let me come back to that in a moment.

What is the end goal of owning a Java certificate? What does it mean? What will it allow you to do? In theory, it's a piece of paper that affirms that you have passed a variety of tests and proved that your knowledge of Java is at a certain level. I'm reluctant to say that it makes you a Java developer, as experience has taught me otherwise. I and others have commented that they've seen many Java-certified people who weren't much on the development front. They knew enough to get through the exam, but when they attempted to use their knowledge in practice it was soon evident that their knowledge of Java didn't run as deep as the certificate would have led them to believe.

Surely some knowledge is better than no knowledge, you may say. Granted. However, it gets a little tricky when a company makes a decision on hiring one of two people purely on the basis of whether they've been Java certified. As we know, merely having the certificate does not a developer make.

With that in mind, we'll first go through the advantages that obtaining the certified status should yield. One of the most important pluses is that it does no harm to your curriculum vitae. As we know, the marketplace is getting tougher, and although there's a shortage of Java skills, positions are still hotly fought for, so the more ammunition you have in your arsenal, the better. Going through the process also allows you to fill in some of the gaps in your understanding. For example, you may know that something works, but you're not quite sure why or how – you've just been taking it on faith that it does work.



What are the disadvantages of Java certification? Well, to be honest, except for time and money there aren't any real reasons not to become Java certified. At the end of the day it's another selling point for you to use when you need it.

But should you take the tests? Well, that's a question only you can answer. For example, I'm not Java certified and I don't need a piece of paper to tell people that I know Java. Many of you will be the same. You've been coding for most of your life, and Java is merely another string you've added to your bow – you're confident in your own ability. But for someone fresh out of university or new to developing, I'd say it would do no harm to become Java certified. That said, it all depends on the job. If you're currently employed as a Java developer, I'd say there's not much point in going for the certification; it won't add any extra to your salary. But if you're planning on leaving or moving on and feel your Java is a little on the weak side, take some time out and go for it.

This is one of the catch-22s facing employers. Should they pay for their employees to become Java certified? Chances are, employers are merely training their staff for some other company's benefit. This is one of the main problems with training as a whole. But certification in the majority of cases will only be a piece of paper stating that employees can do the job they've already been doing successfully. So what's the point?

Let's pull some of this information together and make it a little clearer. There are two main reasons for certification: (1) to come up to a level of technical competency, and (2) to make yourself look more attractive to potential employers. Fortunately if you're looking to compete on the first one, there are many resources, both on and offline, that will help you. The following links will take you to a number of exam questions that will allow you to gauge your worth: http://home-1.world-online.nl/~bmc88/java/javacert/index.html and http://suned.sun.com/usa/cert_test.html?content=scajt_quest.

I'd like to thank Dan Dobrin and Nathan Johnson for supplying these links. If anyone has more, please post them to the mailing list for all of us to check out.There are also a number of books from the major publishers on typical exam questions but a number of **Straight Talking** respondents claim that many of them are riddled with mistakes. So use caution.

The second reason is to make you look more attractive to future employers. I strongly recommend that you take a long hard look at your résumé and see whether it needs it or not. In some cases it will, in some cases it won't. It's your call.

Segue Software

www.segue.com

I'd love to hear from both developers and employers on this subject. Jump on our list and let us know your thoughts.

Support Award of the Month: Hall of Shame

I'm making some headway on the mission to clean up our industry's support and I'm still researching a number of big names. As soon as I have more information, you'll be the first to know.

I'd like to give one particular online merchant a small pat on the back for a job well done. Fortunately, I've been given permission by my esteemed radio cohost, Keith Douglas, to recount this somewhat amusing anecdote. Keith was on a mission to purchase a book on the life of Paul Simon of Simon & Garfunkel fame – well, his life so far at least, since the poor man isn't dead just yet. Keith went shopping on W.H. Smith's Web site and quickly located the desired paperback. He exchanged the necessary credit-card information and was assured the book would be dispatched as soon as possible. Indeed, a couple of days later said book did arrive. It was a biography of one Paul Simon, but not of Simon & Garfunkel fame, more of state senate fame. Keith isn't sure where the mixup occurred, with him or at W.H. Smith. Nevertheless, the book was quickly returned and a refund was made. So beware the next time you order…make sure it's the one you really want.

Mailing List

The **Straight Talking** mailing list is one of those lists that gives you the opportunity to talk to a lot of people in all parts of the industry. We have a number of authors on the list, including a number of hard-core techies who are shaping the very APIs and virtual machines we depend on so much. So sign up at http://listserv.n-ary.com/mailman/listinfo/straight_talking.

In addition to the mailing list, our radio show still broadcasts daily at http://radio.sys-con.com/. Keith Douglas and I present a 15–20 minute daily music/talk/Java **Straight Talking** show. A treat in itself!

Salute of the Month

Over the last six months I've been reviewing each of the big Java IDEs for a forthcoming review article here in ***JDJ***. It's been an interesting journey, one I haven't completed yet. At the moment my machine has CodeWarrior on it from Metrowerks. I had some teething problems with it and jumped onto the newsgroups late one UK night to see if I could get some immediate help. It was here I happened on a name from Metrowerks, Ron Liechty, who answered one or two queries. I e-mailed him, explaining my problem, and to my delight and surprise had an answer within 30 minutes of my original post. Since then Ron and I have traded a number of e-mails in which we've discussed many aspects of the IDE world. I'd like to thank Ron for taking the time to bounce around a number of ideas with me – I've thoroughly enjoyed our dialog.

A piece of useless information for you: this is the twenty-third installment of **Straight Talking**. Next month is our two-year anniversary! Do you believe that? Where has the time gone?

This month I took delivery of what has to be the coolest mouse I've ever seen. Purely by accident I highlighted the wrong item in a list of mice and ended up with the Microsoft Explorer Intelli-eye mouse. It's a nice enough mouse, but the feature that's really cool is the red light that shines at the back of it. It looks like a brake light, and is fantastic at night. Yeah, Yeah, I know: I need to get out more!

See you next month.

AUTHOR BIO

Alan Williamson is CEO of n-ary consulting Ltd., the first pure Java company in the United Kingdom. The firm, which specializes solely in Java at the server side, has offices in Scotland, England and Australia. Alan is the author of two Java servlet books, and contributed to the Servlet API. He has a Web site at www.n-ary.com.

alan@sys-con.com

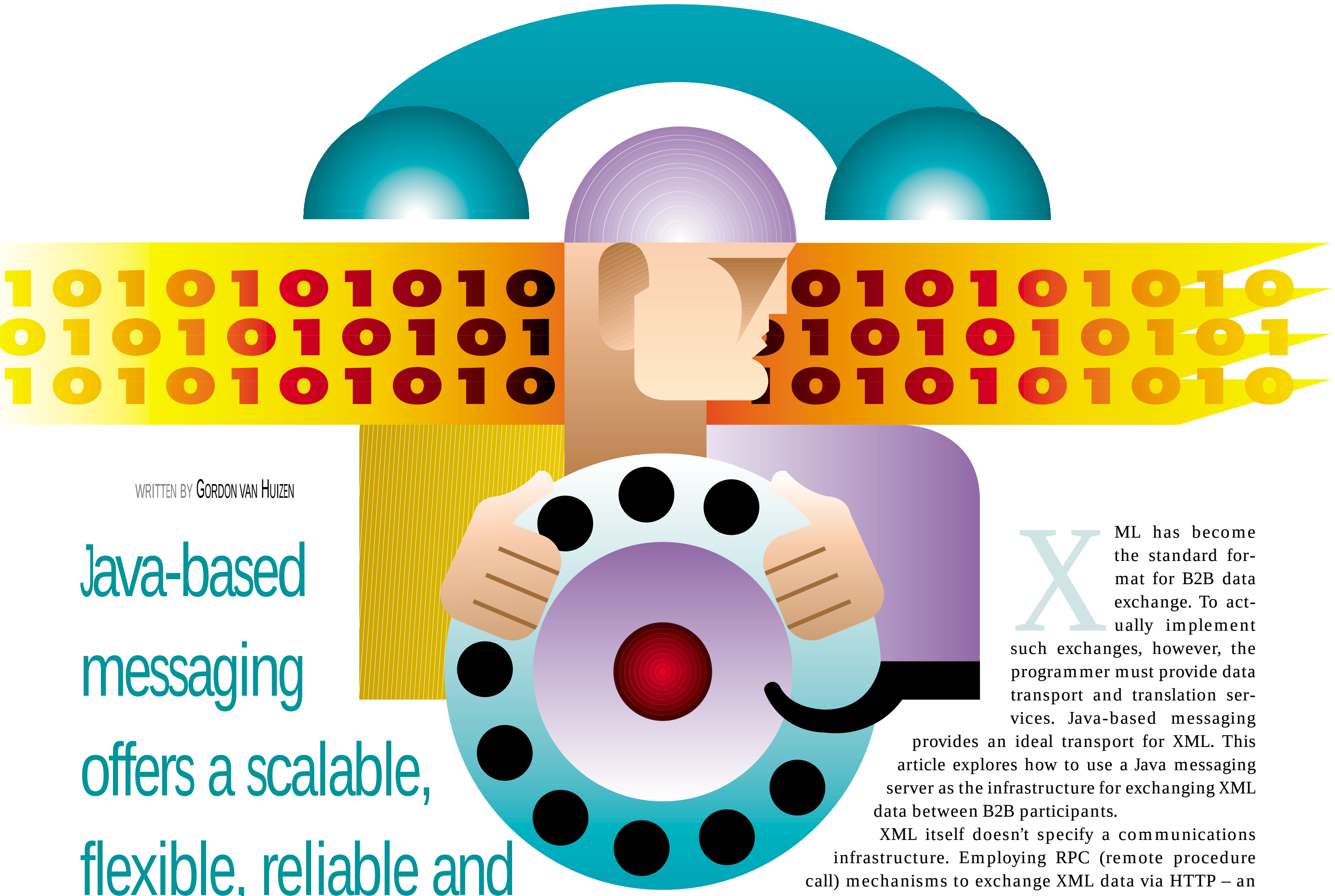
Inetsoft Technology Group

www.inetsoftcorp.com

Unify

www.ewavecommerce.com

Using XML for JMS Messaging in E-Commerce



WRITTEN BY GORDON VAN HUIZEN

Java-based messaging offers a scalable, flexible, reliable and secure communication channel within and between enterprises

XML has become the standard format for B2B data exchange. To actually implement such exchanges, however, the programmer must provide data transport and translation services. Java-based messaging provides an ideal transport for XML. This article explores how to use a Java messaging server as the infrastructure for exchanging XML data between B2B participants.

XML itself doesn't specify a communications infrastructure. Employing RPC (remote procedure call) mechanisms to exchange XML data via HTTP – an approach advocated by some – suffices for rudimentary applications but doesn't scale for distributed applications across multiple systems, such as in business-to-business e-commerce. In such systems parties are often unreachable, yet reliability of communications is crucial.

Take the relatively simple example of communicating price changes. To ensure that each disconnected party eventually receives the data, persistence and verification must be in place. RPC mechanisms don't provide persistence and verification, so the programmer must build these services into the application logic – no easy task! The burden becomes greater when the requirement is to communicate a message such as a bid request either to all parties or none. The programmer would have to provide rollback mechanisms similar to those found in database transactions.

KL Group

www.klgroup.com/interface

JavaSoft's Java Message Service (JMS) specification offers a way to meet these needs. JMS comprises an API and semantics for a middleware messaging service that helps insulate the application from the issues posed by loosely coupled systems by providing services such as persistence, verification and transaction support.

Messaging middleware also improves the flexibility of e-commerce systems. Consider the example of a wireless phone company's marketing department communicating new rate plans to its franchised dealers every month. JMS-based messaging allows the company to provide value beyond simply publishing new rate plan information reliably and on a timely basis. By using a messaging middleware system, the company can decouple use of the rate plan data from the initial application that broadcasts it. Other departments in the company can use the data to create reports or communicate the data to third parties, such as regulatory agencies, with no need to modify the original application.

JMS provides mechanisms to implement virtually any form of e-commerce transaction. In this article I'll use the example of a B2B portal to illustrate how these mechanisms can be applied. The B2B portal will act as the hub of an online trading community, connecting buyers and sellers. The buyers broadcast their interest in obtaining goods and services to the community at large, and sellers selectively bid on these opportunities.

JMS Implementations

Programmers may elect to build their own Java messaging middleware from the JMS specification or obtain this functionality from a third party. Commercial JMS implementations are provided as stand-alone messaging servers or as JMS services embedded within other environments such as application servers.

A stand-alone server provides some advantages over an embedded messaging service by allowing the developer to:

- Place messaging broker(s) where needed to optimize network topology, control traffic loads, provide fail-safe operation and/or leverage existing hardware.
- Embed the messaging server infrastructure within a value-added application or operating platform.
- Be assured that enhancements of, and extensions to, the messaging server aren't tied to those of the environment in which the messaging server is embedded.
- Choose, if used with an application server, which app server(s) to pair with the messaging server.

On the other hand, having the messaging service built into another environment offers the convenience of a single (if less flexible) solution, especially for organizations that have standardized on an application server and anticipate that their needs will not change.

Persistence

Perhaps the greatest benefit of a fully featured JMS implementation is resilience, often referred to as "persistent" or "guaranteed" messaging. Typically this means:

- A sender specifies that a message persists within a broker-managed data store until all subscribed recipients receive it.
- Subscriptions are durable, meaning a client will automatically receive messages sent to the subscribed topics whenever the client attaches to the message broker, including any persistent messages sent when the client was disconnected.

It's important to note that the JMS specification doesn't prescribe how persistence should be implemented. Those specifying or building a JMS system should ensure that: (1) they know how the concepts of persistent and guaranteed messaging are defined in the context of the system that they're considering; and (2) they know how these concepts will be implemented so that their promise is met.

Advanced Functionality

The JMS spec leaves it to the vendor or programmer providing a JMS implementation to add additional functionality necessary for large-scale applications such as:

- *Support for XML-formatted messages*
- *Hierarchical name spaces*
- *Clustered brokers*
- *Graphical administration tools*
- *Extensions to other environments, such as ActiveX*

JMS and XML

In JMS an XML transaction is contained in the body of a message. This message is then broadcast to a number of potential recipients or sent to one recipient in particular. The actual transmission of the message is typically performed over TCP/IP or HTTP, although the JMS implementation shields the application code from the specifics of the wire protocol.

Routing of the message is based on fields in the message header and the manner in which recipients are subscribed to the messaging system. Recipients control which messages they receive by subscribing to specific message topics (described below). Additionally, recipients can request that the messaging system filter messages on their behalf by specifying filter criteria in a SQL-like syntax. For example, a supplier in our online trading community could register an interest in bid requests with a delivery time of 10 days or more by specifying a selector of "Property_DeliveryTime >= 10".

An XML application can use this mechanism to provide content-based routing by programmatically placing appropriate XML data within the message properties. The JMS spec currently doesn't allow the messaging system direct access to the message body, which would be required for XML content-based routing without application intervention.

How JMS Works

The JMS specification describes a JMS provider that implements the APIs and semantics. The most common implementation model for a JMS provider is a Java messaging broker, either a single broker or – in some implementations – multiple brokers. The broker is the heart of the messaging service, routing all messages through its hub (see Figure 1). Multiple brokers – for those JMS implementations that offer this feature – may be clustered, facilitating scalability and offering a higher level of fault tolerance. Clustered brokers also allow for centralized maintenance of security functions.

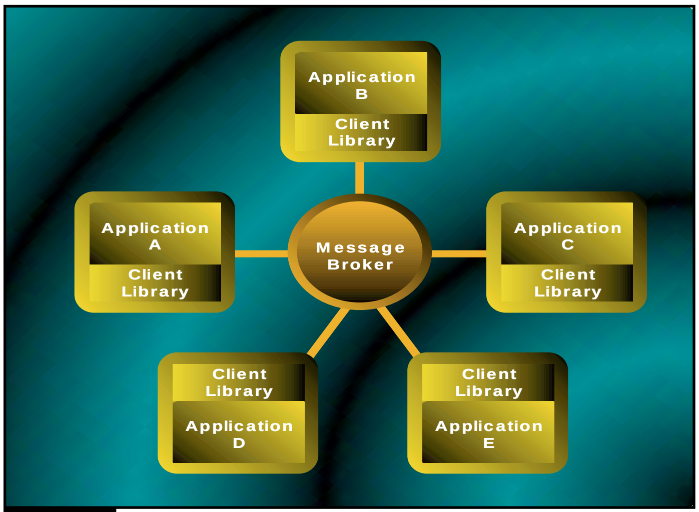


FIGURE 1 Hub-and-spoke JMS topology

Information Architects

www.ia.com

The JMS spec calls for two types of messaging:

- *Publish/subscribe* (or "pub/sub"), a one-to-many model where a client application publishes messages to topics subscribed to by other clients
- *Point-to-point*, where a client sends messages to a queue from which one or more clients obtain messages sequentially

The programming model for JMS-based applications using these two models is virtually identical. In fact, many JMS-based applications use a mix of the two models. Both pub/sub and point-to-point use the same format of JMS message. There are three sections in a JMS message:

1. *Header fields*
2. *Properties*, used for message filtering and routing
3. *Body*, the application-specific portion of the message

While the logical routing of pub/sub messages is quite different from that of point-to-point, both models share a hub-and-spoke architecture between clients and the provider. Clients access the provider through the JMS API. Clients call a messaging client runtime library that implements the JMS interface and communicates with the broker.

To begin messaging, a client opens a connection to the message broker. The client locates a broker through JNDI and establishes a connection through a ConnectionFactory. Within the context of a connection, the client establishes one or more sessions. The process of sending and receiving messages is performed via the session, which holds the transactional and acknowledgment characteristics for messages to be sent or received. It's more efficient to employ multiple sessions per connection than multiple connections per broker. Connections require more resources to establish and maintain than sessions and can easily be reused for communication through different topics or queues.

JMS supports acknowledgment for both sessions and clients. Like handshaking in a low-level communications protocol, acknowledgment informs the message broker when a client has received a message. JMS also provides a ReplyTo mechanism. A message publisher can use ReplyTo to indicate where a receiver of the message should send replies. Unlike acknowledgments, replies are sent as full JMS messages.

JMS also supports transactions. When a session is transacted, the message broker stages the message traffic until the client application either commits or rolls back the transaction. When a rollback is performed in a session that is both a producer and a consumer, its produced messages are destroyed and its consumed messages are re-sent.

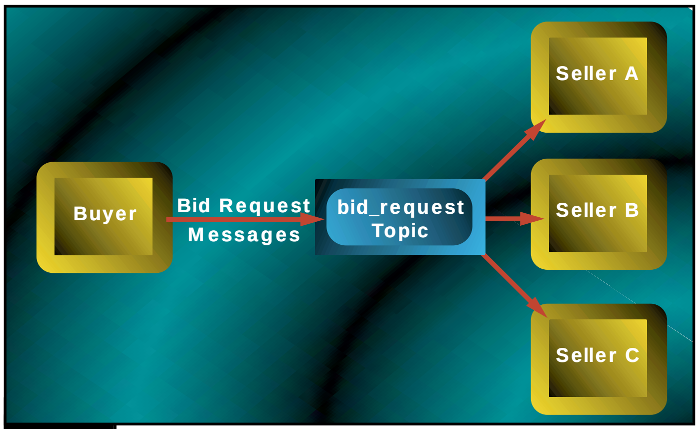


FIGURE 2 Pub/Sub example

"Some JMS implementations allow topics to be defined in a hierarchical structure so that topics can have associated subtopics"

PUBLISH AND SUBSCRIBE

The pub/sub paradigm promotes the independence of producers and consumers from one another and allows for one-to-many and many-to-many configurations.

In pub/sub, each session object publishes and/or subscribes to one or more topics. An authorized publisher produces messages through a specified topic, and authorized subscribers receive messages by subscribing to that topic. Topics may be static or dynamic objects and can be temporary for transitory or anonymous uses.

Some JMS implementations allow topics to be defined in a hierarchical structure so that topics can have associated subtopics. Clients can subscribe to the appropriate topic level to receive the most relevant information. This technique provides a more granular approach to topic subscriptions, substantially reducing the amount of coding needed to filter unwanted messages while supporting highly flexible subject-based routing and security.

In our online trading community example, each buyer publishes messages to a bid_request topic, using the pub/sub model (see Figure 2). The buyer uses the pub/sub model, as each buyer wants to inform all interested sellers of the bid request. In a large community that serves a colorful array of buyers and sellers, we may want to break the bid_request topic into several subtopics, with one subtopic for each category of product or service. These subtopics could further be divided into more granular categories. For example, a given topic hierarchy might be bid_request.vehicles.bicycles.

Each seller subscribes to the bid_request topics that it's interested in. Further, the seller can request that the portal forward only those bid requests that adhere to specified criteria. For example, a seller could subscribe to the bid_request.vehicles.bicycles topic, but indicate that it wants to receive bid requests only for mountain bikes or only when the buyer doesn't require overnight shipment. These filters would be created using the message selector mechanism discussed earlier:

```
"Property_MerchType = 'Mountain Bike' AND Property_ReqOvernight is
NULL".
```

Quality of Service

In publishing a message, the publishing application specifies the quality of service – e.g., delivery mode, time-to-live, priority – and whether the subscriber requests a reply.

The delivery mode can be persistent, where the message is placed in a data store as part of the send operation for a "once-and-only-once" guarantee in which the message must not be delivered more than once and must not be lost.

The time-to-live parameter specifies how long the message broker should retain the message to ensure that all subscribers receive it. If the time-to-live is specified as zero, the message won't expire.

More efficient is the *nonpersistent* delivery mode, in which the provider delivers the message with an "at-most-once" guarantee. The message won't be delivered more than once but isn't protected against loss.

A durable subscription indicates that the client wants to receive all the messages published to a topic even if the client connection isn't active. The broker ensures that all messages published to the topic are retained until the durable subscriber acknowledges them or the messages have expired. For example, a seller in our trading community could go offline for an extended period and still receive bid requests that were broadcast during that time.

Information
Architects

www.informationarchitects.com

POINT-TO-POINT

In point-to-point (PTP) each session object sends and/or receives through one or more queues. Consumers can either receive the message that's first in line – thus removing it from the queue – or browse through all the messages in the queue, causing no changes.

The first message received by the broker is the first message delivered to a consumer. This requires that the broker retain the second and subsequent messages until the first is consumed. Even when no clients are associated with a queue, messages are retained. Durability and persistence are implicit in PTP, unlike in pub/sub. There's only one message consumer for a given message. When each message is acknowledged as delivered, the broker discards it.

Let's refer again to our fictitious trading community to explore using a point-to-point queue. Each buyer in the community creates a queue for receiving bids from interested sellers. We use a point-to-point queue here because in each case only one party needs to receive the bid messages. The buyer can point to this queue from within its bid request messages, using the ReplyTo mechanism discussed earlier, making it easy for interested sellers to submit bids. The system can be made anonymous, that is, bidders don't even know the identity of the requestors until the end of the bidding process (see Figure 3).

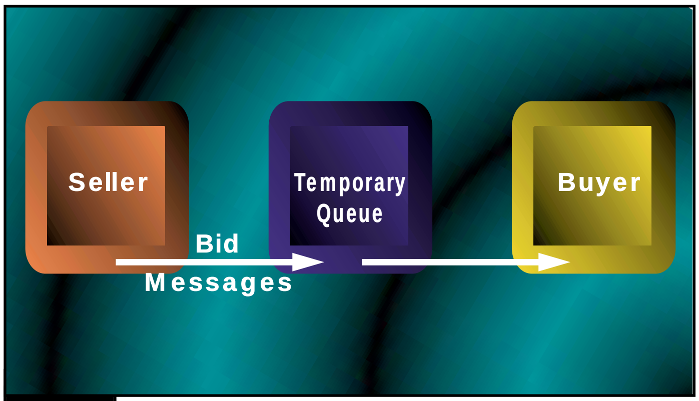


FIGURE 3 Point-to-Point example

Summary

We've examined the key capabilities and concepts of JMS, and looked at an e-commerce scenario that uses the two JMS messaging models: pub/sub and point-to-point. XML applications built using JMS are afforded a great deal of flexibility in how they send and receive messages, as well as how messages are routed within the system. Although JMS doesn't yet support content-based routing directly from XML tags, an application programmer can easily add this functionality by mapping XML tags to JMS message properties.

XML can greatly benefit from a transport that supports the loosely coupled, dynamic environments inherent in the large-scale heterogeneous systems used for B2B e-commerce. JMS provides such a transport and offers the required persistence, verification and transactional capabilities required in a highly flexible programming model. Those considering a JMS solution should look for functionality beyond the basic JMS spec that's necessary for deploying and managing a scalable system.

AUTHOR BIO

Gordon van Huizen has designed and developed solutions for Web site generation and distributed electronic forms systems, network-based object licensing and real-time data merging and imaging. As director of product strategy for Progress Software, he has been responsible for systems engineering of Progress' JMS-based messaging server, Progress SonicMQ, and is currently responsible for the definition of the company's next-generation messaging strategies.

gvanhuiz@progress.com

Buzzeo

www.buzzeo.com

Getting Ourselves on the Applications Fast Track

How Lincoln Re used a new architecture and Java to transform IT department from cost center to business partner

WRITTEN BY
MATTHEW BROWN

The convergence of information technology (IT) and business strategy has become an increasingly critical competitive factor for most businesses. Lincoln Re came to this realization several years ago when we embarked on transforming our IT systems architecture from a mainframe-based environment to a distributed client/server infrastructure.

Our goal in IT was to enable more flexible and faster paced development of "database-aware" and browser-interfaced applications that could be accessed by employees throughout the enterprise and potentially by clients via the Internet.

Three years later the transformation continues with much new groundwork laid. One enterprise application has been rolled out that makes it easier for Lincoln Re employees to access information about our clients. Meanwhile, more enterprise-wide and department applications have been completed or are in development.

A New Mission Statement

The transformation began in 1996 when then new Lincoln Re CIO Linda Fraley redefined the mission of the IT department in the potent phrase, "Establish the systems area as a core competency for the department." In other words, she sought to transform IT from a cost center and service unit to a business partner for both Lincoln Re and our parent company, Lincoln Financial Group, or LFG. (LFG is the marketing name for Lincoln National Corporation and its affiliates, just as Lincoln Re is the marketing name for the reinsurance companies of Lincoln Financial Group. Lincoln National Corporation is a financial services holding company with consolidated assets of $103 billion and annual consolidated revenues of $6.8 billion.)

That same year, Fraley called for an assessment of the Lincoln Re IT department's readiness to move from a mainframe-based development environment to one based on Sybase's PowerBuilder.

An IT Architecture That Can Drive Business

A critical step toward embracing the new IT mission was to identify and establish an enterprise architecture upon which Lincoln Re could build its current and future business. Fraley proposed a large, scalable architecture project to be built over three years. After securing CEO Larry Rowland's endorsement, the project began in January 1997.

Step one involved assembling an architecture team whose members underwent extensive training and mentoring in the development of enterprise architecture. These sessions produced a comprehensive report that included two parallel findings:

- The current state of Lincoln Re's business and its future direction
- The current state of Lincoln Re's IT department and what the department needed to do to align with Lincoln Re's strategic business objectives

Among the key business objectives was to improve customer service by making it easier for clients to do business with Lincoln Re. To reach the objective in the IT realm, the department needed a highly productive environment to develop scalable, interoperable systems for both the enterprise and workgroups. We also needed a new database that would serve as a single central location for maintenance and administration of the life insurance policies Lincoln Re reinsures. And, like the system itself, the database needed to be accessible enterprise-wide by means of a browser-user interface and be accessible to clients who would log on via the Internet.

Systems inoperability and a Web-centric environment led inevitably to certain choices in foundation technologies, including Java, CORBA (Common Object Request Broker Architecture) and MOM (Message Oriented Middleware).

VisualCafé Stands Out as Best-of-Breed

The next step: choosing the development environment. The Lincoln Re development team evaluated the leading Java integrated development environments (IDEs), including PowerJ from Sybase, Visual J++ from Microsoft, VisualAge for Java from IBM, JBuilder from Borland, JDesignerPro from BulletProof and VisualCafé from WebGain, Inc. (formerly from Symantec Internet Tools).

Our requirements were:

- Robustness of Java GUI development capabilities needed to be determined
- Easy-to-use IDE needed to convert a COBOL shop to a Web-based Java shop
- RAD (rapid application development) capabilities to enable developers to be productive while learning Java
- Cost effective
- Market position
- Strength of the company and its commitment to the language

The team chose VisualCafé Professional Edition. VisualCafé was clearly the leader of the pack, and we felt that Symantec would continue to enhance and support the Java language. The RAD environment allowed our developers to create database-aware applications by pointing and clicking the mouse, ensuring a highly productive development environment, even for beginning Java developers.

Currently, about 10 developers are writing Java code using VisualCafé, working on Microsoft NT workstations linked by Token Ring and Ethernet networks.

Access for Employees and Clients

The first application in production was a client information system called

New Atlanta

www.servletexec.com

"LegacyLinc." A four-tier, CORBA-based application, LegacyLinc provides browser-based access to multiple, disparate legacy databases that operate off MVS and Lincoln Re's LAN-based DBMS, Microsoft's SQL Server.

Previously, the legacy systems used multiple CICS-based – Customer Information Control System – interfaces for maintenance. Through browsers, LegacyLinc now provides both structured (database) access for employee maintenance and unstructured (document-based) access for intranet-connected clients. The LegacyLinc runtime environment includes:

- MS Windows NT 4.0 workstations equipped with Netscape 4.7 browsers
- MS Windows NT 4.0 Web servers equipped with Netscape Enterprise Server and MS IIS
- Microsoft Windows NT 4.0 running on in-house–developed application servers using Inprise's Visibroker ORB (Object Request Broker), which resides on the Web server, and JDBC Level Four drivers from BEA WebLogic
- MS SQL Server 6.5 database management system running Windows NT 4.0 systems
- Legacy databases – CICS applications hosted on IBM mainframes

LegacyLinc contains information drawn from the legacy databases about clients and their insurance policies, key client contact names and numbers, and information about the account team members. Built primarily to support the sales and marketing activities of account management teams, the system has been made available to about 800 cross-functional employees to enable them to access information and contribute to it, as appropriate. The unstructured document data available to Lincoln Re employees includes, for example, Trip Reports – descriptions of site visits to clients – in the form of Word documents published to Lincoln Re's Document Management System.

The development process for this enterprise application took an IT team of 10 to 12 developers about a year to complete, from start to finish. VisualCafé was used to develop the user interface, and to develop and compile the business rules and CORBA objects.

Ensuring the Future

Another enterprise-level application – RMI-based (Remote Method Invocation) instead of CORBA – has been moved to production and continues in development. This application will interface to an Informix database running on an IBM AIX RS6000 and several smaller-scale, two-tier workgroup applications being put together by smaller teams of two to four developers. These workgroup-app developers are using the RAD capabilities of VisualCafé, including its database wizards.

As in all development, debugging is a necessary evil. Distributed debuggers play a big role in the ability of an IT organization to develop systems effectively and efficiently. We're looking at the VisualCafé Enterprise Edition product as a way to provide those capabilities to our developers.

We're also developing workgroup-level solutions that are simpler in complexity and lower in scalability requirements. The current 3.0 version of VisualCafé offers us the right toolset for this type of job; we believe we don't always need to hunt with an elephant gun, and expect to use both types of tools in our environment.

The capability to develop distributed objects easily is a real advantage to an IT group – like Lincoln Re's – that needs to produce distributed systems.

Author Bio

Matthew Brown is an assistant vice president and director of research and architecture at Lincoln Re.

MJBrown@LNC.com

Tidestone
Technologies

www.tidestone.com

SQL Embedded in Java

Part 1 Starting Out

Introducing SQLJ, the standard for embedding database SQL statements in Java programs

WRITTEN BY EKKEHARD ROHWEDDER

If you know SQL and Java, and you want to learn SQLJ, this series of articles is for you! I'm going to introduce SQLJ, the standard for embedding database SQL statements in Java programs.

You may be familiar with JDBC, an API for talking SQL with the database. SQLJ takes many ideas from JDBC further by providing a language interface for SQL statements in Java. This allows programmers to concentrate on what to say, not how to say it. Take the following SQLJ statement (here aName is a Java variable of type String, and emp is a database table with a numeric column sal and a character column ename):

```
#sql { UPDATE emp SET sal = sal * 1.1 WHERE ename = :aName };
```

If you're using JDBC, you'd have to write something along the following lines:

```
PreparedStatement st =
 conn.prepareStatement("UPDATE emp SET sal = 3000 WHERE ename = ? ");
st.setString(1, aName);
st.executeUpdate();
st.close();
```

Brevity isn't the only benefit of SQLJ. An equally important feature is that you can check SQL statements against the database at translate time rather than at runtime, when a particular SQL statement may or may not get executed. Last but not least, since the SQL text of SQLJ programs is known ahead of time, it permits database vendors to use all of their compile-time performance tricks – such as compilation into the database.

Elixir Technologies

www.elixirtech.com/dwnload/

You're probably thinking, however, that with all of these goodies there must be limitations. Actually, there's only one: SQLJ supports only static SQL statements. The names, shapes and types of SQL tables, views and procedures must be known and fixed ahead of time. The actual data sent to or received from the database is, of course, not fixed. In the example above the bind variable aName can take on any string value. Note that you couldn't use this variable to specify a table name, as in the following:

```
#sql { UPDATE :aName SET sal = sal * 1.1 };  // BAD! Not a static
SQL statement.
```

So how can you write dynamic SQL programs if you have to? That's easy: just use JDBC. SQLJ actually provides nice interoperability features with JDBC.

Now that I've lured you into reading about SQLJ, here are the topics I'll discuss in the remainder of the article:

- The shape of SQLJ programs
- An introduction to the SQLJ translator
- Using Java host expressions in SQLJ statements
- An introduction to iterators – the strongly typed SQLJ counterpart to JDBC result sets

I'll also sprinkle in a few exercises to keep you busy until next month. Throughout the article, by the way, you'll find important tips. Even if you don't read anything else, read – and heed – these tips! You'll be glad you did and will save yourself time and frustration. Oh, and if you're wondering how to obtain and install SQLJ, check out the first sidebar.

Skeleton of an SQLJ Program

In this section I'll take a look at the essential components that every SQLJ program needs – in other words, the skeleton. One of the first lines in your SQLJ program will be:

```
import java.sql.SQLException;
```

INSTALLING SQLJ

SQLJ works on top of any JDBC driver

STEP 1: Get Java!
Install the Java Development Kit 1.1 or later (you can get it from http://java.sun.com).

STEP 2: Get JDBC!
Get a JDBC driver for your database. For Oracle, you can download JDBC drivers from http://technet.oracle.com. Or you can get the JDBC–ODBC bridge from http://java.sun.com. Install the JDBC driver's classfiles in your CLASSPATH.

STEP 3: Get SQLJ!
Download Oracle SQLJ from http://technet.oracle.com or the SQLJ Reference Implementation from www.sqlj.org (under Implementations-Oracle) or IBM DB2 SQLJ from www.softwareibm.com/data/db2/java/sqlj/ or the Informix JDBC 2.x drivers at www.intraware.com/informix (which bundle SQLJ).

The Oracle and Reference Implementation distributions result – when unpacked – in a directory hierarchy under a directory sqlj. The SQLJ executable (named sqlj or sqlj.exe depending on your flavor of operating system) lives in sqlj/bin, which should be in your PATH. You must also put sqlj/lib/translator.zip – which contains the SQLJ translator – in your CLASSPATH.

STEP 4: Run it!
Are you greeted with a help screen when you say sqlj? Yes? Then you're in business!

Tip: Make sure that you have a JDK and your JDBC drivers properly installed before starting the SQLJ installation. Also, if you have several versions of Java or Java Development Environments installed, I recommend that you "build up" your PATH and your CLASSPATH environment from scratch to make sure you properly pick up a known Java and JDBC configuration. On NT, consider creating a .bat file that you can use in a DOS window to provide the appropriate setup.

SUMMARY OF SQLJ SYNTAX AND USAGE

• DATABASE CONNECTION

```
 import sqlj.runtime.ref.DefaultContext;
 DefaultContext.setDefaultContext(new DefaultContext(url, user,
password, auto-commit));
…
 DefaultContext.getDefaultContext().close();
```

• STATEMENTS

```
 #sql { … :hostVariable … :(host expression) … };
```

• ITERATORS

```
#sql modifiers iterator iterator-name(type1 name1, …, typen
namen);
…
iterator-name iter;
#sql iter = { SELECT … };
while (iter.next()) {
    … iter.name1() … iter.namen() …
}
iter.close();
```

If something goes wrong while you're running your SQLJ program, your SQLJ statements and any methods in the SQLJ runtime API throw a SQLException. Either declare that your program throws an SQLException, or put

```
try { ... } catch (SQLException exn) { ... }
```

blocks in your program.

What good is an SQL program without a database connection? (See the second sidebar for a summary of SQLJ syntax and usage.) Another important import line is the following:

```
import sqlj.runtime.ref.DefaultContext;
```

Before executing an SQLJ statement you had better connect to the database.

```
new oracle.jdbc.driver.OracleDriver();
DefaultContext.setDefaultContext
        (new DefaultContext("jdbc:oracle:oci8:@", "scott",
"tiger", false));
```

The first line creates an instance of your JDBC driver and – as a desired side effect – registers that driver with the JDBC DriverManager. Of course, if you don't use an Oracle JDBC driver, you'd use a different class name here. The second statement sets the SQLJ default connection. Your username – equivalent to the database schema you're connecting to – is "scott" and your password "tiger." The first argument to the DefaultContext() constructor is the JDBC URL. If you want to connect to a different database and/or through a different JDBC driver, you need to adjust the URL accordingly. Refer to your JDBC driver documentation for specifics. The last argument is the auto-commit flag. For serious database work you want to turn auto-commit off. Only at the end of the day will you decide whether to commit or roll back your work. How do you do that? With #sql {COMMIT}; or #sql {ROLLBACK}; of course!

It's also good style to close your connection context (by now it's rather obvious that's what we call connections in SQLJ) rather than leave the cleanup to your JVM's finalization. You can accomplish this with:

```
DefaultContext.getDefaultContext().close();
```

Are you still with me? Now pull all the pieces together into a file HiScotty.sqlj (see Listing 1).

PointBase

www.pointbase.com/jdj

***Tip:** Every SQLJ statement must be terminated with a semicolon ";"*

EXERCISES

- What happens if you omit the semicolon ";" at the end of the SQLJ statement (after the closing curly brace)?
- What happens if you put a semicolon ";" at the end of the SQL statement (just before the closing curly brace)?

Meet Your Translator

Now you've got a file with a bunch of text in it – nothing to write home about. How do you bring this to life? One small step for you, one big leap for the translator:

```
sqlj HiScotty.sqlj
```

Yes, this translates your SQLJ source to a Java source and compiles it in the same fell swoop. This should – if everything goes okay – create some *.class files (and a *.ser file), and you can then issue:

```
java HiScotty
```

Even though you're familiar with .class files – the result of Java compilation – you'll be curious about these .ser files that the SQLJ translator produces. We also call them *(serialized) profiles*. They're serialized Java objects that contain all the information about the static SQL statements in your .sqlj source files, such as the SQL source text, the types and names of the host variables that occur in the SQL statement and what kind of SQL statement this is (a commit/rollback, a query, a DML statement and so on).

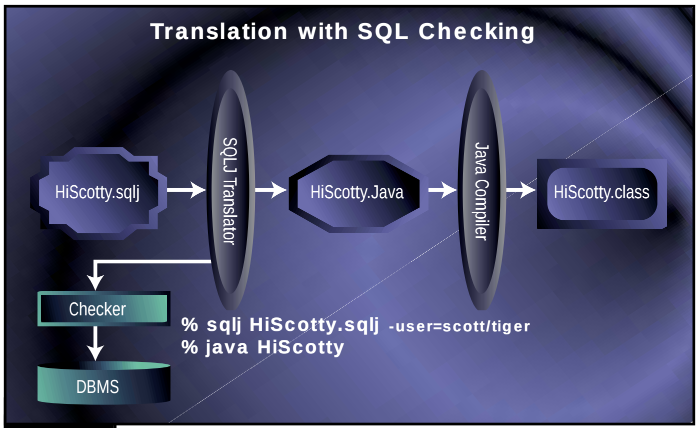
Without a database the SQLJ translator can perform only offline checking of your SQL code. If you want to get your database involved, that is, if you want SQLJ to perform online checking, then you must tell the translator how to connect to it (see Figure 1).

Specifically, you must supply a username (corresponding to the database schema you want to connect to) and a password:

```
sqlj -user=scott/tiger HiScotty.sqlj
```

Of course, you also want to be able to say which database you'd like to talk to and how – that is, using which driver and protocol. Because SQLJ uses JDBC underneath, this is accomplished by – yes, you guessed it! – a JDBC URL. Depending on the version of your SQLJ translator, this may already be set up (Oracle SQLJ, for example, uses "jdbc:oracle:oci8:@"), or you can provide it on the SQLJ command line with the -url= option. For example, you can use Oracle's thin (Type IV) JDBC driver as follows:

```
sqlj -user=scott/tiger
     -url=jdbc:oracle:thin:@my_host:1521:my_oracle_sid
HiScotty.sqlj
```



```
% sqlj HiScotty.sqlj -user=scott/tiger
% java HiScotty
```

FIGURE 1 Translation with SQL checking

The -user and -url flags are two of the 46 or so option flags that SQLJ accepts. Issue sqlj -help to get an introduction to the most important ones.

***Tip:** The translator will try to check your SQLJ programs against the database if, and only if, you specify the -user option.*

EXERCISES

- The translator will try to check your SQLJ programs against the database if, and only if, you specify the -user option.
- When run offline, SQLJ checks the legality of Java types used in SQL statements, and some (rather superficial) syntax. Show some errors caught by the translator offline.
- When run online, SQLJ additionally checks the shape of SELECTs and the type compatibility between SQL and Java, and – if you use the Oracle SQLJ translator – it asks the database to parse SQL DML statements. Show some errors caught by the translator online, but not offline.
- Show some errors caught only at runtime, but not at translation time.

  Exercise: In the examples above, the password was given on the command line. Usually, you want to avoid doing this:

  1. What happens if you omit the password in the -user option?
  2. You can use the sqlj.properties file for storing command line options used for sqlj invocation. Investigate the format of this file, and store the password information in it. What happens if an option is given in both the command line and the sqlj.properties file?

Cooler Than Host Variables: Host Expressions

As we already saw, host variables are Java variables prefixed with ":", placed inside the SQL statement, that can retrieve and/or send data values:

```
String aName  = "SCOTT";
Double raise = new Double(1.08);
Double salary;

#sql { UPDATE emp SET sal = sal * :raise WHERE ename = :aName };
#sql { SELECT sal INTO :salary FROM emp WHERE ename = :aName };
```

But SQLJ is more flexible than that – you can use Java expressions instead of host variables. Just make sure that the host expression is enclosed between ":(" and ")" (see Listing 2).

***Tip:** All host expressions are evaluated once, and only once, from left to right (including side effects) before any values are sent to the database.*

- A host expression in an INTO list must be able to receive a data value. Write a host expression that is not legal in an INTO list. What happens when you compile and run your program?
- Show that you can use SQL comments in SQL text (between { and }), and that you can use Java comments inside Java host expressions in SQLJ statements.
- The examples above use a "SELECT INTO" statement. What happens if no row meets the selection criteria? What if more than one row is selected? How would you have to write JDBC code to implement similar functionality?

Look, Ma! Result Sets Are Typed . . . and Are Called "Iterators"

When you execute a query in JDBC, it will return a java.sql.ResultSet. You then retrieve the rows in the result set through a processing loop. The next() method on the ResultSet returns true if another row is available. In this case, the row is retrieved and the individual columns can be accessed through getXxxx(column_number) calls, where Xxxx represents the Java type with which you want to retrieve the column, such as String, Int (for int), Double (for double) and so on.

SQLJ does not have the "amorphous" result sets of JDBC. SQLJ query results are always strongly typed – each column in the result has a particular Java type. To differentiate these "typed result sets" from the JDBC notion of ResultSet and from the SQL notion of cursor, we call them *iterators*. First, you declare an iterator type by specifying both the column types in Java and the column names. The names also serve as names of accessor functions, with which the column value is retrieved.

Softwired

www.softwired-in.com/ibus

How do you get your iterator with all of these names and types? You declare it, of course!

```
#sql iterator Iter (String ename, Double sal);
```

This line creates a Java class declaration for the Iter class – right where you wrote it. This class has next() and close() methods – just like the java.sql.ResultSet. Instead of the getXxxx(column_name) accessors, however, your Iter class sports two fully customized, tailor-made, individualized accessor methods known as String ename() and Double sal(). A minor detail: you'll have the most success with this declaration if you put it where Java class declarations are permitted.

Let's declare ourselves an Iter.

```
Iter n;
```

And – better yet – populate it with the result from a query:

```
#sql n = { SELECT ename, sal FROM emp };
```

How do you use this iterator? Whaddayaknow, I *told* you all about these methods you find in Iter.

```
while (n.next()) {
    System.out.println(n.ename()+" would like to make "+
(n.sal()*2));
}
n.close();
```

Questions, questions! You should now have a gazillion questions about iterators, such as:

- Where do you declare an iterator type? (*Answer:* Wherever you declare a Java class, and with the same Java class modifiers.)
- Does the order in the SELECT list matter? (*Answer:* No.)
- How do you match SQL and Java names? What about case-sensitive and case-insensitive names? (*Answer:* Names always match in a case-insensitive way.)
- Can you say "SELECT * FROM EMP"? (*Answer:* Yes.)
- What happens when you use computed columns? (*Answer:* You need to employ SQL aliasing to get a name match on the column.)

You're encouraged to tackle some of the exercises below to confirm the answers given here.

***Tip:** If you want to declare an iterator locally (as an inner class), I recommend that you declare it as follows.*

```
#sql public static iterator IteratorName( … );
```

You must always close() your iterators once you are done using them, or you will run out of cursors with which to connect to the database.

EXERCISES

- Does the order in the SELECT list matter? Write a SQLJ program using the example above, and run it. Now reverse the order of the columns. Which behavior do you expect? Run the modified program and test your hypothesis.
- Can you use the query SELECT * FROM EMP? Change the example to use this form of SELECT. What behavior do you expect? Run the program and verify your guess.
- Is it a good idea to use a wildcard in SELECT statements in a SQLJ program? If yes, why? If no, why not?
  1. How do you match SQL and Java names? Show that the case doesn't matter by changing the case of the column names in the iterator declaration.
  2. How can you populate an Iter variable from a query with computed columns, such as SELECT 'BILL', 5000.0 FROM emp? Show that the case doesn't matter by changing the case of the column names in the SELECT statement. Also show that this is the case with case-sensitive column names.
  3. Which restrictions do you expect on column names in iterator declarations themselves? Show that SQLJ issues an error when these restrictions are violated.
  4. Which restrictions do you expect on column names in SELECT statements? When can SQLJ check these restrictions? Show that SQLJ can issue an error when these restrictions are violated.

Enough for today. Stay tuned! There's another important flavor of iterators that we need to talk about another time. So, next time, I'll consider how to call stored procedures and functions in the database and survey all of the Java types that are supported in SQLJ statements. Most of the SQLJ translator's mysteries still have to be unraveled as well. I'll also give a little history lesson about the numbering of SQLJ parts 0 through 2. In the meantime, keep your feedback, your answers to exercises and your questions rolling! I also encourage you to check out some of the SQLJ resources listed in the "SQLJ Resources" section.

SQLJ Resources

1. The SQLJ distributions (from www.sqlj.org and from http://technet.oracle.com) contain runtime API documentation, papers, and – most important – demo programs.
2. The Oracle Technology Network site (http://technet.oracle.com) provides SQLJ (as well as JDBC) downloads, tutorials, examples, discussion groups, and the full Oracle *SQLJ Developer's Guide and Reference.*
3. Oracle's JDeveloper Java development environment has supported SQLJ since version 1.1. Another vendor is said to be providing integrated SQLJ support in their Java development environment soon. Keep a lookout for that.
4. Morisseau-Leroy, N. et al. (1999). *Oracle 8i SQLJ Programming.* Osborne/McGraw-Hill. Other SQLJ books are in the works.
5. The ANSI SQLJ standard ANSI X3.135.10-1998, SQL Part 10: Object Language Binding (SQL/OLB) can be ordered from www.ansi.org.

AUTHOR BIO

Ekkehard Rohwedder has been hacking on the SQLJ translator since before it was called SQLJ. Prior to that he earned an MS from Carnegie-Mellon University. When he gets a break from leading the SQLJ development at Oracle, he dabbles in SQLJ primers.

erohwedd@us.oracle.com

Listing 1

```
import java.sql.SQLException;
import sqlj.runtime.ref.DefaultContext;
public class HiScotty {
  public static void main(String[] args) throws SQLException
  { new oracle.jdbc.driver.OracleDriver();
    DefaultContext.setDefaultContext
      (new DefaultContext("jdbc:oracle:oci8:@", "scott",
"tiger", false));
    String name = "SCOTT";

    #sql { UPDATE emp SET sal = sal * 1.1 WHERE ename =
:name };
    #sql { COMMIT };

    DefaultContext.getDefaultContext().close();
} }
```

Listing 2

```
String[] emps = new String[] { "Scott", "Miller", "King" };
double[] raises = new double[] { 8.0, 4.0, 0.0 };

for (int i=0; i<emps.length; i++)
   #sql { UPDATE emp SET sal = sal * :(1.0 + raises[i] /
100.0)
          WHERE ename = :(emps[i].toUpperCase()) };

int j=0; double[] s = new double[emps.length];
while (j<emps.length) {
   #sql { SELECT sal INTO :(s[j]) FROM emp
          WHERE ename = :(emps[j++].toUpperCase()) };   }
```

Flashline

www.flashline.com

Interview...with Doug Carrier

PRODUCT MANAGER, JAVA TECHNOLOGY
COMPUWARE CORPORATION'S NUMEGA PRODUCT LINE

JDJ: *Would you tell us a little about the recently released NuMega DevPartner 2.0 Java Edition?*
O'Brien: DevPartner Java Edition is a suite of productivity tools created to help developers build reliable, high-performance applications with Java technology. The suite contains a performance analyzer, memory profiler, thread analyzer and code coverage analyzer. Using the new DevPartner Remote Agent software, this release provides extensive server-side support for Java application servers, Enterprise JavaBeans and servlets on Solaris, Linux and Windows NT/2000.

JDJ: *How does DPJ help Java developers?*
O'Brien: Today's Web-enabled applications use many different technologies from HTML and JavaScript to JSP, servlets and EJBs. When these technologies are integrated into a Web-based app, developers usually end up spending a substantial amount of time chasing bugs and performance problems involving a number of components running on several different systems. DevPartner Java Edition helps developers quickly solve problems with runtime performance, memory utilization and multithreading impact application reliability, performance and overall software quality.

JDJ: *The memory profiler is new to DPJ. How do you see this tool assisting Java developers?*
O'Brien: Memory allocations are very expensive in Java, so when your code makes small allocations repeatedly, it can cause a noticeable degradation in performance. Inefficient use of memory can further compound this effect by depleting resources, causing serious reliability problems. Memory utilization is particularly critical with server-side Java, where small amounts of poorly utilized memory can quickly become fatal when the problem is multiplied by thousands or tens of thousands of simultaneous users.

The DevPartner memory profiler automatically locates memory-intensive and allocation-intensive Java methods and lines of code. With an accurate profile of a Java program's memory use, developers can improve the runtime performance and reliability of their application by optimizing methods that perform small, frequent allocations, and methods that waste or consume the most memory.

JDJ: *What can DPJ do for server-side Java, specifically for servlets, EJBs and JavaServer Pages?*
O'Brien: DPJ supports distributed application analysis using the new DevPartner Remote Agent software on remote Solaris, Linux and Windows systems. DevPartner Remote Agents can gather runtime session data on Java servlets, JavaServer Pages, JavaBeans, Enterprise JavaBeans, JavaScript, applets and other components. The runtime data collected from these distributed components are combined into a single end-to-end session file that can be analyzed on the developer's console.

DevPartner Java Edition is compatible with a range of popular server-side Java hosting environments including BEA WebLogic, IBM WebSphere, Allaire JRun, Apache JServ, Sun ServletRunner, Microsoft IIS and others.

JDJ: *What's unique to DPJ that isn't found in other Java productivity tools and suites?*
O'Brien: DevPartner Java Edition offers more breadth, depth, performance and compatibility than any other Java productivity tool suite. For starters, it contains four discrete tools, each tuned for a specific function – performance profiling, memory profiling, thread analysis and code coverage analysis. Next, it supports a wide range of Java application servers, Web servers and servlet engines across Solaris, Linux and Windows platforms. DevPartner Java Edition is also the first suite of its kind to support Web scripting languages like HTML, JavaScript, JSP and ASP.

DevPartner Java Edition works with standard VMs and maintains higher levels of performance than other Java tools. Most of the other Java tools on the market have a large memory footprint and can easily consume more than 20MB of Java resources just to load a single tool. DevPartner Java Edition tools run outside the VM using a minimal, on-the-fly instrumentation technique that's totally transparent to the user. It's fast, easy to use and provides more visibility into JIT-compiled code and native methods, where other tools usually fail.

JDJ: *Will the DPJ tools work with any Java VM?*
O'Brien: DevPartner Java Edition supports all the popular VMs with no special modifications, including Sun Java 1.1.7, 1.1.8, Java2, derived VMs from Symantec, Borland, Oracle and IBM, as well as the Microsoft VM.

JDJ: *Which Java development environments does DPJ work with?*
O'Brien: DPJ integrates with the most popular Java development environments – WebGain's [formerly Symantec] VisualCafé, Borland JBuilder, Oracle JDeveloper and IBM VisualAge for Java. It's also compatible with other development environments based on the Sun 1.1 or Java 2 SDKs.

JDJ Reader Feedback...

A robust rejoinder in response to a January 2000 news item

Blinded by the Sun?

The three contributors to your ***JDJ*** News item, "Sun's Withdrawal from ECMA May Not Signal End of World" [***JDJ***, Vol. 5, issue 1], seem to have been blinded by the light. It's clear to many of us that Sun's intention is to build for its own exploitation a platform that can compete with the Win32 Platform.

To many techies, a portion of Sun's appeal is its boisterous anti-Microsoft rhetoric, led by Scott McNealy himself who once, in a huge misjudgment, dismissed Bill Gates with the characterization: "Big hat, no cattle."

Is a platform that can compete with Win32 a bad thing? No. Do I embrace it? Emphatically. In the last 90 days I've developed tens of thousands of lines of tested code for an ASP business in which Java and Java servlets are pivotal.

Yet when Microsoft started pushing Win32 (in the form of Windows 95), it was viewed largely as a dominating and controlling move by the MS "Empire." Now, explain AWT Swing? Certainly you can't differentiate the two moves based on the technological merit of Swing over AWT, versus Win32 over Win16! Time marches on. The Java Platform must progress. So must the Microsoft Platform.

And on Sun's moves, such as their acquisition of Forté, surely you can't imagine these as philanthropic gestures. The fact that they chose to shed portions of the business to the Open Source movement must be recognized as a move carefully calculated for its impact on short- and long-term profits. If not, the public – holding nearly $150 billion of Sun stock – would be entirely within its rights to file suit against Sun's management for breach of fiduciary duty.

Yes, there are differences in the details. But Sun and Microsoft are in business *for their shareholders only*. If you'd held either company's shares in your portfolio, you'd have been handsomely rewarded by now. But as far as the "environmental consciousness" of either investment is concerned, don't be mistaken: these two 800-pound gorillas will take all comers to the mat.

The ECMA is not the central issue here. As I said, your three news-item contributors – and perhaps many of your readers – seem incapable of realizing, or perhaps admitting, that there are a great number of businesses profitably aligned on either side of this technology equation. It's just that one camp appears to take more comfort in its self-righteousness.

—**John Thompson,** Boulder, CO
JTBldrCO@aol.com

P.S. My personal opinions expressed above notwithstanding, keep up the great work on ***JDJ***...you're the best of breed!

Slangsoft

www.slangsoft.com/code/spirus.html

InstallShield Java Edition 3.0

by InstallShield Software Corporation

Cross-platform installation-authoring solution for multiplatform software developers

REVIEWED BY JOE MITCHKO

AUTHOR BIO

Joe Mitchko is a senior consultant with Computer Sciences Corporation where he specializes in Internet architecture.

jmitchko@csc.com

JAVA DEVELOPER'S JOURNAL JDJ WORLD CLASS AWARD

InstallShield Java Edition 3.0
InstallShield Software Corporation
Web: www.installshield.com
Phone: 800 374-4353
Fax: 847 240-9120
Address: 900 National Parkway
Suite 125, Schaumburg, IL 60173
E-mail: info@installshield.com

Pricing: $795.00 (upgrade $295.00)

Test Environment:
The evaluation was performed on a Dell Optiplex 8100 Pentium III with 256MB memory running Windows NT 4.0 SP6.

Having been involved in a fair number of development projects over the years, I've often wondered what goes on during the process of setting up and configuring installation programs using authoring tools such as InstallShield. The task of generating a setup program was always assigned to some unsuspecting junior programmer on the team. How they were able to get all the various components, registry settings and so on set up in the tool and compressed into a single file was always a well-guarded secret (I always thought it had something to do with job security). Anyway, driven by curiosity and answering the call of duty, I accepted the offer to evaluate the new 3.0 version of InstallShield Java Edition.

InstallShield is the de facto standard for installing MS Windows-based software today. If you're a Windows user, you've probably used it on more than one occasion. With the increasing amount of software being developed in Java today, InstallShield Software Corporation has developed a product tailored toward deploying cross-platform applications.

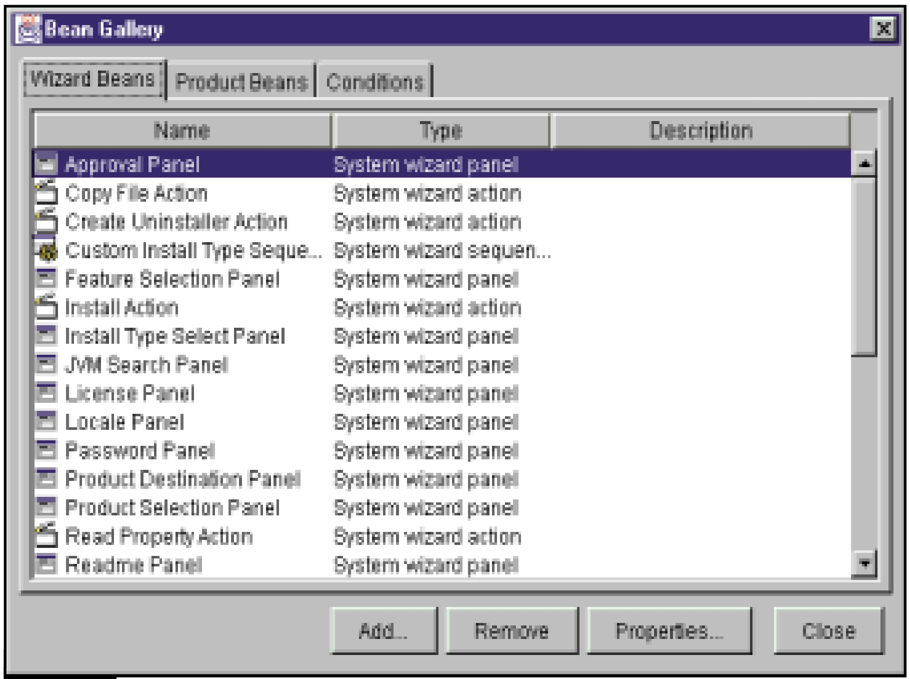
The new 3.0 release introduces a more powerful user interface, making it even easier to deploy software to multiple platforms in a seamless manner. The 3.0 release also provides more control over your installations by allowing you to develop custom UI actions and conditional rules.

Installation

As I popped in the install CD, I found myself wondering how many chances you get in a lifetime to install a program that's used to create other install programs. It's like compiling compiler source code – it just doesn't happen too often. The question did enter my head about whether the company that has essentially standardized the software installation process might turn out, when it came to installing its own products, to suffer from the same problem as the cobbler – you know, that of not having decent shoes for his (or her) own children. On the contrary, though, except for some minor issues that I'll explain shortly, the install went flawlessly.

Some things that you might reasonably expect when installing software from a CD were missing. Most CDs today incorporate an auto-install feature that starts things up when you place the CD in the disk drive. After placing the InstallShield CD in the test machine, the drive clicked, hummed and made its normal assorted sounds giving the impression that the install would soon begin. After a minute or so of silence, I had to make the inevitable decision that most of us have come to at one time or another when installing something: Is the installer program still thinking or is it in some zombie-like state? After giving it another minute or two, I took the next logical step and checked for an autorun.exe file on the root directory of the CD. I found none.

It turns out that in order to install InstallShield you need to click on the setup.html file located in the root directory of the CD. (I know, I know: read the installation manual first!) Another thing that caught my attention was that the volume label for the CD was set to some identification code (presumably the release or version number) as well as using the default icon for a CD. Again, a cosmetic issue, but something that can be improved on.



Name	Type	Description
Approval Panel	System wizard panel	
Copy File Action	System wizard action	
Create Uninstaller Action	System wizard action	
Custom Install Type Seque...	System wizard sequen...	
Feature Selection Panel	System wizard panel	
Install Action	System wizard action	
Install Type Select Panel	System wizard panel	
JVM Search Panel	System wizard panel	
License Panel	System wizard panel	
Locale Panel	System wizard panel	
Password Panel	System wizard panel	
Product Destination Panel	System wizard panel	
Product Selection Panel	System wizard panel	
Read Property Action	System wizard action	
Readme Panel	System wizard panel	

FIGURE 1 The Bean Gallery makes it easy to author your installation program.

Sybase

www.sybase.com

To start the installation, I double-clicked on the CD's setup.html file in Windows Explorer. This opened a browser page containing full setup instructions. I was on my way. A Java applet on the browser performed the actual installation – one of the features provided in InstallShield 3.0.

The installation requires you to choose a default Java Virtual Machine for deploying Java-based software on target machines lacking a preexisting JVM install. You're provided with several options, including the ability to use one of the JVMs currently installed on your machine (it even has the ability to search for existing JVM installations on your PC), or you can use the JVM that comes preinstalled with the InstallShield software. The remainder of the installation continued without any difficulties.

New Features

InstallShield's core architecture relies on an extendable set of JavaBeans, each specializing in one aspect of the installation process. Authoring a setup program involves stringing together a series of both visual and nonvisual JavaBean components, the end result being the finished installation (see Figure 1).

Version 3.0 provides a totally rebuilt IDE that breaks up the authoring process into a set of manageable views; I found it easy to use and intuitive (see Figure 2). Version 3.0 also provides the ability to export your installation setup to multiple environments including Java applications, Java applets or self-extracting setup files. For UNIX installations you can export your setup to run as a UNIX shell script (limited to Sun Solaris for now).

As mentioned earlier, this version includes an IDE that provides you with an easy-to-use interface for authoring your installation. The IDE uses a user interface similar to that found in Microsoft Outlook and includes a project tree, several project views, menu choices, toolbars and integrated help.

The view bar contains several icons that represent the various views (or steps) in the setup and configuration process. Authoring your installation involves configuring each view according to your requirements followed by a build.

The first icon located on the top of the view bar is the Project setup. This is where you identify your work environment from a project level, similar to setting up a project in other IDE environments. Here you create the names of your project and of your work file directories.

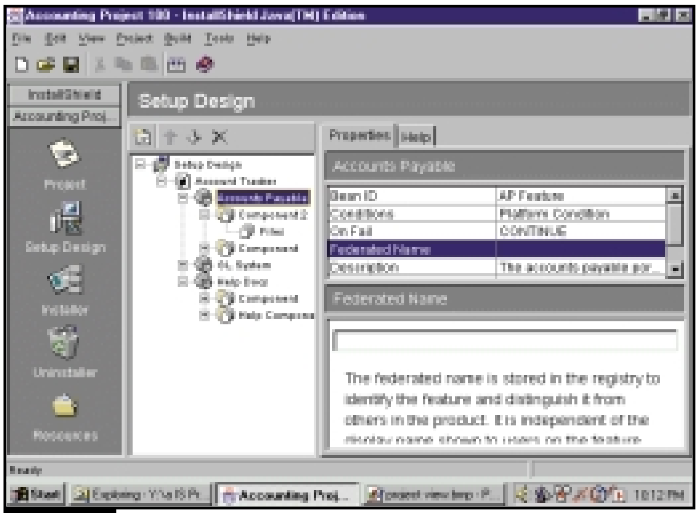
The next icon, the Setup Design view, allows you to define the product you need to install and its features. Each feature can likewise be assigned one or more components. The products, features and components are represented in a hierarchical tree structure in the IDE. The components can be files, registry settings, user environment settings or program groups. It's in this view that you specify how a component is to be installed on the target machine. The decision about when and if a feature and its associated components are to be installed is specified later in the authoring process.

The Installer view also allows you to set up a design tree that includes the installation properties and the sequence of events that take place during the install process. Depending on conditional rules, the setup panels in the design tree will call on various features and their components.

One nice feature that version 3.0 provides is support for uninstalling a product. The Uninstaller view provides capabilities similar to those in the Installer view, except that the actions performed are designed toward uninstalling various components instead of installing them.

The next icon is the Resource view. This view allows you to attach additional resource class files to your setup including support for internationalization and customized JavaBeans.

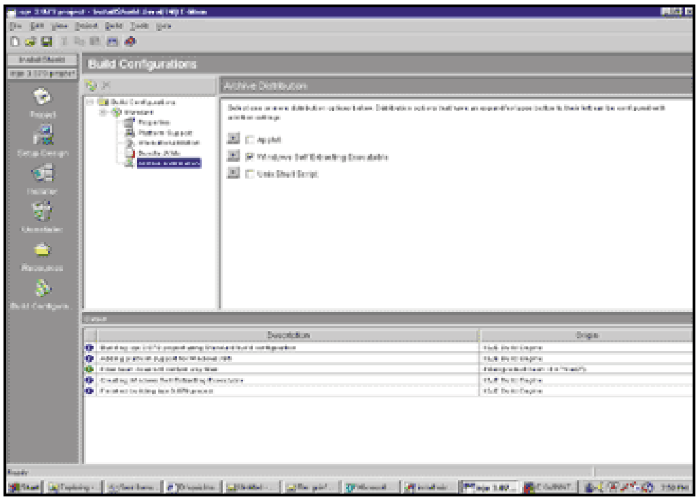
The last view, the Build Configuration view, allows you to configure property values describing the setup program's appearance to the user. In addition to choosing the export file type, you can specify which hardware and/or software platforms are to be supported by the setup as well as providing optional languages.



Bean ID	AP Feature
Conditions	Platform Condition
On Fail	CONTINUE
Federated Name	
Description	The accounts payable por...

The federated name is stored in the registry to identify the feature and distinguish it from others in the product. It is independent of the display name shown to users on the feature.

FIGURE 2 The InstallShield Java Edition 3.0 provides a more powerful IDE.



Build Configuration – create the archive

Taking a Test Run

Authoring your installation requires the following steps:

1. *Create a new project.*
2. *Set your project properties.*
3. *Design your application setup.*
4. *Create the install/uninstall sequence.*
5. *Configure the build.*
6. *Build the archive.*

Although I'd never worked with InstallShield before, I was able to put together quite a polished self-extracting install package with little effort that looked and behaved like the real thing. I started off by simply configuring InstallShield to install the nearest file I could get my hands on – an unsuspecting readme.txt file located in the file browser. The setup program did a fine job of installing this file to the target directory on my PC. Not much of an install, but it did force me to go through each step in the authoring process.

The setup panels you'd normally see in an InstallShield setup program were easily identifiable as JavaBean components in the Installer view. Anybody who's used InstallShield would easily recognize the purpose for each JavaBean component.

Authoring my simple setup program was relatively easy and was done in one authoring session. To help you along the way, InstallShield provides you with step-by-step instructions and wizards. As I tried out the various install panels, I found myself repeatedly building a setup program, then executing the resulting setup to see how it looked. One thing not provid-

Hotdispatch

www.hotdispatch.com

ed in this release, but which would be nice, is a preview button that would allow you to go through the install process without actually having to build and run the setup program. Maybe in the next release....

After several dozen build and test cycles, I started running into some memory problems on my PC – the JVM's garbage collection mechanism probably hadn't been able to catch up. After rebooting my machine, things were back to normal. I can't foresee experienced users, who can create a finished product with fewer development cycles, having this problem.

The ability to customize the installation process by assembling JavaBean components together is a nice feature. Each install panel and its functionality are represented in a separate JavaBean. For example, there are panels designed to allow the user to enter the install directory or display the legalese required for licensing (it also includes agree and disagree buttons). Each of the window frames typically found in an install program can be selected from a panel and assembled into the installation sequence. Although each JavaBean component available in the "Add Bean Wizard" did include help information describing its purpose, it would be nice to see an example of how the frame would look in the install program ahead of time. Also, the same icon was used to represent each JavaBean component in the IDE and didn't help you visualize what their representative panel looks like.

One additional note is that when I attempted to author a Web site installation involving approximately 10M of image and HTML files on a notebook computer with 128MB of RAM, the authoring process was problematic. When I performed the same operation on a more powerful desktop computer having 256MB of memory, it completed the authoring without any problems.

Solaris Installation

Another feature introduced in InstallShield 3.0 is the ability to install software on the Sun Solaris operating system using the same interface used for Microsoft Windows. I recently was involved in installing Solaris on a Sun Enterprise 450 Server. Being accustomed to UNIX installations in the past on other platforms, I wasn't looking forward to going through the ordeal. Instead of the usual set of command line instructions, buggy shell scripts and out-of-date documentation, I was pleasantly surprised to find InstallShield used in the Solaris Web Start program setup. It made installing Solaris a snap.

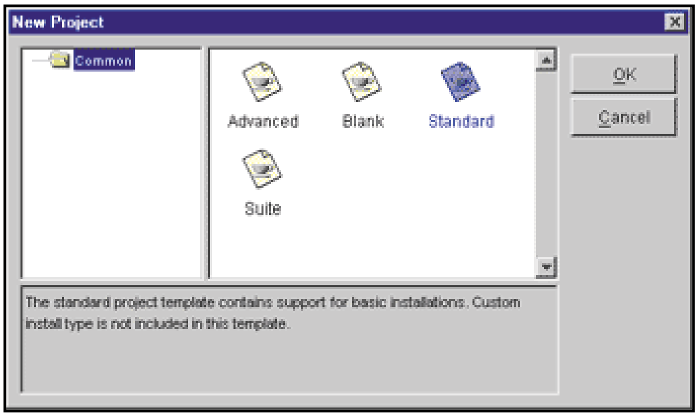
This is actually one of the better examples of how Java's write once run everywhere architecture can be used to create standardized user interfaces across different hardware and software platforms. For UNIX administrators who have grown accustomed (and inured) over the years to the crude shell script-based install "programs" found in UNIX, InstallShield Java Edition comes as a welcome relief.

Additional Features

Some additional features include support for migrating version 2.5 projects to version 3.0 and additional language support for Chinese, Korean and Swedish. The IDE environment provides useful help information to guide you through the authoring process. You can also design custom JavaBean components and employ them in your setup, providing unlimited customization.

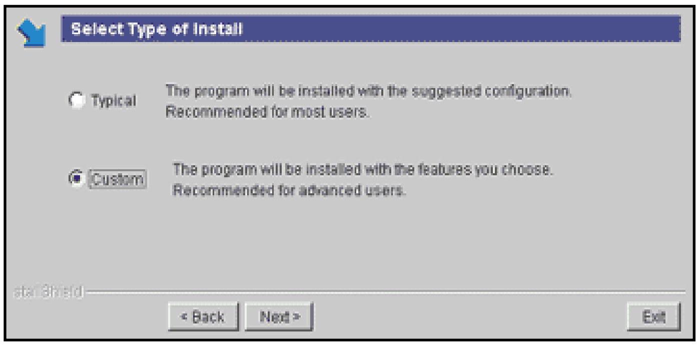
Summary

InstallShield 3.0 Java Edition introduces many new features and makes it even easier to author your installation programs. It provides out-of-the-box functionality that will cover the majority of your installation needs, as well as cross-platform support for deploying software to Windows- and UNIX-based systems. The InstallShield software is easy to install and to set up – as befits a company that specializes in installing software!



The standard project template contains support for basic installations. Custom install type is not included in this template.

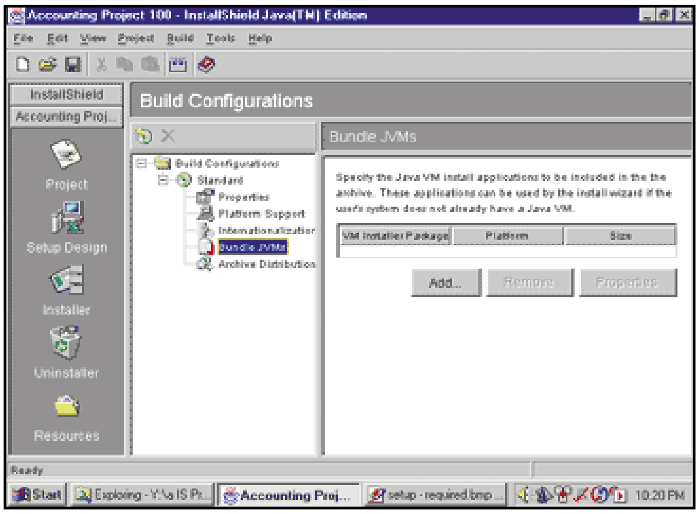
Ready-to-use project templates



Typical — The program will be installed with the suggested configuration. Recommended for most users.

[✓ Custom] — The program will be installed with the features you choose. Recommended for advanced users.

Support for custom and typical install types

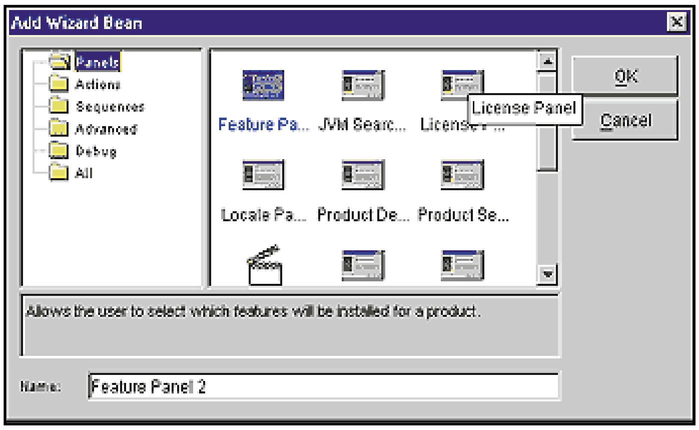


Build Configurations

Bundle JVMs

Specify the Java VM install applications to be included in the the archive. These applications can be used by the install wizard if the user's system does not already have a Java VM.

One-step Java Virtual Machine bundling



Allows the user to select which features will be installed for a product.

Name: Feature Panel 2

Installation UI management

The Object People

www.objectpeople.com

Team Programming with VisualAge for Java

How to create a new business application for a typical corporation

WRITTEN BY
Susan Yeshin
& Ellen McKay

As the software development process becomes ever more complex, end users demand more and more functionality in less and less time. Companies are extending their business applications to run on both intranets and the Internet, and new applications have to run on many different platforms. All this calls for large development teams to design, build and maintain applications.

Often, an entire team of managers, developers and testers have to work together to deliver an end product. Team members share more than responsibility for developing the code; they also share the actual files that contain the code. Thus it's essential to control each team member's access to the code and how code changes affect others. To manage the activities of the team, many application development products provide source code management tools and team programming environments in which the roles of individual developers are clearly defined and enforced.

IBM VisualAge for Java, Enterprise Edition, is a good example of an application development product that provides a collaborative team development environment. This article examines the concepts of the source repository, program element ownership and team roles in VisualAge for Java. It also walks you through a scenario that helps you understand how team development works.

Repository

VisualAge for Java, Enterprise Edition, has a collaborative team development environment based on a shared repository. The repository is a source control mechanism that allows you to track changes made to program elements. When you start the IDE, it connects to a shared repository on a server. As team members create and modify program elements in their individual workspaces, their changes are automatically saved in the shared repository. (The workspace is a storage area on your client machine; it contains the Java code that you are currently developing and the class libraries on which your code depends.) When you remove program elements from your workspace, they remain in the repository so you can undo changes by retrieving previous editions. The repository contains all editions of all program elements.

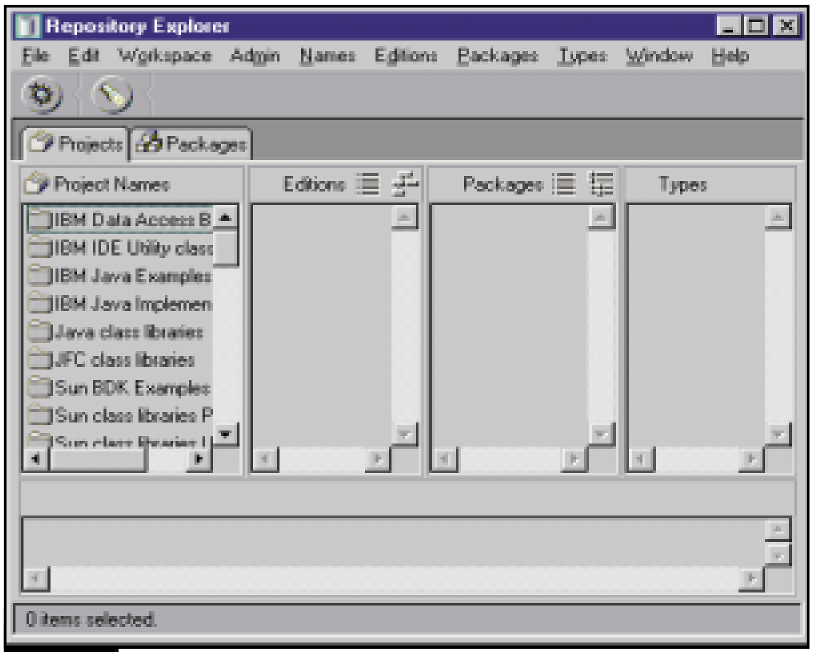


FIGURE 1 The Repository Explorer window

The Repository Explorer window provides a visual interface to your repository (see Figure 1).

Within the Repository Explorer you can open or compare program elements that are stored in the repository. Programmers select the editions they want to add to their workspace, and then work with those program elements in another window called the Workbench.

Ownership

In the team development environment, source control management is based on program element ownership. VisualAge for Java shows code to you as a hierarchy of program elements (see Table 1).

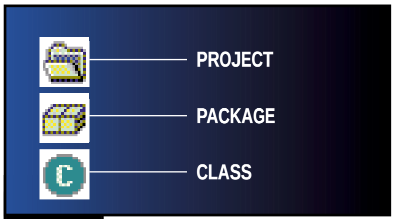


TABLE 1 Program elements in VisualAge

Intuitive Systems, Inc.

www.optimizeit.com

A project is an organizational construct that groups packages together. In VisualAge for Java every program element – projects, packages and classes – has an "owner" who is responsible for its stability and coherence.

Team members have certain privileges based on their team role. These privileges help the team manage their code development by ensuring that developers don't overwrite each other's code and that they're working from the same code baseline.

A class can be "owned" by only one developer, but several developers can work concurrently on the same class, in separate open editions. Each developer can version only their own edition of the class, and create a new open edition of it.

The owner of a class is the only developer with the authority to compare various editions of the class and merge them to create a single new version. And only the class owner can release the new version of the class into its containing package. Releasing indicates to other team members that this is a good baseline to work from; they can pick up the latest changes by reloading the whole package (or project) into their workspaces. A project or package can be versioned or reopened only by its owner.

Open Editions

Open editions are works in progress. Before you can make changes to an existing project, package or class, its owner must create an open edition of it. Only one open edition of a program element is permitted in your workspace at any one time. In the repository, however, you can have multiple open editions of the same program element, with each one implemented differently. For example, if you're adding features to an application that you've customized for different industries, you might have multiple open editions of a package with the same name stored in the repository.

Versioned Editions

Versioned editions can't be changed. You may version your open editions for any of the following reasons:

- *To keep a copy of a program element at some significant development point so you can return to it at a later date.* In the case of packages and projects, versioning freezes a specific configuration of the contained classes.
- *To make your changed classes available to other team members browsing the repository.* Only versioned editions are visible to other team members.
- *To release a class into its containing package in order to update the team baseline.* Only versioned editions can be released.

After you've versioned a program element, you can open a new edition if you want to make more changes. You can restore the open edition of a program element in your workspace to a versioned edition of it at any time.

"The owner of a class is the only developer with the authority to compare various editions of the class and merge them to create a single new version"

Team Roles

There are four main development roles and one administrative role in VisualAge for Java. In a development team that consists of a manager, a team leader, a senior developer and junior developers, the roles are often distributed as shown in Table 2. What follows is a brief explanation of each role.

REPOSITORY ADMINISTRATOR

The *repository administrator* is responsible for the creation, installation and maintenance of shared repositories on a server, and for controlling access to these repositories. The administrator creates the shared repositories, prepares a list of the repository users and enables password validation. User lists and password validation ensure that only the appropriate team members can access the code in the repository.

PROJECT OWNER

A *project owner* is responsible for the overall quality of a project. A project owner coordinates the activities of the package owners who release their packages into the project owner's project.

Only the owner of a project can perform the following tasks:

- Create an open edition of the project so that package owners can update the team baseline by releasing their packages.
- Add packages to the project and delete packages from the project.
- Version the project to preserve a team baseline.

If the project owner doesn't create an open edition of the project, package owners can't release their packages into the project. This gives the project owner complete control over when a project is frozen and when it's reopened for further development.

PACKAGE OWNER

As you might guess, a *package owner* is responsible for the overall quality of a package. The owner of a package coordinates the activities of the class owners who are releasing their versioned classes into that package.

The creator of a package is, by default, the owner of that package. Each edition of a package has a group of developers who are assigned to work with classes in that package. These developers are the *package group* for that edition. Only package group members can own, create and change classes in the package.

POSITION	VISUALAGE FOR JAVA TEAM ROLE
LAN administrator, or a team member	Repository Administrator
Development manager/project architect	Project Owner
Lead developer	Package Owner
Other developers on the team	Class Owner, Class Developer

TABLE 2 Team roles in VisualAge for Java

Visicomp, Inc.

www.visicomp.com

Only the owner of a package can perform the following tasks:

- Add members to the package group and delete members from the group.
- Create an open edition of the package so that class owners can update the team baseline by releasing or deleting classes.
- Version the package to preserve a team baseline.

Package owners control when a package is frozen and when it's opened for further development. Both the project owner and the package owner can release the package into, or delete it from, its containing project edition.

CLASS OWNER

A *class owner* is responsible for the integrity of a particular class and its methods. Only the owner of a class can update the team baseline, in the following ways:

- Release editions of the class into the containing package.
- Delete editions of the class from the containing package.

Often the owner and developer of the class are the same person. If two or more developers are working on the same class, the class owner is responsible for comparing and merging everyone's changes into a single edition of the class. The class owner then versions the merged edition and releases it.

CLASS DEVELOPER

Any team member can create new classes and open editions of existing classes, thereby becoming the developer of that open edition of the class.

Only the developer of a particular class edition can version that edition, and only the class owner can release the version. The *class developer* and the class owner are usually the same person, but there may be more than one developer working on the same class.

A Team Programming Scenario

The following simple scenario shows how a small development team would use VisualAge for Java to create a new business application for Xyz Corporation. In the scenario the team works together to develop some sample program elements for the application. There are five developers in the following roles (see Figure 2 for who reports to whom).

In a team development environment, there are two main steps in project development:

1. Setting up the project
2. Concurrent development

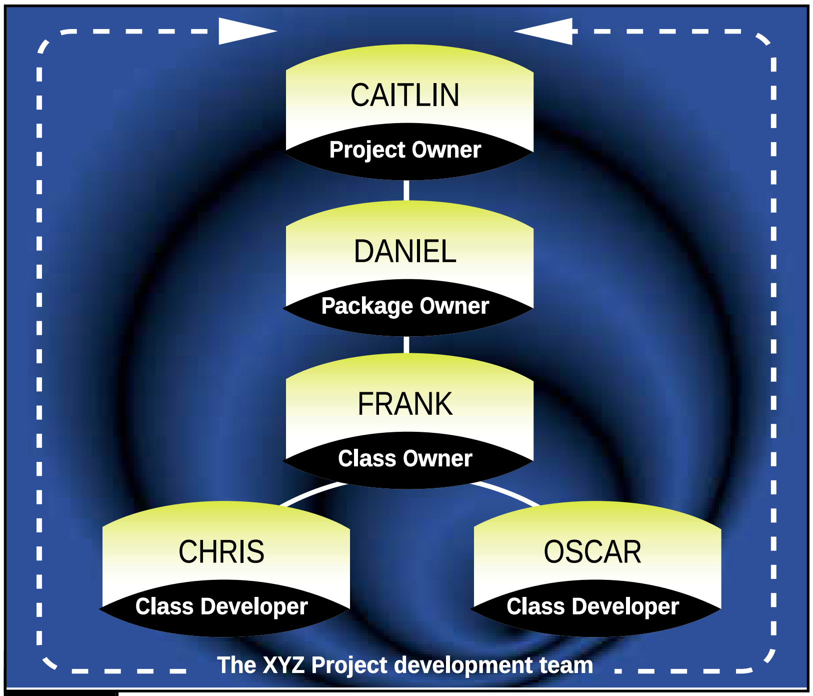


FIGURE 2 The team roles

When the project is set up, ownership of the project elements is assigned to the team members.

STEP 1: SETTING UP THE PROJECT

1. Caitlin creates a new project, Xyzcorp, and a new package within that project, com.ibm.xyz. Open editions of these program elements reside in both Caitlin's workspace and the shared repository.
2. Caitlin adds Daniel, Frank, Oscar and Chris as package group members. She makes Daniel the owner of the com.ibm.xyz package. They can all now work with classes in the com.ibm.xyz package.
3. The team members add the package to their workspaces. They now all work with the same open project and package editions.

Now that the project is set up and ownership is assigned, the developers can begin working concurrently on the code.

STEP 2: CONCURRENT DEVELOPMENT

1. Frank creates a new class, Xyzgrid, and Oscar and Chris load it into their workspaces.
2. Oscar adds a new method to the class, XyzGetCol. He versions the class, so it's available to the other team members. He then creates an open edition of the class and starts working on another method, XyzGetRow. His new changes won't be available to the rest of the team until he versions his class again.
3. Chris adds a new method to the class, XyzLayout. He versions the class, so it's available to the other team members.
4. Frank takes the versioned classes that Chris and Oscar own and merges their changes. He then versions the new edition of Xyzgrid and releases it.
5. Daniel releases the com.ibm.xyz package and Caitlin versions the Xyzcorp project.

After Caitlin has versioned the project, the development cycle can begin again. New editions of the Xyzcorp project and the com.ibm.xyz package can be opened, and the team members can reload these new editions into their workspaces and start enhancing the code again.

Conclusion

As an application grows and becomes more complex, its development team also grows and becomes more complex. Managing and integrating source code are key to successful project development. The team development environment in VisualAge for Java provides change control and stability without sacrificing flexibility or programmer productivity. Emphasis is on roles and responsibilities rather than on file locking. This approach is especially well suited to the iterative, collaborative, object-oriented development that tends to characterize most successful large e-business applications.

syeshin@ca.ibm.com ecmckay@ca.ibm.com

AUTHOR BIOS
Susan Yeshin, an information developer at IBM for the past three years, currently writes online help for various VisualAge for Java components, including the IDE and Team Programming.

A recent graduate of the University of Waterloo, Ellen McKay has been an information developer at IBM on the VisualAge for Java team for the past year. She currently writes the Getting Started guide and Team Programming help.

Prosyst

www.prosyst.com

Developing Coarse-Grained Business Components

Everything you ever wanted to know about coarse-grained entity beans and dependent objects . . . but were afraid to ask

WRITTEN BY
JASON WESTRA

Discussion groups have recently been abuzz with talk of "coarse-grained entity beans" – a slight misnomer deriving, I suspect, from the addition of mandatory entity beans in EJB 1.1. This month I'll examine the finer points of the Enterprise JavaBeans specification regarding coarse-grained entities, as well as my own, and provide an example for you – with plenty of comments to provide food for thought when you tackle the challenge yourself.

For those of you who finally came out of your Y2K shelter, EJB is a specification for building server-side business components on the Java platform. Its initial release required only enterprise beans called *session beans*, which are a logical extension of a client's business process. The specification mentioned *entity beans*, which represent persistent business components, but didn't mark them as mandatory for EJB 1.0 compliance.

However, industry demand for details about entity beans forced Sun and its partners to include them in the EJB 1.1 release. The updated specification provides some structure around entity bean development, but it created a whole new set of questions about EJB design patterns. The main EJB design pattern I'll cover this month revolves around coarse-grained entity beans and dependent objects.

Granularity of a Bean

I was recently on "The Hill," near the campus of the University of Colorado at Boulder, and noticed that the sidewalk had numerous quotes from philosophers and great thinkers – they must be there to provide inspiration to students who look at their feet when they walk or something! I'm going to start off with a few quotes myself, by way of helping to describe the granularity of an entity bean.

Section 9.1.2 of the EJB specification 1.1 reads: "In general, an entity bean should represent an independent business object that has independent identity and lifecycle, and is referenced by multiple enterprise beans and/or clients. A dependent object should not be implemented as an entity bean. Instead, a dependent object is better implemented as a Java class...and included as part of the entity bean on which it depends."

Great, but now you're asking what in the heck "dependent objects" are. Richard Monson-Haefel sums up the notion of dependent objects well in this quote from his book *Enterprise JavaBeans* (O'Reilly and Associates, 1999): "Dependent objects are objects that only have meaning within the context of another business object. They typically represent fairly fine-grained business concepts, like an address, phone number or order item. For example, an address has little meaning when it is not associated with a business object like Person or Organization. It depends on the context of the business object to give it meaning. Such an object can be thought of as a wrapper for related data. The fields that make up an address (street, city, state, and zip) should be packaged together in a single object called Address. In turn, the Address object is usually an attribute or property of another business object; in EJB, we would typically see an Address or some other dependent object as a property of an entity bean."

If I could say it better, I would.

Why Coarse-Grained?

The three core reasons for building coarse-grained entity beans are:

1. *Reusable business value*
2. *Less work to develop and deploy*
3. *Scalability, manageability of EJB Server*

First, let me reiterate: Enterprise JavaBeans is a specification for server-side business components for the Java platform. EJB strives for reusable, configurable business components. With EJB you're no longer working with fine-grained objects. Small business objects by themselves provide little value. Most likely they'll rely on helper objects to provide real value, and together they make a component. Coarse-grained components provide a reusable piece of business functionality that can be applied to many different solutions as a single unit. Although I'm happy to see people get excited about innovative EJB design patterns, coarse-grained components are really nothing new; the design pattern has been around for years. I used to call it a "constellation," a collection of interrelated objects, all fetched from a root business object and dependent on that parent for management of their life cycle.

Second, enterprise beans aren't easy to develop, so the fewer beans you need to develop and deploy, the faster your applications are going to get to market. For instance, when developing a typical entity bean, you must code and test a remote interface, home interface, bean class, primary key class (unless you're using the J2EE reference implementation – has anybody figured that thing out yet?) and deployment descriptor.

You need to walk through various steps to package these classes in the correct format and deploy them in a nice little JAR file along with a manifest file describing each entry. Couple your EJB classes with the classes your container generates to manage your component, and you have a major class explosion on your hands. Sounds like a lot of work, and it is. However, building coarse-grained components eliminates tedious steps and class explosion by encapsulating the functionality of dependent classes within their parent entity bean.

Third, fewer entity beans means fewer resources for your server to manage and greater performance for object interactions within your components. Entity beans are remote objects, which can be costly for a server to manage when their numbers are high. For instance, a bean's home is registered in a naming service for remote lookups. As the naming service becomes full of entries, lookups may take longer. Have you ever done a file search on a directory with 10 files, then the same search on a directory with 100, 200 or 1000+ files? You get the picture.

Your EJB Server should provide a scalable, secure execution environment for your EJBs; however, you should work with it, not against it, to promote this scalability in your bean design. Because beans are remote objects, the server is

Modis Solutions

www.idea.com

required to manage remote connections and garbage collection, and usually maintains caches for your bean instances. Fine-grained objects usually interact closely with one another to perform their business functions. Fine-grained interactions aren't suitable for remote objects, which must marshal calls to each other, repeatedly creating copies of parameters and return values that must be garbage-collected at some point. These interactions are more conducive to a coarse-grained business component that encapsulates all logic behind its umbrella of locality.

While your application logic may be scalable, sometimes the pipeline to it or the server itself gets taxed. Keeping the number of remote connections to your server, remote method invocations and bean instances to a minimum is thus advisable. A coarse-grained business component encourages these best practices.

An Example: PersonBean

It's time for an example of a coarse-grained entity bean that manages dependent objects according to the EJB specification. I've chosen the Person business component because it appears in various shapes and sizes across numerous industries. My example (see Listing 1) consists of a Person entity bean and an Address, a dependent object and a test class to see the bean in action. It's written on WebLogic 4.5.1, an EJB 1.0-compliant server; although 5.0, which is 1.1 compliant, isn't available at this writing, I've nevertheless adhered to the EJB 1.1 specification's guidelines for building coarse-grained entity beans.

The example demonstrates a coarse-grained, bean-managed persistence (BMP) entity bean that manages the whole life-cycle of its dependent objects from insert to delete. It is configurable via its deployment descriptor to perform either lazy-loading (fetching only when requested) or eager-loading (loading dependents in anticipation of their use) of its dependent objects, both of which are supported by the spec. It then performs smart writes to the database, updating only modified dependent objects in its cached data.

As you'll see from the code at the end of the article, the example is loaded with comments to help you think about the issues behind building your own BMP entity beans.

Person to Address Association

The basic rules governing the PersonBean is that a Person can exist in the system without an address; however, an Address is dependent on the Person – it can't be inserted without its parent (i.e., PersonBean) and must be removed when the Person is deleted. This logic is evident in the factory interface of the PersonBean. For instance, the PersonHome.create method takes only Person attributes, but requires no Address object(s).

```
public interface PersonHome extends
EJBHome {
    PersonEJB create(int personID,
String title,
        String firstName, String last-
Name)
     throws CreateException, Remote-
Exception;
```

This inherently implies that it may be inserted without addresses, though the interface is flexible enough for an application requiring a Person to have at least one address. Such an application could wrap the PersonBean in a session bean and call PersonEJB.addAddress(newAddress) in the same transaction, ensuring that an Address was always inserted along with the Person. Already, you've seen a reusable scenario for this coarse-grained business component!

Caching Dependent Objects

When developing coarse-grained entity beans, you'll have to understand the EJB 1.1 specification's demands around loading and storing cached data.

LOADING CACHED DATA

The EJB specification lists three ways to load cached data in an entity bean. The first is an *eager-load*, in which your code loads all cached data at once in the ejbLoad method (see Listing 1). The second approach is to *lazy-load* entity data, loading it only when it's needed to satisfy a business method. The last method to caching entity bean data is not to cache at all, but to pass all access to entity bean state directly to the underlying database.

The PersonBean caches its own fields as well as dependent Address objects in a local java.util.List. The loading of the Address list is determined at deployment of the bean via its "lazyLoad" bean environment property. If it's true, the ejbLoad method will *not* load address data. Instead, it will clear the list to ensure proper management of the cached data according to the EJB spec, section 9.1.7 (see Listing 1). Cached data that's not loaded needs to be cleared here, since ejbLoad is called whenever a container needs to synchronize state with the database, such as in the event of a javax.transaction.TransactionRolledBackException.

If "lazyLoad" is false, it will perform a JDBC query to fill its list right away. The PersonBean uses environment properties for flexibility and reusability by allowing bean deployers to modify the behavior of the component declaratively without ever touching the code.

A copy of the PersonBean's Address list is publicly accessible via the component interface method getAddresses. When the container invokes this method on behalf of a client, the PersonBean will check to see if the address list is already available. If so, it returns the list without reading from the database. Otherwise the PersonBean will use a JDBC query to fetch all of its addresses, caching each in its local list before returning the list to the caller.

UPDATING AND REMOVING CACHED DATA

The EJB specification states, "When the container invokes the ejbStore method on the instance, the instance must push all cached updates to the entity object's state to the underlying database" (EJB specification 1.1, sec. 9.1.7).

The key phrase in this quote is *cached updates*. In other words, you can apply "smart updates" to your bean's cached data by checking for modifications and updating only this information back to the database.

The PersonBean uses smart updates for its dependent Address objects (see Listing 2). For instance, when the client updates an Address, the PersonBean will mark the Address as "modified" and replace the old version with the new one in the local list. The PersonBean will wait until ejbStore is invoked before updating the modified Address object to the database. When the container calls ejbStore, the PersonBean will loop through each Address in its list, calling a JDBC Update only for modified objects.

Similar to the way ejbStore manages all cached state for the entity bean, ejbRemove also handles the referential integrity of the PersonBean component, ensuring that all Address objects are removed when the PersonBean is deleted (see Listing 3). There are a couple of ways to maintain component integrity upon deletion of a parent. One approach is to create a cascading delete in the database that will automatically delete all related rows in the Address table when its Person is removed. In this case there would be no need to code a JDBC Delete call from within ejbRemove. A second option, the one I chose, is to manually perform the JDBC Delete of child-dependent object rows. This ensures the component's integrity no matter what JDBC-compliant database platform it's deployed against. Both methods have advantages and disadvantages, so "pick your poison."

The management of cached data in an entity bean is confusing at first, but soon you'll find it flexible enough for you to devel-

Iam Consulting

www.iamx.com

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

op tuned, configurable data access code. The PersonBean uses two tuned data access patterns: lazy-loading and smart updates.

Summary

I hope this discussion has dissipated the confusion that's been circulating within numerous discussion groups about how to develop these components. The example demonstrated how to build a BMP entity bean that controls the life cycle of a dependent object and focused on some gray areas of the EJB 1.1 specification, such as how to manage cached state when loading, storing and removing an entity bean that encapsulates dependent objects. Please refer to the full code listing on the Web (www.sys-con.com) and modify the test case to perform more in-depth exercises on the behavior of a coarse-grained component.

jwestra@verge-tg.com

Listing 1: PersonalBean.ejbLoad

```
/**
    * Loads the entity bean from the persistent storage.
    * Checks whether or not to load dependent address objects
    *
    * @exception                java.rmi.RemoteException
    *                           if there is a communications
                                or systems failure
    */
    public void ejbLoad() throws RemoteException {
 System.out.println("ejbLoad ("+personID+")");
 try {
     // SQL actually performed in the read() method
     read((PersonPK) ctx.getPrimaryKey());

     // discard or reload any cached data, as per EJB 1.1
     // spec, sec 9.1.7
     if (lazyLoad.equals("true")) {
  System.out.println("ejbLoad ("+personID+") - lazy load
                     addresses");
// ensures reset of address for this PersonBean!
  addresses.clear();
     }
     else{
                 System.out.println("ejbLoad ("+personID+") -
                                   loading addresses...");
  getAddresses();         // loads the address list now
     }

        }
 catch (FinderException fe) {
     throw new RemoteException (fe.getMessage());
 }
    }
```

Listing 2: PersonBean.ejbStore

```
/**
    * Stores the entity bean's PersonBean fields in the
    * persistent storage.
    * Also saves any modified Address objects.
    *
    * @exception                java.rmi.RemoteException
    *                           if there is a communications
                                or systems failure
    */
    public void ejbStore() throws RemoteException {
 System.out.println("ejbStore (" +personID+ ")");

 Connection con = null;
 PreparedStatement ps = null;
 try {
   con = getConnection();
   ps = con.prepareStatement("update Person set title = ?, "+
        "firstname = ?, "+
        "lastname = ? "+
        "where personid = ?");
   ps.setString(1, title);
   ps.setString(2, firstName);
   ps.setString(3, lastName);
   ps.setInt(4, personID);

   int i = ps.executeUpdate();
   if (i == 0) {
  throw new RemoteException
 ("ejbStore: PersonBean ("+personID+") failed to update");
   }

   // update cached entity state (i.e. our addresses)
   // as per EJB spec 1.1, sec: 9.1.7
   Iterator list = addresses.iterator();
   while (list.hasNext()) {
       Address addr = (Address)list.next();
// only update if it was modified
       if (addr.isModified()) {
  sqlUpdateAddress(addr);
       }
   }
 }
 catch (RemoteException re) {
     throw re;
 }
 catch (SQLException sqe) {
     throw new RemoteException (sqe.getMessage());
 }
 finally {
   try {
     if (ps != null) ps.close();
     if (con != null) con.close();
   }
   catch (Exception e) {
     throw new RemoteException (e.getMessage());
   }
 }
    }
```

Listing 3: PersonBean.ejbRemove

```
/**
    *  Deletes the entity bean and all addresses (dependent
    *  objects)from the persistent storage.
    *
    * @exception                java.rmi.RemoteException
    *                           if there is a communications
                                or systems failure
    * @exception                javax.ejb.RemoveException
    *                            if the entity does not
                                 allow removing the bean
    * @exception                java.rmi.RemoteException
    *                           if there is a communications
                                or systems failure
    */
    public void ejbRemove() throws RemoveException, RemoteEx-
     ception {
 System.out.println("ejbRemove ("+personID+")");

 addresses.clear(); // reset

 Connection con = null;
 PreparedStatement ps = null;
 try {
   con = getConnection();

   // get the PK from the context.  This is because ejbLoad
   // does not have to be called (see sequence diagrams in
   // EJB spec1.1) on an ejbRemove() and using the personID
   // at this point may give the personID of a different bean!
   PersonPK pk = (PersonPK) ctx.getPrimaryKey();

   ps= con.prepareStatement("delete from person where person
                             ID = ?");
   ps.setInt(1, pk.personID);
   int i = ps.executeUpdate();
   if (i == 0) {
     throw new RemoteException ("ejbRemove : PersonBean
("+pk.personID+ ") not found");
   }

   ps.close();

   // delete all addresses for the personBean
   ps = con.prepareStatement("delete from address where per-
                              sonID = ?");
   ps.setInt(1, pk.personID);

           ps.executeUpdate();
 }
        catch (SQLException sqe) {
   throw new RemoteException (sqe.getMessage());
 }
 finally {
   try {
     if (ps != null) ps.close();
     if (con != null) con.close();
   }
   catch (Exception e) {
     throw new RemoteException (e.getMessage());
   }
 }
      }onBean.ejbRemove
```

Fiorano Software

www.fiorano.com

Interview... with RANDY HIETTER

Vice President of Product Management, Persistence Software

JDJ: *Randy, tell us a little bit about Persistence.*
Hietter: Well, Persistence Software is an EJB application server vendor company. We actually got started back in 1991 building object-relational mapping tools. Our founders saw that relational databases weren't going to go away, that they were going to be the mainstream way of persisting data. However, they saw that the traditional coding methods were going to be eclipsed by object-oriented programming and design, so they married the two through object-relational mapping. We can take information out of a relational database and expose it as software objects, making it much easier for developers to interact with data, rather than the clumsy APIs and now portable SQL statements that a lot of developers have suffered through in the last few years. Our genesis, again, is object-relational mapping, which we have since taken up to a full EJB application server that supports it.

We do provide container-managed persistence, which means that developers don't have to code the low-level [JDBC] statements and SQL to access and update data from a relational database. We're one of the few vendors that do provide container-managed persistence and that's a key aspect of our product, Power Tier, in terms of giving developers very high productivity. Let me give you an example. We have the ability, through our object builder utility, for developers to pump in an object model from a Rational Rose system where they can manually enter their object model. Or if they already have a database schema that pretty much reflects their object model, they can suck that in from an Oracle or Sybase environment and through our utility create, literally with a one-button operation, all of the entity beans and interfaces necessary for that application. The productivity is tremendous in terms of what it gives developers – the ability to automatically create entity beans. All they have to worry about are the business rules representing 20 – 30% of application code typically implemented as session beans or increasingly in servlets.

JDJ: *Essentially, the Java developer doesn't need to go anywhere near SQL statements?*
Hietter: Correct. But they still can create their own SQL for other complex operations if they choose.

JDJ: *How does the developer build up queries then?*
Hietter: You can do that when you define your object model. You do have an ability to go around and create your own SQL statements and dynamic queries if you choose to do so. But in terms of queries, again, what Power Tier does is create the relationships between the objects and store these complex relationships. As a result, a developer can create complex queries against objects cached as part of Power Tier.

> "In terms of **queries,** what **Power Tier** does is **create the relationships** between the **objects** and store these **complex relationships**"

JDJ: *What service do you need at the back end to support your software?*
Hietter: Power Tier runs on Solaris, HP, UNIX, IBM AIX, and Windows NT. As for the databases it automatically supports as part of container-managed Persistence, it's the usual suspects – DB2, Oracle, Informix, Sybase, Microsoft and SQL Server. That's it, basically.

JDJ: *You pretty much covered all of the big boys there.*
Hietter: Right. The other thing that I think is carrying Persistence forward is the fact that we're able to cache these entity beans and session beans so that they're in memory and available for rapid access. You aren't necessarily making trips to the database to satisfy queries from the session bean requests.

JDJ: *You've already got something like a built-in optimization layer there that the developer will essentially get free of charge?*
Hietter: Yes. Part of our focus is to minimize the interaction with the database layer. If there are changes occurring to the entity beans, we can batch and write those so that it's not a serial update-by-update operation. They can be batched, then through a batch operation go ahead and update the relational database.

JDJ: *Tell us, what kind of clients are using this infrastructure at the moment?*
Hietter: There's a very large and important one in London, Instinet. As you know, Instinet is handling their equities with a group in New York City. They handle about 30% of NASDAQ's trading volume on a daily basis. These people know all about trading systems. There was a division in London that saw an opportunity to take that expertise in trading systems and apply it to the fixed income or bond market. About a year and a half ago Instinet began an effort to create a Java-based trading system for bond traders that will handle Euro bonds, U.S. government bonds and so forth, and it's designed to operate in a distributed fashion where you have a trading environment based in London with replica servers in New York, Tokyo and the trading offices of Instinet's customers. You have this trading system running on top of this EJB application server and the caches – remember, the Power Tier application server is hosting all the information about the financial instruments, the latest bid and ask, and the trades that occur. You have caches of the market (the market being the instrument, the bid and ask) replicated throughout the world now. The caches are synchronized across Power Tier servers through a Power Tier feature called Power Sync as the synchronization mechanism between the replica servers. That's another key advantage of Power Tier – it supports these global distributed kinds of applications where you have many users looking at the same information simultaneously without creating a huge bottleneck in one particular geographic location. This is an example of one customer that's using Power Tier for real world serious applications.

JDJ: *And that needs to stay up 24 hours a day?*
Hietter: Well, it's a worldwide trading system and it does stay up 24 hours a day. The kind of transaction volume that this was designed for is a thou0sand transactions a second. It's a high-volume environment and that's the area where Power Tier seems to excel.

JDJ: *Where can we find out more information about Persistence?*
Hietter: For more information you can visit us at our Web site, www.Persistence.com.

Software Ag

www.softwareag.com

Sterling

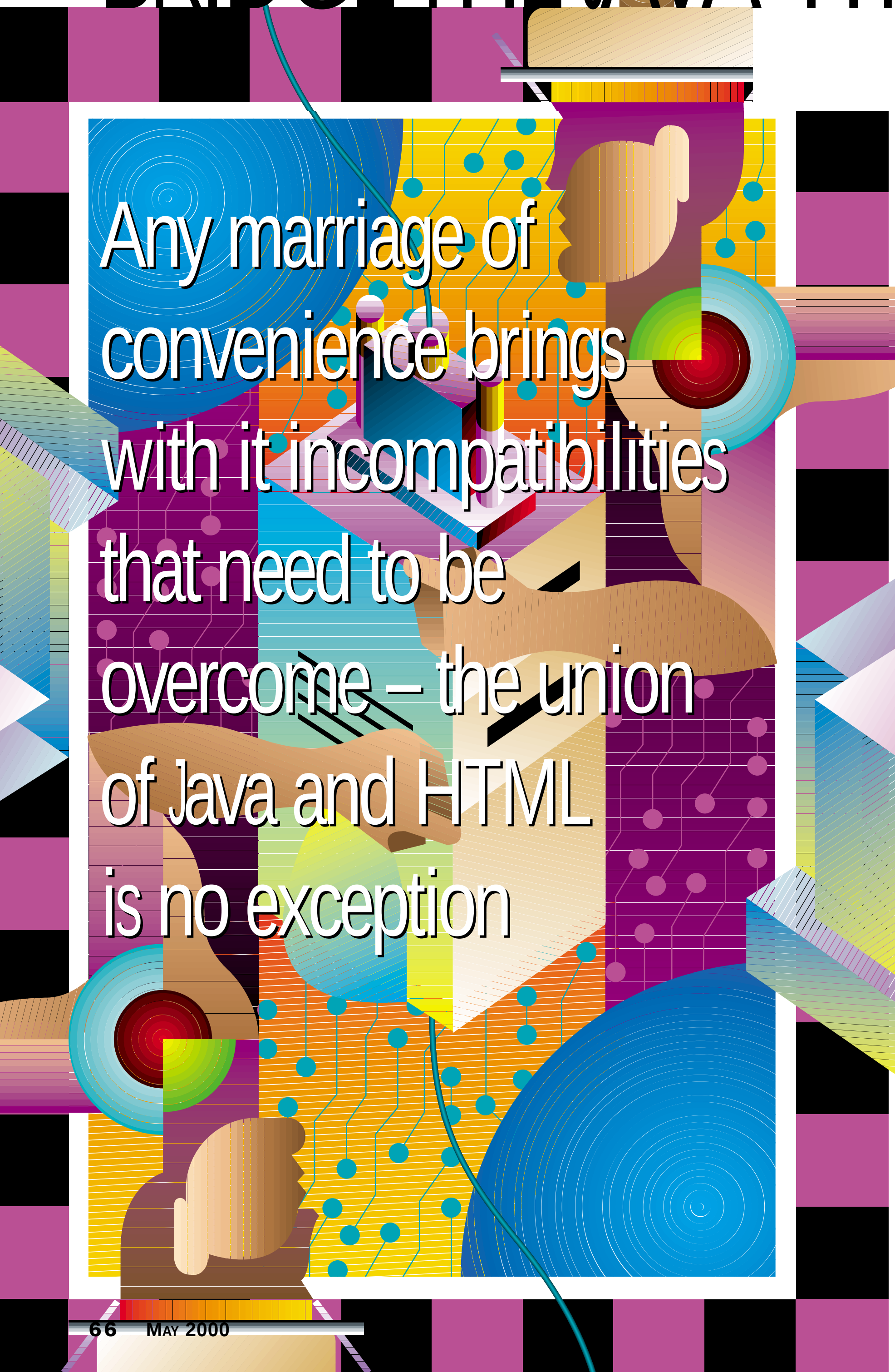
www.cooljoec

Software

challenge.com

PRESENTATION LOGIC AND EJBS: USING SESSION BEANS TO HELP BRIDGE THE JAVA–HTML GAP

WRITTEN BY MICHAEL LACY



Any marriage of convenience brings with it incompatibilities that need to be overcome – the union of Java and HTML is no exception

With the proliferation of Java-based application servers at the core of today's Web applications, the preferred Web architecture that has emerged places Java in the middle tier, gathering data from myriad sources, and HTML presenting that data through a Web browser.

As developers, we have a number of options for merging the two (Java and HTML, that is): JavaServer Pages (JSP), JHTML, Java Servlets and other proprietary APIs specific to vendors' application servers. But whichever path we choose, we all eventually struggle with the same challenge: keeping the Java and HTML code separate. Whether we're writing Java code directly into JSPs and JHTML pages or inserting HTML code in servlets, the merging of Java and HTML frequently create development headaches and, occasionally, strife between Java and HTML programmers during the development process. Once the application is finished, our headaches turn into migraines upon the realization that we have to dig through all of the intermingled code for maintenance and enhancements.

This article provides some relief for those headaches in the form of presentation logic. After we look at why there's this chasm between the two languages, and its implications, I'll demonstrate how to use session beans from the Enterprise JavaBeans (EJB) specification in conjunction with the Model-View-Controller design pattern to solve some of our problems. Then I'll give you my thoughts on bridging the Java–HTML gap in the future.

VSI

Web Application Architecture

Before we go about building the bridge, let's take a look at the source of the problem: the typical Web application architecture (see Figure 1).

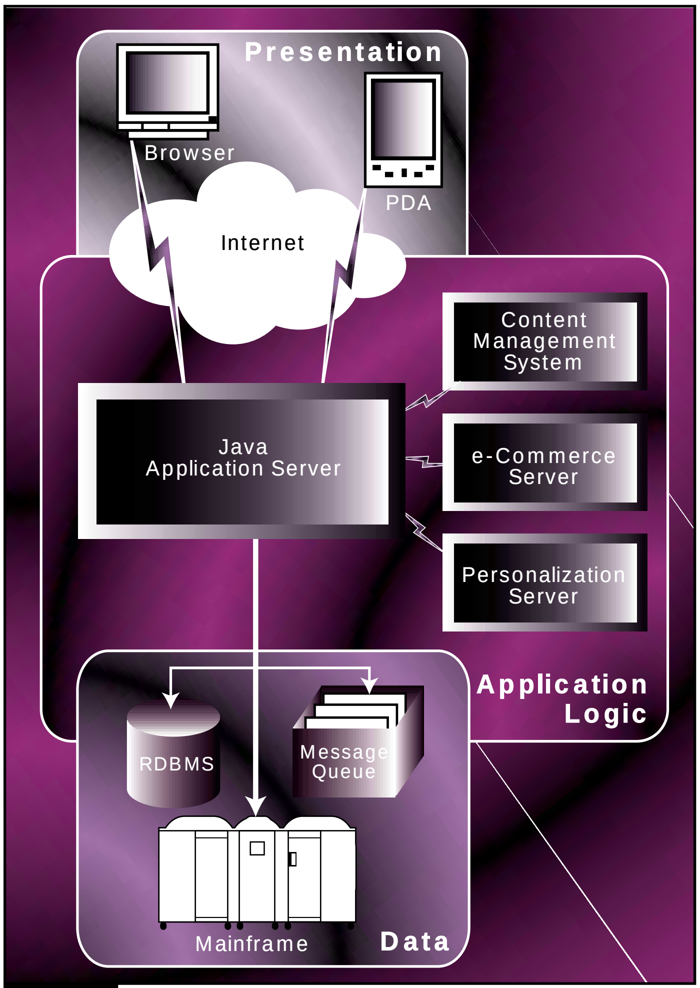


FIGURE 1 Typical Web application architecture

Over the past few years, Java application servers have increasingly been positioned at the core of many Web applications' middle tier. Application servers are responsible for accumulating data from a number of sources so as to create and deliver dynamic Web pages to HTML browsers. With the support that Java-based application servers offer for rapid development and access to countless numbers of application services and data sources, it's no surprise that their use continues to grow. For any given Web application, it's likely that business objects of all shapes, sizes and flavors are created with the ultimate goal of presenting their attributes in an HTML page.

However, Web browsers don't display Java objects. They parse and display HTML documents, which, as we all know, are character-encoded files. And let's not forget that HTML provides no direct APIs for manipulating and processing the data contained within its pages as requests are sent to the application server; rather, it's accomplished indirectly using the data encoded in an HTTP request.

The Java—HTML Gap

Now that we've identified the gist of the problem, let's see why it exists. Starting with a quick definition of each of these languages:

- ***Java:*** According to Sun Microsystems, Java is "a simple, object-oriented, network-savvy, interpreted, robust, secure, architecture-neutral, portable, high-performance, multithreaded, dynamic language." It's safe to assume that we all know what Java is (or else we wouldn't be reading this magazine!).
- ***HTML:*** HyperText Markup Language has established itself as the lingua franca for publishing content on the Internet. It's a nonproprietary language based on Standard Generalized Markup Language (SGML) which, to all intents and purposes, means that it's a tag-based language for marking up text and graphics on the Web and linking pages together via hyperlinks.

In layman's terms Java is the language used for implementing business logic and connecting to back-end data sources and legacy systems, while HTML is the language of choice for creating the user interface of a Web application. The interdependencies between the two are apparent: Java code processes business logic and HTML code displays the results. However, as the two languages evolved independently of one another, little thought was given to their marriage in Web applications until recently.

Looking at some of the specific problems we're trying to solve will help crystallize the issue at hand. Keep in mind that we're walking down a two-way street. Data is not only sent from the server to the Web browser for display, but is also entered by a user in a Web page and sent to the application server for processing.

There are different complexities associated with both routes. Following are some typical steps that occur as users interact with Web pages:

FROM THE BROWSER TO THE APPLICATION SERVER

1. ***Form submission:*** Users enter information into an HTML form and click a submit button. The data is marshaled into name-value pairs and sent via an HTTP request to the Web server. The Web server, in turn, forwards the request to the application server.
2. ***HTTP data extraction:*** Using APIs supported by the application server, data is extracted from the HTTP request in the form of Java strings. For example, the Servlet API provides access to an HttpServletRequest object on which we call the getParameter() method to retrieve data from the HTTP request. This step often involves the hard-coding of the name attributes of form fields directly in Java code.
3. ***Form field validation:*** Before setting business object attributes and sending updated information to a database, we often want to check the validity of the information entered by the user through the Web page. One example is checking that dates are specified in the format of MM/DD/YYYY and that the month, day and year are all integers.
4. ***Set business object attribute(s):*** Once the data has been validated, business object attributes are set and the application uses the updated business objects as defined in its workflow.

FROM THE APPLICATION SERVER TO THE BROWSER

1. ***Workflow:*** As seen in the previous example, several steps are required to process an HTTP request and update the business entities of the application. Once this process is complete, the application now needs to decide what to do next. If the user-entered data is invalid, then an error page is displayed. If the data is valid, the application needs to determine what to display next.
2. ***Business object instantiation and data retrieval:*** A business object containing the data that will eventually be displayed is needed. Where the business object acquires its data is inconsequential provided that the appropriate data is readable. We can even reference the business objects updated in the previous steps as long as they are still in memory.
3. ***Conversion to "Web-friendly" strings:*** Since HTML pages are entirely character-encoded, we'll need to convert the nonstring data to "Web-friendly" strings. This includes converting Java primitives and objects (such as java.util.Date) to strings.
4. ***Populate HTML pages with dynamic data:*** The final step in this process is to construct an HTML page with the dynamic content that's accumulated and converted in the previous steps.

The next section looks at the different approaches to populating HTML pages with dynamic content.

Youcentric

www.youcentric.com/nobrainer

Mixing Apples and Oranges

Ever since CGI scripts enabled the creation of dynamic Web pages, the complexity of Web applications has continually increased, establishing the need for powerful server-side programming solutions capable of gathering data from a countless number of sources and processing it. Enter Java. With the robust features of Java and the ubiquity of HTML as the window to the Internet, application servers began to cultivate the relationship between Java and HTML with several different solutions emerging:

- ***Servlets:*** Thought of as a replacement for CGI scripts, servlets burst onto the Java development scene in 1997 promising all the benefits of the Java programming language as an extension to the Web server. Servlets are essentially HTTP request handlers that provide full access to Java APIs and endow Web programmers with a powerful tool for creating dynamic Web pages. To display a Web page, HTML tags are encoded into the servlets, with dynamic content added as necessary. Servlets were the first Java–HTML solution that began to appear in industrial-strength application servers and are in wide use today.
- ***Proprietary HTML Rendering Engines:*** As Java application servers became more widely used, the limitations of the servlet model became apparent, especially with the increasing size and complexity of Web applications and development teams. HTML programmers were responsible for building the front end of the Web application in HTML, while the Java programmers wrote code for the middle-tier business logic and back-end data sources. Embedding HTML within Java Servlets was no longer viable. Proprietary tagging schemes and Java data structures were created by some of the application server vendors as an attempt to separate Java and HTML as much as possible. Java programmers would populate data structures with the dynamic content of a Web page, and an HTML rendering engine would merge that data into proprietary tags in HTML templates. If you ever worked with Netscape application server in its AppLogic days, you know what I'm talking about.
- ***JavaServer Pages (JSP):*** Recognizing that the separation of Java and HTML in a nonproprietary and standardized format was highly desirable, Sun Microsystems led an industry-wide, collaborative effort to develop an extension to the servlet specification that provided an XML-like scripting syntax for incorporating dynamic content from Java components into Web pages. JSP offers the most promising solution to date for componentized development and separation of business and presentation logic. However, realizing the benefits of JavaServer Pages requires a sound architectural design.

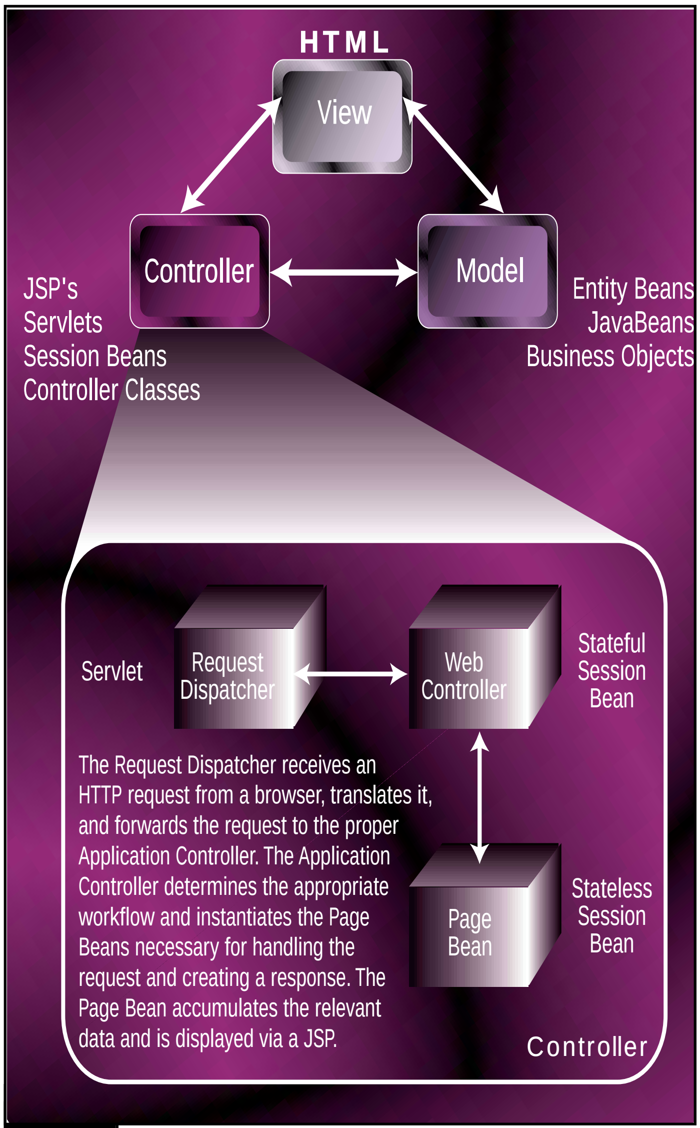
Presentation Logic

A quick recap of what we've looked at so far. First, with the explosion of *n*-tier Web applications using Java application servers at their core, we're continually building systems that connect to myriad data sources through Java APIs and displaying that information via HTML pages. Second, utilizing HTML as the user interface through a Web browser carries several implications from a programming perspective, primarily focused on how data is moved back and forth between Java and HTML. Third, in order to address the issues of the Java–HTML gap, competing technologies including servlets and JSP have been developed. Fourth, without an established framework for developing with these technologies, Java and HTML code are often mixed together, resulting in a nightmare of code development, flexibility, scalability and maintenance.

Building a presentation logic framework is essential for bridging the Java–HTML gap in our efforts to build successful Web applications that are both scalable and flexible. The remainder of this article illustrates one of several reasonable presentation logic frameworks based on the Model-View-Controller design pattern.

Model-View-Controller

Let's refresh our memories on the Model-View-Controller (MVC) design pattern. It's commonly used for designing GUI applications and is the underlying architecture for the Java Swing components. In the simplest sense, an application stores its data in some format (the model) but has different ways of displaying it (the views). The controller portion of the application serves as the central point of control, coordinating a user's actions with the views to update the model(s) and ensuring that the views are supplied with the latest additions to the model(s). Okay, enough with the abstract talk, let's dive into the specifics and see how the MVC pattern translates into the core components of a Web application. (Take a look at Figure 2.)



HTML
View
Controller
Model
JSP's
Servlets
Session Beans
Controller Classes
Entity Beans
JavaBeans
Business Objects
Servlet
Request Dispatcher
Web Controller
Stateful Session Bean
The Request Dispatcher receives an HTTP request from a browser, translates it, and forwards the request to the proper Application Controller. The Application Controller determines the appropriate workflow and instantiates the Page Beans necessary for handling the request and creating a response. The Page Bean accumulates the relevant data and is displayed via a JSP.
Page Bean
Stateless Session Bean
Controller

FIGURE 2 MVC architecture

- ***Model:*** Business objects hold the data needed by the application and play the role of model in the MVC pattern. With a Java application server, business objects are often built using Enterprise JavaBeans, JavaBeans and other Java classes. They represent the business entities of the system (e.g., User, Account).
- ***View:*** The user interface of a Web application represents the view. For simplicity's sake we'll restrict the user interface to HTML pages viewed through a Web browser (although it's entirely conceivable that the UI could be an applet, a PDA, and so on). Note that the view does *not* include JSP pages but rather the end product of processed JSP pages (which is, of course, HTML).
- ***Controller:*** Everything between the business objects and Web pages in a Web application can be considered controllers. This includes Java classes, servlets and JSPs. Think of it as all the Java code to process incoming data, delegate workflow and populate pages with business object data.

StarBase

www.starbase.com

In Control

We already discussed steps that occur when data is moved from the browser to the application server and vice versa; the question is how to develop presentation logic that's efficient and scalable with the goal of eliminating the jumble of Java and HTML code. Breaking out the controller into a group of discrete components with specific functions to perform will help answer that question.

- ***Request Dispatcher:*** As the central point of control for a Web application, the Request Dispatcher is a Java servlet that receives all incoming HTTP requests and forwards the HttpServletRequest and HttpServletResponse objects to the appropriate Web controller. Exactly how this is accomplished is a design issue that's far too lengthy and complicated to address here. The Request Dispatcher also performs application-wide functionality such as user authorization and session management.
- ***Web controller:*** Web controllers are the glue that hold a Web application together. They are session beans and can be stateful or stateless depending on the needs of the application. The primary responsibility of a Web controller is to process the business logic defined for workflows within functional subsets of the application. For example, there's a Web controller for login, registration and shopping cart. Some of the logic Web controllers perform includes session data management, exception handling and workflow.

At a very basic level, a Web controller is instantiated via the Request Dispatcher and is forwarded the HttpServletRequest and HttpServletResponse objects. Reading data from the HTTP request, the Web controller identifies the page the request came from and forwards the request to the corresponding page bean. Once the page bean is finished with processing the incoming request data, the Web controller regains workflow control and instantiates a page bean for the next Web application page to be displayed. If an exception occurs at any point in the process, the Web controller catches the exception and displays an informative error page using the HttpServletResponse object. Listing 1 shows a representative pseudocode for a Web controller.

- ***Page bean:*** Page beans are session beans conforming to the JavaBeans standard (i.e., they have private attributes with appropriate get/set methods for the purpose of referencing from a JSP). For each page of the application requiring interaction with the application server, there is a corresponding page bean. Let's say we have a user account maintenance page to display. There will be an analogous page bean that populates itself with the user data the page needs as well as processes updates to the user's information once the page is submitted. The page bean is the Java equivalent of the HTML page and processes information both going to and coming from the Web browser (at different times, of course). The superclass of all page beans contains utility methods for common functionality such as conversion of Java primitives to strings, HTTP data extraction and form validation. Over time, page beans can be grown to accommodate a number of different user interfaces rather than strictly HTML (more on that later). As the implementation details are lengthy enough for a book, we'll reserve that discussion for some other time.

Why Session Beans?

I know what you're thinking: why use session beans? The way I think of them is as replacements to the service(), doPost() and/or doGet() methods of a typical servlet. Their primary task is processing the presentation logic in a manner that's as independent of the user interface as possible. In the preceding examples, I realize that I've focused on what happens as requests are sent in from a Web browser. In these instances, servlets make sense. Down the road, as requests are dispatched from PDAs, pagers and toaster ovens, the servlet model makes less sense. A client-agnostic method for processing incoming requests and outgoing responses in a Web application environment provides greater flexibility for future growth. With session beans we also have the ability to distribute the aforementioned processing across many systems as well as perform session management without having to rely on cookies or URL rewriting. Remember, it's the architectural framework that's vital to ensuring the health of development efforts as the system grows in functionality and complexity.

Making It Work

Okay, so we have all these request dispatchers, Web controllers and page beans, but how are we going to get this architecture to work? With any Web application the possibilities for implementing presentation logic that varies in detail and scope are limitless. What's important is realizing the value of separating your Java and HTML code. In a perfect world, using today's technologies, HTML developers will create the look-and-feel of the pages, in which Java developers will then insert JSP tags to incorporate dynamic content. The only overlap between the two will be the JSP tags. However, this utopia is achievable only through intelligent application and presentation logic design that is enforced throughout the development process. This means taking into consideration details such as data that needs to be moved between screens and the mechanism for instantiating a page bean for a given page. What's presented above is just one of many viable solutions that provide the flexibility and scalability we're all looking to achieve.

The Future of Presentation on the Web

All of us who have built Web applications know that the Java–HTML solution is, at best, far from elegant. Invariably, Java and HTML will be mixed with each other, creating a muddle of confusion that will be called a Web application at the end of the day. With a solid presentation logic design fleshed out from the beginning of a Web application project, some of the obstacles imposed by the Java–HTML gap can be overcome, but never will all of them be eradicated. What's needed is an intermediary step between the Java and HTML code. My opinion is that the XML (eXtensible Markup Language) is the answer.

Without going into a great amount of detail, the ideal Web application development environment I envision is one in which Java application servers sit in the middle tier churning through business logic as requests come in and creating XML documents on the fly as responses. With XML documents being sent to clients, we don't have to worry about the details of transporting data from our Java code directly into HTML. That becomes the job of the XML Stylesheet Language (XSL) and an XSL processor...a true separation of content and its layout. With this technique the data required for a page is moved around in a platform-neutral and well-defined format that can have its view changed as required depending on the device users are accessing it from. I can't wait!

AUTHOR BIO

Michael Lacy is a senior engineer at Modem Media in San Francisco, California, where he leads engineering teams in the development of Web applications for Modem's clients. He holds a BS in biomedical engineering and electrical engineering from Duke University.

michael_lacy@yahoo.com

Listing 1

```
try {
 Hashtable h1 = new Hashtable();
 Hashtable h2 = new Hashtable();

 // populate hashtable with data not in request object
 // that is necessary for processing

 PageBean1 pb1 = new PageBean1(request, response, h1);
 pb1.processIncomingRequest();

 // workflow to determine next page to display
 // populate hashtable 2 with data needed for processing

 PageBean2 pb2 = new PageBean2(request, response, h2);
 pb2.processOutgoingResponse(request, response, h2);
} catch (Exception e) {
 // handle exceptions
}
```

Career Central

www.careercentral.com/java

Keeping Up with OMG

An up-to-the-minute report from the latest Object Management Group meeting

WRITTEN BY
JON SIEGEL

OMG members met in Denver, Colorado, the week of March 6–10, 2000. In this column I'll summarize as many of the various efforts that progressed as space permits.

Brief Background on the OMG

CORBA and Java enjoy a special relationship. Although CORBA defines an environment in which many languages interoperate on an equal footing (including Java, of course), when you get beyond basic interoperability you find that – because of the similarity of the Java and CORBA object models – Java enjoys advantages that other languages don't:

- The reverse Java-to-IDL mappings that turn Java RMI objects, as if by magic, into CORBA objects.
- The isomorphic relationship between basic-level CORBA components and EJBs.

Because of this similarity, new developments in the CORBA world help move the Java world along too, and Java developers need to keep abreast of developments at OMG as they move along.

While almost everyone knows that the Object Management Group (OMG) produces the CORBA specifications, they don't necessarily know, for example, that OMG produces specifications in many other areas, ranging from analysis and design – the Unified Modeling Language (UML), the Meta-Object Facility (MOF), XML Metadata Interchange (XMI) and the new Common Warehouse Metamodel (CWM) – to the CORBA infrastructure and domain (that is, vertical market) specifications in healthcare, finance, telecommunications and more than a half-dozen other areas.

OMG specifications are produced by the members, not by staff, with most of the cooperative work happening at member meetings. OMG meetings are a big deal. Because of the way the OMG technology adoption procedure encourages consensus-building among the large number of task forces working in different areas, the meetings typically attract 500 to 700 people for a full week and take place five times a year in various locations around the world. To take full advantage of this "face time," members do a lot of preparation between meetings: documents are drafted and shared via OMG's document server, and discussions take place via e-mail, teleconference and occasionally even in small groups that get together in person between meetings.

No longer able to fit into five weekdays, the meetings themselves now start on the Sunday before with tutorials and meetings of smaller groups. During the week itself, many people get together at breakfast time to get an early start, while informal contacts continue through dinner and into the evening. (The bar is a favorite place for evening get-togethers; at a meeting in Dublin recently one wag suggested that OMG really stood for "Order More Guinness!")

At the meetings new technology adoption efforts are started and existing ones either progress or conclude with the adoption of new specifications. Because each meeting produces a lot of milestones in a wide range of areas, a postmeeting wrapup provides a good opportunity to summarize OMG efforts to nonmembers; every reader learns the areas where standards-compliant products will appear on the market not far in the future, and some (you too, perhaps?) will become interested enough to come to one of OMG's next meetings. If you already work for one of the nearly 800 member companies (listed at www.omg.org/cgi-bin/membersearch.pl), you can register for the meeting at OMG's Web site and just show up; if you don't, you're invited to attend as an observer – just send me an e-mail (siegel@omg.org) for an invitation.

New Technology Adoptions

Because new specifications are the most newsworthy developments, let's start at the end of OMG's process and work our way around to the beginning. To ensure OMG member consensus, every technology adoption concludes with a series of four votes. These votes can't start until the submission/evaluation process has concluded and the proposed specification has stabilized; even if you're not an OMG member, you can download and read the likely future specification in its final form at this stage. At the meeting the task force that initiated the adoption votes to recommend adoption, the architecture board certifies that the submission is consistent with OMG's Object Management Architecture, and existing specifications and the responsible Technology Committee (either Platform or Domain) starts an e-mail poll of its members – including those who aren't present – lasting for about eight weeks. When the Technology Committee vote concludes, a final vote by OMG's board of directors formally declares that the submission is an official OMG-adopted specification.

New Technology Procedure

At the Denver meeting, four new technologies passed through task force and architecture board votes and started votes in the technology committees. They are described as follows.

COMMON WAREHOUSE METAMODEL

Data warehousing improves business decisions by facilitating access to timely data. To make the best use of data warehousing, companies must integrate data from multiple warehouses across their enterprise, but inconsistencies in the data model from one warehouse to another make this difficult. Because these inconsistencies result from real differences in requirements from one business unit to another, it's not possible to standardize on a common data model for data warehousing. However, by defining a standard for interchange of information about data models, termed *warehouse metadata* in the

Evergreen Internet, Inc.

www.evergreen.com

specification, OMG's new standard for a Common Warehouse Metamodel (CWM) provides standard ways to run processes that integrate data from multiple warehouses. Coming from OMG's Analysis and Design task force, this specification builds on UML and the MDF rather than on CORBA. Modeling and Metamodeling, although recent additions to OMG's suite of specifications, are crucial to the kind of large enterprise applications that CORBA does well and so receive a great deal of attention.

CLINICAL IMAGE ACCESS SERVICE

CORBAmed, OMG's task force for healthcare, recommended adoption of a Clinical Image Access Service (CIAS) that standardizes access and retrieval of clinical images for nondiagnostic use. Jointly written and submitted by Siemens Health Services and Philips Medical Systems, the new specification provides an important supplement to the well-established DICOM (Digital Imaging and Communication in Medicine) standard that covers diagnostic image transfer.

ELECTRONIC COMMERCE AND TELECOMS

In addition, the Electronic Commerce Domain Task Force recommended a specification for a CORBA-based Public Key Infrastructure, and the Telecommunications Domain Task Force recommended a specification that supports telecom network management with file transfer functionality.

File-based functionality is a standard part of telecom network management systems, making use of FTAM or FTP interfaces. This specification defines a single set of IDL interfaces providing a CORBA view of the files that's the same regardless of whether the underlying files are FTAM- or FTP-based. Even though it abstracts the complexity of the various underlying file systems into a single set of interfaces, it's not a generalized File server application.

Maintenance Revisions

Maintenance revisions get a lot less press than new specifications, but at OMG these important efforts get a lot of attention and work and go through the same voting process. At the Denver meeting eight revisions passed through task force and AB votes. They covered CORBA interoperability, mappings from IDL to Java and C++ as well as from Java to IDL, the Firewall specification, the Distributed Simulation Facility, Workflow Facility and, in the telecommunications domain, CORBA/IN Interworking.

Board of Directors Actions

The board of directors vote that concludes the process is on the one hand significant – it's the step that actually creates a new specification – and on the other hand somewhat anticlimactic, because everyone has known what the standard would be for months. In fact, because people are so anxious to find out about new specifications, OMG typically announces important new specifications during their TC vote without waiting for board confirmation. Nevertheless, this remains an important formal step, and a large number of new and newly revised specifications were created by the board when they met in Denver. These included Enhanced View of Time, Fault-Tolerant CORBA, Portable Interceptors (that enable security and technical functions), Management of Event Domains and a level 2.0 revision of the Ada language mapping. The board also ratified the 1.0 version of the Interoperable Naming specification, Revision 1.7 of CORBA security, and maintenance revisions of the Currency specification and the Person Identification Service (PIDS). Once specifications pass this final vote, they appear on OMG's "Newly Adopted Specifications" Web page (see www.omg.org/techprocess/meetings/schedule/tech2a.html).

"The board of directors vote that concludes the process is the step that actually creates a new specification"

Architecture Board Member Election

With two technology committees and 11 task forces creating specifications, OMG members realized years ago that the adoption process needed to ensure consistency across the wide scope of the specification landscape. To do this, OMG created an Architecture Board composed of 10 representatives elected from OMG's member companies and chaired by Andrew Watson, OMG's technical director and vice president of architecture. Technology adoptions kick off with the issuing of a Request for Proposals (RFP), which is the requirements document for the new specification. Before any RFP can be issued, the AB must certify that its requirements are consistent with OMG's Object Management Architecture (OMA) and all existing standards. In a similar step at the end of the adoption process, the AB must certify that the about-to-be-adopted specification conforms, in its own right, to the OMA and is consistent with existing standards.

The 10 member representatives on the AB are all volunteers who devote at least 20% of their time (and frequently more) to these duties. Since AB members serve only two-year terms, at least five seats come up for election every year. Resignations and job changes occasionally necessitate additional elections beyond these. (Each seat is assigned to an individual as a representative of his company, so members must resign when they change companies even if they move to a company that's a member of the OMG.)

At the Denver meeting OMG held an election for seven AB seats: reelected for two-year terms were Tom Rutt of Lucent Technologies, Jishnu Mukerji of Hewlett-Packard and Dave Frankel of Genesis Development, Inc. Newly elected for two-year terms were Peter Walker of Sun Microsystems, Inc., and Michi Henning of Object Oriented Concepts, Inc. Finally, Sean Baker of IONA Technologies Plc and Jeff Mischkinsky of Persistence Software Inc. were elected to one-year terms, filling positions vacated by members halfway through their terms. Continuing on the AB for another year are current members Carol Burt of 2AB, Inc., Sridhar Iyengar of Unisys Corporation and Nilo Mitra of Ericsson.

New Domain Technology Committee Standards Efforts

The Domain Technology Committee develops standards in vertical markets. The DTC has the business world well covered with nine task forces in nine domains:

- Business Objects
- Finance and Insurance
- Electronic Commerce

Inetsoft Technology Group

www.inetsoftcorp.com

- Manufacturing
- Utilities
- Telecommunications
- Healthcare
- Transportation
- Life Science Research

Special Interest Groups (SIGs) represent other technology areas including Human Resources, Retailing, Customer Relationship Management, and Space and Satellite Science.

Although all nine task forces are currently working on technology adoptions (and most are working on several at once), only one, the Life Science Research task force, which issued three new RFPs, started new work at the meeting. The following section describes each of them and shows how CORBA and domain knowledge come together at OMG to define services with a unique advantage.

CHEMICAL STRUCTURE ACCESS AND REPRESENTATION

Chemical compounds are the basis for biotech research, so you might figure that workers in the field had solved the problem of storing and retrieving information about chemical structure years ago. You'd be right if you figured that the problem had been solved many different ways, but wrong if you thought there was one common way, either standard or even commonly used. The newly issued RFP for a Chemical Structure Access and Representation facility will standardize OMG IDL interfaces to access, represent and manipulate both individual 2D chemical structures and collections of structures, such as structure databases, lists and virtual libraries – all commonly used in current research.

GENE EXPRESSION DATA

The second RFP will standardize computer representation and manipulation of gene expression data. Gene expression, a complex process that starts with transcription of the gene by a messenger RNA, proceeds through the production of protein defined by the gene and ends (typically) with the catalysis of a chemical reaction by the protein. As you might expect, scientists who study gene expression collect a lot of data in a lot of different forms and formats and analyze it using clever tools that, today at least, don't interchange data with one another. This RFP will define a set of interfaces, data structures and services that will allow gene expression systems to exchange data and, later, serve as a common basis on which advanced facilities and services can be built.

AUTHOR BIO
Jon Siegel is the director of technology transfer, Object Management Group.

ENTITY IDENTIFICATION SERVICE

Finally, in the field of Life Science Research scientists deal with countless numbers of things with attributes, filed away neatly in databases where they have to be identified with some degree of certainty in order to be retrieved and worked with. This is similar to the person identification problem solved by CORBAmed's PIDS specification, although not a precise analog. (The RFP points out both the similarities and differences between what it seeks and what PIDS provides. If the PIDS facility were suitable, the Life Science work would just use it and not define a new specification with duplicate functionality.) The Entity Identification Service will provide matches and confidence levels using a standard set of interfaces, useful in many ways both inside the Life Science suite of standards and outside it in other OMG task forces' work.

If your company wants to help write these new specifications, you can start by downloading the RFP documents from www.omg.org/schedule. You don't need to be an OMG member to download the RFP and read about it, but your company will need to join if you want to submit technology in response or to vote on the outcome. There are deadlines, too: initial deadlines appear on the RFP document itself, but they change frequently. The latest deadlines always appear at www.omg.org/schedule. For information on OMG membership, surf to www.omg.org/membership.

OMG members frequently ask the industry for information about the state of a software area before starting a standard-setting effort. The vehicle for this is the Request for Information. Two were issued at this meeting. Because anyone – member or not – may respond to an OMG RFI, you can respond to these yourself if they're in your area. One of the newly issued RFIs asks for information about the Knowledge Management Services Marketplace and Infrastructure; the other about Interoperability for Rail Software Systems. The RFIs are documents issued by OMG; you can read them and their brethren at www.omg.org/schedule.

New Platform Technology Committee Standards Efforts

The Platform Technology Committee, responsible for the core CORBA architecture as well as analysis and design, issued two RFPs. One, recommended by the ORBOS (Object Request Broker and Object Services) Task Force, will standardize CORBA support for data-parallel processing – that is, applications that partition processing into multiple parts, each working simultaneously on a subset of a large data volume. The other, recommended by the Analysis and Design Task Force, will supplement the A&D language UML with a profile for event-driven Enterprise Application Integration (EAI).

Tidbits

Lots of other things happen at OMG meetings. Groups are formed or rearranged and outside news is announced. Here's a brief potpourri from Denver:

- ***The Business Objects Domain Task Force*** has been spending more time on cross-domain technology efforts and less on "business objects," and is considering changing its name and charter to reflect this. A group started by members of manufacturing, working to synchronize representations of people across OMG specifications and nicknamed "People who like people," was moved into the Business Objects TF in anticipation of this move.
- ***CORBAmed*** announced that Brazil has adopted their specifications across that country, and also that they will sponsor a demonstration of interoperability of CORBA-based healthcare applications at the upcoming HIMSS conference.
- ***EPRI CCAPI:*** OMG frequently hosts colocated meetings of organizations that are working with OMG specifications or considering it. In Denver OMG hosted a meeting of the EPRI CCAPI (Electric Power Research Institute Command and Control Application Programming Interface) committee.
- ***The Life Science Research Group*** hears about a set of CORBA-based interfaces to lab instruments based on ASTM LECIS standard by Torsten Staab of LANL.

Date of Next Meetings

We haven't reviewed absolutely everything that happened in Denver, but we have covered the most newsworthy items – besides, we're out of space.

OMG's next meeting will be in Oslo, Norway, the week of June 12–16 (coincidentally, these dates are very close to "white nights weekend" celebrating the longest day of the year); following that, we'll meet in Burlingame, California, the week of September 10–15. We'd be pleased to see you there!

siegel@omg.org

Chicago Internet World 2000

JavaCon 2000 Spread

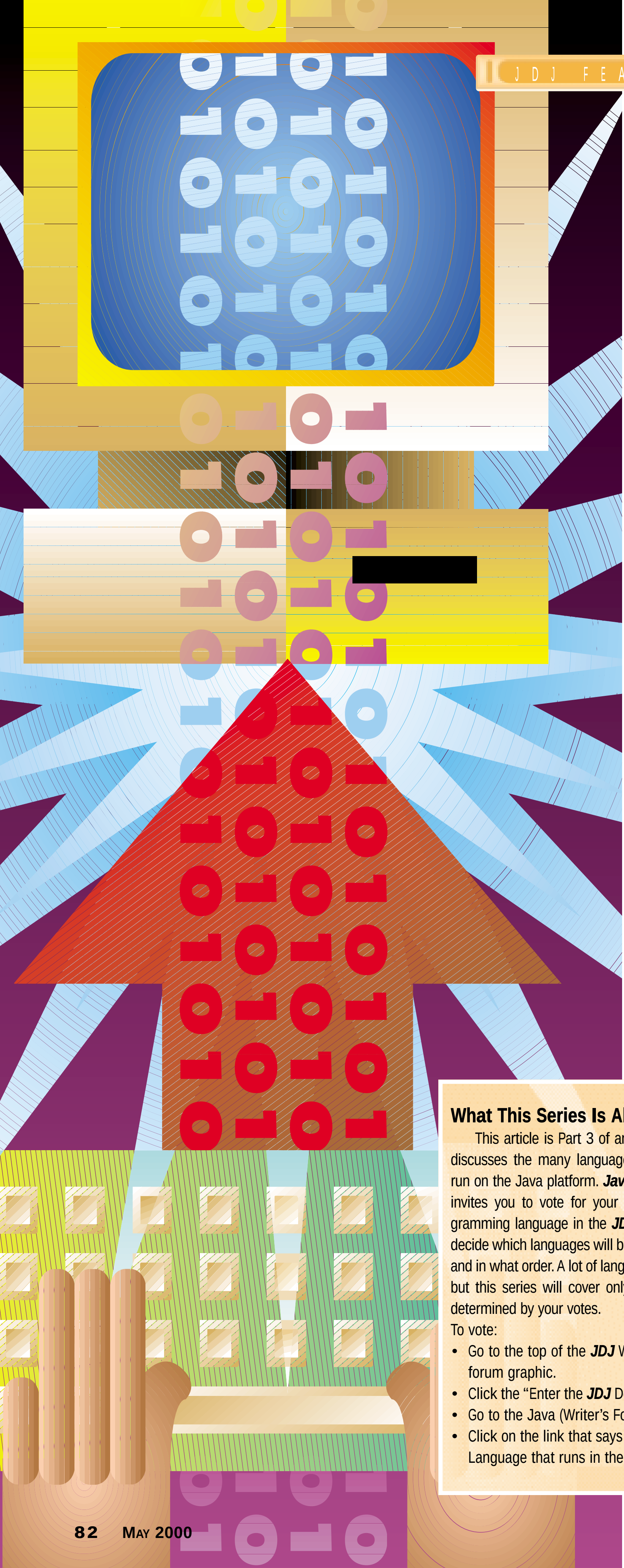
p/u from JDJ

www.javacon2000.com

JavaCon 2000 Spread

p/u from JDJ

www.javacon2000.com



NetRexx Programming for the JVM

NetRexx fills the gap between high-level languages and Java

WRITTEN BY RICK HIGHTOWER

What This Series Is About

This article is Part 3 of an interactive series that discusses the many languages that compile and/or run on the Java platform. ***Java Developer's Journal*** invites you to vote for your favorite non-Java programming language in the ***JDJ*** forum. Your vote will decide which languages will be covered by the series, and in what order. A lot of languages work in the JVM, but this series will cover only the most popular, as determined by your votes.

To vote:

- Go to the top of the ***JDJ*** Web page and click the forum graphic.
- Click the "Enter the ***JDJ*** Developer's Forum" link.
- Go to the Java (Writer's Forum) section.
- Click on the link that says "Vote for your Favorite Language that runs in the JVM."

This column is the resting place for non-Java–language, JVM-related topics for ***JDJ.*** It focuses on topics such as:

- *Creating JavaServer Pages (JSP) in JavaScript, WebI and Python*
- *Integrating with Tcl, Python and Perl scripts*
- *SWIG for Java*
- *Open source Java initiatives like EnHydra and Apache Tomcata*
- *COM/DCOM from pure Java*
- *CORBA to Legacy integration, etc.*

Let us know what else you want to see covered.

In the first two articles in this series Java was presented as the system language, and the higher-level language was presented as the glue language. This allows you to define frameworks, libraries and components in Java and glue them together to make applications. This was described in detail in the February ***JDJ*** (Vol. 5, issue 2).

Though it wasn't on the first suggested list, NetRexx received a lot of votes. Those votes count, as evidenced by the fact that NetRexx is covered here – the last time I checked it was running neck and neck with JPython.

NetRexx 101

NetRexx from IBM is a human-oriented language that's geared to Java. It marries REXX, one of the best-known scripting languages, with Java. Thus you get the readability and ease of use of REXX with the universal Java platform. NetRexx compiles to Java bytecode. Its claim to fame is quickly developed, maintainable code.

"NetRexx is a human-oriented programming language that makes writing and using Java classes quicker and easier than writing in Java," from www2.hursley.ibm.com/netrexx/. NetRexx is a REXX variant; REXX is a popular scripting language for IBM operating systems like OS/2 and increasingly for other platforms as well.

NetRexx is a lot like Java, although, unlike JPython, not more dynamic than Java. Instead, it's an easier syntax that makes system programming easier. NetRexx mixes nicely with Java – for example, it can subclass Java classes and implement interfaces. Also unlike JPython, which augments Java, NetRexx takes Java on head to head as an easy-to-use systems language while maintaining some of the features that made REXX programming popular. At first this may seem to put NetRexx at a disadvantage. However, NetRexx does a nice job of being a better REXX than REXX and a better Java than Java.

What does *human-oriented* mean? I was originally fooled into believing that NetRexx was a scripting language, like its parent. It isn't. It's described as making things easier on the programmer and harder on the compiler. You have the option of running NetRexx in a mode that's more like a scripting language or more like a systems language. This option manifests itself in the NetRexx syntax via the option keyword (similar to the option keyword in Visual Basic).

NetRexx advocates claim NetRexx is easier to learn than Java. I tend to agree, with some reservations. If you're a novice programmer, NetRexx is easier to learn than Java. Also, if you come from a Python, JavaScript, Visual Basic or Delphi background, then NetRexx would be easier to learn than Java. However, if you're from a C++ background, Java is probably easier to learn. Of course, if you're from a REXX or ObjectRexx background…well, you get the picture.

I found NetRexx extremely easy to learn and use. And the code I created was easier to read than the equivalent Java code. NetRexx has every language feature that Java has, plus some additional features that make JavaBean development easier.

Rosetta Stone

For comparison, each NetRexx sample application will have a corresponding Java implementation. This article covers the following sample applications (last month we compared JPython to Java):

- *A simple GUI application*
- *A simple statistics application*
- *Embedding the script into an application (if applicable)*
- *A simple example parsing text*

NETREXX 101: A SIMPLE CLASS

Listing 1 is a sample class in NetRexx and Listing 2 is the equivalent class in Java.

Use of the NetRexx class is the same as for the Java class. Let's break this down and compare it to the Java equivalent. To declare private instance variables in Java you'd do this:

```
private String firstName, lastName;
 private int id, dept;
```

In NetRexx you'd do this:

```
Properties Private
  firstName=String
  lastName=String
  id=int
  dept=int
```

To create a bean property in Java you'd do this:

```
private Employee manager;

public Employee getManager(){
  return manager;
}

public void setManager(Employee manager){
  this.manager=manager;
}
```

In NetRexx you'd do this:

```
Properties indirect
  manager=Employee
```

As you can see, NetRexx is much less verbose. The handling of properties reminds me of Delphi.

Instance variables and class variables are called *properties* in NetRexx. Note two things: (1) the use of the indirect keyword with properties, and (2) the call to ron.getManager(). The indirect keyword is a way to declare private variables that can be accessed outside of the class. In other words, the indirect keyword is shorthand for Java properties.

Meet JDJ

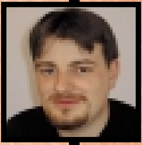
EDITORS AND COLUMNISTS

Attend the biggest Java developer event of the year and also get a chance to meet ***JDJ***'s editors and columnists!

JavaCON 2000

Sean Rhody
Editor-in-Chief, JDJ
Sean is the editor-in-chief of ***Java Developer's Journal.*** He is also a principal consultant with Computer Sciences Corporation.

Alan Williamson
JDJ Straight Talking Columnist
Alan is the "Straight Talking" columnist of ***JDJ,*** a well-known Java expert, author of two Java books and contributed to the Servlet API. Alan is the CEO of n-ary Consulting Ltd., with offices in Scotland, England and Australia.



Ajit Sagar
Editor-in-Chief, XML-Journal
Ajit is the founding editor of ***XML-Journal*** and a well respected expert in Internet technologies. He is a member of a leading e-commerce firm in Dallas, Texas, where he focuses on Web-based e-commerce applications and architectures.

Jason Westra
EJB Home Columnist
Jason is the "Enterprise JavaBeans" columnist of ***JDJ*** and CTO with Verge Technologies Group, Inc., a Java consulting firm specializing in Enterprise JavaBeans solutions.

MEETING
September 24-27, 2000
Santa Clara Convention Center
Santa Clara, CA

JAVA DEVELOPER'S JOURNAL CAMELOT COMMUNICATIONS

There are four forms of visibility for instance variables in NetRexx: public, indirect, inheritable and private. The NetRexx public and private visibilities are just like their Java counterparts. The inheritable visibility is like Java's protected visibility, and the indirect provides bean properties.

The indirect automatically provides assessor methods via the JavaBean "design patterns" for bean properties. If you add your own getter and setter methods to the class, they override the default. Actually, if you override your own, NetRexx won't generate the default assessor methods for that property.

Compare the listing for the NetRexx Employee class to the listing for the Java Employee class. The former class is shorter and in my opinion more readable. Can you imagine a class with five to 10 bean properties? The NetRexx class is significantly shorter.

NetRexx also has support for indexed properties. It takes up to four Java methods to do what one indexed property declaration does! If you use properties a lot, NetRexx is an easy sell.

To create an instance of an Employee and print it to the screen, you'd do the following:

```
say Employee()
```

The equivalent Java statement would be:

```
System.out.println(new Employee());
```

Next we create two instances of employee called Joe and Ron and print them to the console. We print Joe, who is Ron's manager, by invoking the getManager method of Ron. First in NetRexx, then in Java:

NETREXX:

```
 say Employee()
 joe = Employee Employee("Joe", "Batista", 100, null, 1);
 ron = Employee("Ron", "Furgeson", 101, joe, 1);
 say ron
 say ron.getManager()
```

JAVA:

```
 Employee joe = new Employee("Joe", "Batista", 100, null, 1);
 Employee ron = new Employee("Ron", "Furgeson", 101, joe, 1);
 System.out.println(ron);
 System.out.println(ron.getManager());
```

The syntax of NetRexx is close to Java (but actually closer to Python). One feature NetRexx has is default values. As mentioned in the March article on JPython, this feature can save some coding effort, not to mention some headaches. Have you ever had several different versions of the same method? And you just wanted to have different default values? Every default value is another overloaded method, which can get messy. I like that NetRexx has default values.

Rosetta Stone GUI

Now that we've created a simple class, we'll create a simple GUI. The employee class and the employee form examples are nonsensical – the idea is to demonstrate and compare NetRexx to Java. In the last article we implemented the EmployeeForm example in both Java and JPython, so you can compare NetRexx side by side with Java and JPython.

In the JPython article we covered the development of the GUI in the interactive interpreter, which is great for prototyping. This isn't currently possible with NetRexx, but Mike Cowlishaw, creator of NetRexx, is adding an interactive interpreter to NetRexx as the next feature.

The EmployeeForm is simple: it has two text fields and an OK button. The first text field is used for the name. The user enters the first and last names in the first text field. When the user hits the OK button, the EmployeeForm handleOkay method parses the text for the first and last name. Listings 3 and 4 show the NetRexx and Java versions of EmployeeForm.

The keywords are similar between the Java and NetRexx versions. For example, they both use import-to-import classes. A couple of key differences will be highlighted.

NetRexx has inner classes, called *minor classes* in NetRexx-speak. However, it doesn't have the concept of anonymous inner classes – I like anonymous classes for handling methods. Thus I had the EmployeeForm implement the ActionListener in the NetRexx version. However, in the Java version I used an inner class.

Thus the NetRexx event handling looks like this:

```
class EmployeeForm extends JFrame implements ActionListener
…
 method EmployeeForm
…
  okay.addActionListener(this)
…

 method actionPerformed(event=ActionEvent)
  handleOkay()
```

While the Java version looks like this:

```
 public EmployeeForm(){
…
  okay.addActionListener(new ActionListener(){
   public void actionPerformed(ActionEvent event){
    handleOkay();
   }
  });
```

Of the two approaches I prefer Java because I don't like to clutter the interface of my class with unnecessary methods (like actionPerformed). I could have declared an inner (minor) class in the NetRexx version and then used the minor class as the event handler, but I honestly don't think I'd bother. I like anonymous inner classes and am quite spoiled using them.

The Java and NetRexx versions of the EmployeeForm parse the string from the name text field in entirely different ways. The latter uses the REXX object, which is a cross between a string, a collection, an associative array (like Perl) and a generic data type. I'm not sure if I love the REXX object or hate it, but it sure makes parsing strings easy. Here's the NetRexx version of parsing the name.

```
  _name = Rexx(this.name.getText())
  fname = _name.word(1)
  lname = _name.word(2)
```

Compare the ease of the above approach to the Java version of the same routine.

```
  name = this.name.getText();
  index = name.indexOf(" ");
  fname = name.substring(0, index);
  lname = name.substring(index+1, name.length());
```

Of the two, I prefer the NetRexx approach – the REXX object is growing on me.

Rosetta Stone Statistics

Just to highlight how well the language can do simple, common things, we'll create an easy application that calculates statistics for house prices in a neighborhood. We'll create a program that gives a list of number finds: the averages (mean, mode, median) and range. We'll list each function's code, then break it down and describe the function line by line.

IMPLEMENTING GETRANGE

Since getRange is the easiest, we'll do it first. Essentially, we want a function that returns the minimum/maximum values in a list and their range.

First try implementing getRange (see Listing 5), which uses the java.util.Collections class to get minimum and maximum value. When the getRange function is done it returns the min, max and the range in an ArrayList instance.

Thus, to declare the min, max and ranges variable in Java, do this:

```
  double min, max;
  ArrayList ranges;
```

In NetRexx, do this:

```
 min=double
  max=double
 ranges=ArrayList
```

Note that in NetRexx declaring the variable before assigning it is an optional step, as we'll show later.

To get the value min and max values from the min and max methods of the java.util.Collections class, do this in Java:

```
  min =
((Double)Collections.min(nums)).doubleVal-
ue();
  max =
((Double)Collections.max(nums)).doubleVal-
ue();
```

In NetRexx you'd do this:

```
 min = ( Double Collections.min(nums)).dou-
bleValue()
 max = ( Double Collections.max(nums)).dou-
bleValue()
```

This is fairly similar to the Java way. (The Java versions of Listings 5–8 can be found on the ***JDJ*** Web site in the March 2000 [Vol. 5, issue 3] digital edition, www.JavaDevelopersJournal.com.)

IMPLEMENTING GETMEAN

The getMean function figures out the mean of a sequence of numbers. It iterates through the list, adds all the values together and stores them in a sum. It then figures the mean by dividing the sum divided by the length of the sequence of numbers. The getMean sample (see Listing 6) uses an argument called *sample* to determine if this is a sample mean or a population mean.

This example shows sample usage of:

- *Loop while*
- If *and* else *statements*
- *Default argument*

Let's break down the getMean function step by step. First notice the method declaration:

```
method getMean(nums=ArrayList,
sample=boolean 1) public static returns
double
```

This method is passed an ArrayList (nums) and a boolean flag (sample). Notice that the visibility and modifiers are at the end. Also notice that the sample argument has a default value. Thus, if the second argument isn't specified when calling this method, the value of 1 (true) will be implied.

The equivalent Java code would look like this:

```
 public static double getMean (ArrayList
nums){
getMean (nums, true);
}

 public static double getMean (ArrayList
nums, boolean sample){
 …
 }
```

In Java you'd have to declare two methods. As you can see, if you had a method with four default values, the equivalent Java code could get quite verbose.

The getMean method defines three local variables called *average*, *sum* and *value*, which hold the sum, average and current iteration value.

```
 value=Double
 sum=0.0
 average=0.0
```

To iterate through the ArrayList we need to get the iterator. The method gets an iterator from the ArrayList argument passed to the method as follows:

```
 iterator = nums.iterator()
```

Notice that we didn't declare the iterator first.

Now we can use the iterator to iterate through the numbers in the nums ArrayList while adding the numbers together.

```
 loop while iterator.hasNext()
  value = Double iterator.next()
  sum = sum + value.doubleValue()
 end
```

To cast the value from the iterator.next from an Object into a java.lang.Double we do this:

```
  value = Double iterator.next()
```

The equivalent Java code would look something like this:

```
  value = (Double) iterator.next();
```

Next, the getMean method checks to see if this is a sample mean. If it is, the method figures the average by dividing the sum by the

Embarcadero p/u

www.

embarcadero.

com

number of items in nums less one, or else it divides the sum by the number of items in the nums ArrayList.

```
  -- Check to see if this is a sample mean
 if sample then
  average = sum / nums.size()-1
 else
  average = sum / nums.size()

 Last, the method returns the average:

return average
```

The foregoing is not much different from what you'd do in Java. It's actually very similar. In fact, you could probably read and understand the NetRexx code without much coaching.

IMPLEMENTING GETMODE

The getMode function (see Listing 7) finds the value that repeats the most. (*Note:* This is not a complete implementation of mode because it works only with discrete values.) The first thing this function does is duplicate the ArrayList because it's going to modify it. Then the method iterates through the items in the nums sequence and counts the number of occurrences of the current items (we use the built-in sequence method count). Once we count an item, we remove it from the duplicated sequence.

This example shows example usage of:

- *Loop while*
- If *statement*

Because of space constraints we won't cover getMode step by step. Please compare it to the Java version of getMode.

IMPLEMENTING GETMEDIAN

The getMedian function (see Listing 8) finds the middle-most value once the sequence is sorted.

Listing 8 shows example usage of:

- *The modulus operator (%)*
- If *and* else *statements*

Comparing Java to NetRexx

The functionality of the rest of the code for this example is very similar. The NetRexx version is typically less verbose than the Java version, but unlike Python there aren't a lot of features in the NetRexx example to make this development significantly shorter or different. Compare Listing 9, the Java version of this statistics package, to Listing 10, the NetRexx version. Listings 9 an 10 can be downloaded from the ***JDJ*** Web site, www.JavaDevelopersJournal.com.

Notice that in the NetRexx version the public static methods don't have to be inside a class declaration. If static methods are defined outside a class definition, the methods are treated as though they're part of a class definition corresponding to the name of the source file. Notice also that the RunReport isn't in a static main method. If code is in a file outside a method declaration, the code is assumed to be in the main method. These features make writing scripts easier and less verbose.

Vote for Your Favorite

There are a lot of 100% pure programming languages that work in the JVM, and ***JDJ*** wants your opinion of which are the best, and why. An example list follows: JPython, NetRexx, Rhino, Instant Basic, Jacl, BeanShell, Pnuts, Bistro and Kawa (Lisp/Scheme-like).

Scorecard

How does NetRexx score? Here's my opinion.

AUTHOR'S SCORECARD FOR NETREXX

• Ease of use	9
• Embeddability	?
(Not yet, maybe via Bean Scripting Framework)	
• Resemblance to parent language	8
• Unique features	9
• String parsing	10
• Productivity	9
• Working well with Java classes	10
• Development environment/debugging	7

Cimmetry Systems, Inc.

www.cimmetry.com

Developementor

www.developmentor.com/course/ijava.htm

Let's drill down a bit on the above criteria.

- ***Ease of Use:*** Compared to Java, I feel that NetRexx is easier to learn, use and read. I have a lot of experience working and training people who are new to Java. Java is tough if you're used to working with a higher-level language. NetRexx bridges the gap, giving developers the ease of use of a higher-level language with the power of Java. I think NetRexx hits the same sweet spot as Visual Basic, for example, a good balance of dynamic typing and ease of use, as discussed in the first article
- ***Embeddability:*** N/A. I define embeddability as the ability to create and provide language for an application, for example, a scripting language (or a macro language). Since NetRexx doesn't have an interactive interpreter or an interpreted mode, I don't think this is applicable yet, although you could approximate it with the Bean Scripting Framework, which is also from IBM and will be covered in the next article. Mike Cowlishaw is adding an interactive interpreter, which should be out by the time you read this article.
- ***Resemblance to Parent Language:*** It does resemble REXX, of course, but it departs drastically from ObjectRexx and favors Java-like syntax over the less popular ObjectRexx syntax. I think the departure is a good thing; it morphs NetRexx into a language that's at home in the JVM.
- ***Unique Features:*** The so-called human-oriented features acquired from REXX give it a nice feel. REXX was designed to be easy to use, and was essentially written for nonprogrammers, that is, system administrators. This is a real advantage as the code is easier to read and understand.
- ***String Parsing:*** String can be manipulated with the methods of the REXX class or parsed with the parse instruction. The parse instruction is a powerful instruction for string manipulation. The parse command uses templates that can consist of string literals, numbers and symbols. The template is much easier to use and learn than regular expressions, yet the template is suprisingly powerful. The parse instruction comes straight from classic REXX.
- ***Productivity:*** As I've indicated throughout this article, NetRexx is less verbose than Java. Talk less, say more. In addition, it's easier to read and understand, which makes it easier to inherit someone else's code and get up to speed.
- ***Working Well with Java Classes and APIs:*** You can compile NetRexx to Java bytecode. You can easily use Java APIs from NetRexx. The syntax for importing classes is very similar to that for Java. (By the way, you can develop JavaServer Pages with NetRexx via IBM's Bean Scripting Framework. You could also use JavaScript(Rhino) and Python(JPython). Cool beans!)
- ***Development Environment/Debugging:*** I've been told by a reliable source that you can use a regular Java IDE to debug NetRexx source code, but I didn't find any instructions on the Web site describing how to do it. Since NetRexx is human-oriented, you might think that they'd provide some human-oriented instructions, but they didn't (at least none that I could find). It would be nice if IBM had a VisualAge for NetRexx or added NetRexx support to their VisualAge product. REXX and Object REXX, for example, have been good to IBM. I'd like to see IBM support this in their products.

One feature that NetRexx has that's great for debugging is the trace instruction. You can specify the trace level of a class, for example, to trace every line, method calls and more. I found this great for debugging code. Think about it. How often have you added System.out.println calls throughout your code so you could "trace" methods? This is a great feature!

Related Links

NetRexx tutorial:
http://consult.cern.ch/news/netrexx/html/nr_toc.html
Main NetRexx site:
www2.hursley.ibm.com/netrexx/

Random Thoughts

I install and try out a lot of software development tools and can usually make up for the lack of instructions if I'm given enough breadcrumbs. I found the NetRexx install a bit clumsy and quite easy to screw up. I figured it out, but I feel it could easily stump new users.

If NetRexx is geared for nonprogrammers or new Java programmers coming from a fancy 4GL with a fancy IDE, then its install is really lacking. There's no real install program – just a zip file with some instructions.

The NetRexx user base is strong and involved. While there are many NetRexx tutorials, most of them aren't linked to or from the main NetRexx site. And strangely enough, there's no real strong tutorial on the main NetRexx Web site, just a getting-started page and the language reference (which is very well written). This is a big gap and a real problem. There's also an IBM Redbook on NetRexx, but it seems pretty dated.

If you want to use the latest NetRexx features, you need to refer to the language reference. It would be nice if someone at IBM could loan Mike Cowlishaw a resource or two to come up with a good language tutorial. (A book on NetRexx written by Mike Cowlishaw probably fits the gap nicely.)

In short, you're going to have to hunt and peck at many sites to get a good feel for NetRexx programming. I did, and it wasn't

Embar
cadero
p/u

www.
embarcadero.
com/design

too difficult, but I'm lazy and would like to do one-stop shopping for information like this. Also, this may throw someone who's new to NetRexx – I assume the goal is to increase the NetRexx user base, not deter new people. If you don't make it easy, they'll go somewhere else.

I like NetRexx a lot in its current incarnation but it's not different enough from Java. I think it needs to expand on cool features like indirect properties. I want to see more. I'd like to see language support for events so that one line of NetRexx code replaces the need to declare a listener interface, event class and add/remove listener methods. The event syntax could be similar to the way Visual Basic (5.0 and above) defines events for ActiveX components.

The fact that NetRexx is a new beast and doesn't have to be backward compatible with another language is good – unlike JPython, which endeavors to be compatible with Python. This gives NetRexx a real advantage. (Python has its advantages as well – numerous advantages. Read the March ***JDJ*** [Vol. 5, issue 3] to learn more about JPython.)

NetRexx needs an interpreter but Mike is taking care of that. In addition, NetRexx also needs additional language syntax for dealing with collections, iterators and so forth, à la JPython.

Last, if IBM isn't going to dedicate more resources to NetRexx, then they should really open-source it. NetRexx has a strong user base that's involved and dedicated. It would do well in the open-source world. I'd like to see IBM open-source and add more resources.

Parting Shots

NetRexx fills the gap between high-level languages and Java. It allows you to safely declare method signatures and interfaces, which is important for systems development. I didn't find NetRexx as easy to use or as dynamic as JPython; however, comparing JPython and NetRexx is like comparing apples and oranges since one is a systems language with scripting features and the other is a dynamic scripting language. I see NetRexx closing the gap on JPython, but never getting as dynamic or as feature-rich.

However, unlike JPython, which has to be compatible with Python, NetRexx doesn't have to be compatible with ObjectRexx. This gives NetRexx a real advantage as it can add new language features that make sense in the Java Virtual Machine (e.g., indirect properties).

As you may know, we had an informal poll and JPython and NetRexx were the top two languages for the JVM. I also solicited feedback from developers who had used both JPython and NetRexx.

The feedback I got was this: if you're from a REXX background, use NetRexx. If you're just looking for a good scripting language and don't know REXX, consider using JPython instead.

NetRexx has a lot of promise, and I look forward to its evolving into an even better and more productive tool. If VisualAge supported NetRexx, I think a lot more people would use it in place of Java – including maybe me.

• • •

In the next article we'll cover the Bean Scripting Framework from IBM. The BSF promises to unify integration of scripting languages into applications. It also promises, in the future, to provide universal debugging support for most programming languages for the JVM.

AUTHOR BIO

Rick Hightower currently works at Buzzeo Corporation (www.buzzeo.com), the maker of ZEOLogix, an EJB application server, rules engine and workflow. He is a principal software engineer working on an EJB container implementation and distributed event management. In addition, he is author of a book, Programming the Java APIs with JPython, *to be published by Addison Wesley.*

rick_m_hightower@hotmail.com

QuickStream

new

www.quickstream.com

Interview...with

MIKE COWLISHAW Creator of NetRexx

R. Hightower: ***Have you considered an open-source license?***
M. Cowlishaw: Until recently the licensing issues seemed something of a minefield with so many different ideas on what open source should be. Also, my translator/compiler is very much a research scaffolding (for example, it has hooks, which look like dead code, for multiple input syntaxes). At the moment I'm (finally) implementing the interpreter pathways, which is very much a work in progress; this sort of thing is much easier to do when only one person is working on the code as a whole.

RH: ***Are there any plans to do a VisualAge for NetRexx?***
MC: You should probably ask the VisualAge people. I've shown screenshots of VisualAge running NetRexx – since it's pure Java underneath, all the debugging mechanisms work just fine. Classes generated by NetRexx should run fine under VAJava.

RH: ***How many people subscribe to your NetRexx mailing list?***
MC: To be honest, I don't know. It's run by folks in a data center in the south of England, and I don't have admin privileges on it. It must be many hundreds, though – when I put out 1.160 last week there were several hundred downloads within a day, and it was announced only on the mailing list.

RH: ***Where do you see NetRexx going in the next five years?***
MC: I see it continuing to show the value of clean syntax, etc. It currently ships with the VM/ESA operating system and I hope to see it ship with other operating systems and implementations, too.

RH: ***Is the use of NetRexx on the rise or decline?***
MC: Rising as far as I can tell. The Java platform has only really been accepted this year for "mission-critical" work now Y2K is over, and the shops that are starting to use it often have a long REXX or PL/I heritage, so they're "natural" NetRexx users.

RH: ***Is the use of REXX and Object REXX on the rise or decline?***
MC: I've lumped these together, as Object REXX is upwards compatible with "classic" REXX, so new sales (especially on PCs and workstations) are all Object REXX. On the whole, "classic" REXX usage is flat, except on the MVS (OS/390) platform, where it's increasing (in most cases, REXX is built into the operating system). ORexx is growing, especially on Windows and *X (perhaps as people move from OS/2, etc.).

RH: ***How many people are working on NetRexx?***
MC: In IBM, on the reference implementation, it's just me. There are other people working on other implementations. Lots of people outside IBM contribute ideas and code not directly concerned with the compiler – Dion Gillard, for example, is building a NetRexx IDE which should be good....

RH: ***How important does IBM view REXX? NetRexx? Object REXX?***
MC: There's a permanent development group for REXX products, based in Boeblingen, Germany. They produce and maintain all the classic REXX interpreters and compiler, and also Object REXX. REXX products are a multimillion-dollar business – there are huge amounts of code written in REXX.

RH: ***Is there a good debugger for NetRexx?***
MC: NetRexx classes are Java classes, so any Java debugger can be used with NetRexx classes. By default (unless you specify format), line numbers are correct too, due to a "cunning trick." So any debugger that correctly picks up the source file name from the .class file should work fine. I tend not to use debuggers, so I can't really advise much on this one.

Listing 1: NetRexx Employee class

```
package employee;

class Employee
 Properties Private
  firstName=String
  lastName=String
  id=int
  dept=int

 Properties indirect
  manager=Employee


 method Employee()
  firstName = "John"
  lastName = "Doe"
  id = 1
  manager=null
  dept=1


 method Employee(fname=String,
lname=String, aId=int, aManager=Employ-
ee, aDept=int)
  firstName      =        fname
  lastName       =        lname
  this.id                 =        aId
  this.manager   =        aManager
  this.dept      =        aDept


 method toString() returns String
  buf=StringBuffer  StringBuffer()
  buf.append(lastName ',')
  buf.append(firstName ',')
  buf.append("" id)
  return buf.toString()


 method main(args=String[]) static
  say Employee()
  joe = Employee Employee("Joe",
"Batista", 100, null, 1);
  ron = Employee("Ron", "Furgeson",
101, joe, 1);
  say ron
  say ron.getManager()
```

Listing 2: Java Employee class

```
public class Employee{
```

Embar
cadero
p/u

www.

embarcadero.

com/develop

```
 private String firstName, lastName;
 private int id, dept;
 private Employee manager;

 public Employee(){
  firstName = "John";
  lastName = "Doe";
  id = 1;
  manager=null;
  dept=1;
 }

 public Employee(String fname, String lname, int id, Employee
manager, int dept){
  firstName      =       fname;
  lastName       =       lname;
  this.id                =        id;
  this.manager   =       manager;
  this.dept      =       dept;
 }

 public Employee getManager(){
  return manager;
 }

 public void setManager(Employee manager){
  this.manager=manager;
 }

 public String toString(){
  StringBuffer buf = new StringBuffer();
  buf.append(lastName+',');
  buf.append(firstName+',');
  buf.append(""+id);
  return buf.toString();
 }
 …
 …
}
```

Listing 3: NetRexx EmployeeForm

```
import javax.swing.
import java.awt.GridBagLayout
import java.awt.GridBagConstraints
import java.awt.event.ActionListener
import java.awt.event.ActionEvent
import employee.Employee

class EmployeeForm extends JFrame implements ActionListener
 Properties private
  name=JTextField
  id=JTextField

 method EmployeeForm
  super("Employee Form")
  pane = JPanel()
  getContentPane().add(pane)

  pane.setLayout(GridBagLayout())


   -- Create a name, and id text field.
  name = JTextField(25)
  id = JTextField(10)

   -- Create and add a "Name" and "ID" label.
  nameLabel = JLabel("Name")
  nameLabel.setLabelFor(name)
  nameLabel.setDisplayedMnemonic('N')
  constraint = GridBagConstraints()
  pane.add(nameLabel, constraint)

  idLabel = JLabel("ID")
  idLabel.setLabelFor(id)
  idLabel.setDisplayedMnemonic('I')
  constraint.gridy=1
  pane.add(idLabel, constraint)

   -- Add the name and ID text field to the form.
  constraint.gridy=0; constraint.gridx=1
  constraint.weightx=80.00
  constraint.anchor=GridBagConstraints.WEST
  pane.add(name, constraint)
  constraint.gridy=1
  pane.add(id, constraint)

   -- Create an okay button, add it, and set up its event  handler.
  okay = JButton("Okay")
  okay.setMnemonic('O')
  constraint.gridx=1; constraint.gridy=2
  constraint.anchor=GridBagConstraints.EAST
  pane.add(okay, constraint)
  okay.addActionListener(this)

  this.setVisible(1)
  this.pack()

 method actionPerformed(event=ActionEvent)
  handleOkay()

 method handleOkay

  fname = String
  lname = String
  _id = 0

  _name = Rexx(this.name.getText())
  fname = _name.word(1)
  lname = _name.word(2)

  _id = Integer.parseInt(this.id.getText());

  theEmployee = Employee(fname, lname, _id, null, 100);
  say theEmployee
  say theEmployee.getClass().getName()

 method main(args=String[]) static
  EmployeeForm();
```

Listing 4: Java EmployeeForm

```
import javax.swing.*;
import java.awt.GridBagLayout;
import java.awt.GridBagConstraints;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import employee.Employee;

public class EmployeeForm extends JFrame{
 private JTextField name;
 private JTextField id;

 public EmployeeForm(){
  super("Employee Form");
  JPanel pane = new JPanel();
  getContentPane().add(pane);

  pane.setLayout(new GridBagLayout());


   // Create a name, and id text field.
  name = new JTextField(25);
  id = new JTextField(10);

   // Create and add a "Name" and "ID" label.
  JLabel nameLabel = new JLabel("Name");
  nameLabel.setLabelFor(name);
  nameLabel.setDisplayedMnemonic('N');
  GridBagConstraints constraint = new GridBagConstraints();
  pane.add(nameLabel, constraint);

  JLabel idLabel = new JLabel("ID");
  idLabel.setLabelFor(id);
  idLabel.setDisplayedMnemonic('I');
  constraint.gridy=1;
  pane.add(idLabel, constraint);

   // Add the name and ID text field to the form.
  constraint.gridy=0; constraint.gridx=1;
  constraint.weightx=80.00;
  constraint.anchor=GridBagConstraints.WEST;
  pane.add(name, constraint);
  constraint.gridy=1;
  pane.add(id, constraint);
```

4th Pass

www.4thpass.com

```
   // Create an okay button, add it,and set up its event handler.
  JButton okay = new JButton("Okay");
  okay.setMnemonic('O');
  constraint.gridx=1; constraint.gridy=2;
  constraint.anchor=GridBagConstraints.EAST;
  pane.add(okay, constraint);
  okay.addActionListener(new ActionListener(){
   public void actionPerformed(ActionEvent event){
    handleOkay();
   }
  });

  this.setVisible(true);
  this.pack();
 }

 public void handleOkay(){

  String name, fname, lname;
  int index=0;
  int id =0;

  name = this.name.getText();
  index = name.indexOf(" ");
  fname = name.substring(0, index);
  lname = name.substring(index+1,
          name.length());

  id = Integer.parseInt(this.id.get
                         Text());

  Employee employee = new Employee(fname, lname, id, null, 100);
  System.out.println(""+employee);
 }

 public static void main(String [] args){
  new EmployeeForm();
 }
}
```

Listing 5: getRange

```
method getRange(nums=ArrayList) public static returns  ArrayList
 -- Find the range. Returns a tuple with the minimum,
 -- maximum, and range value
 min=double
 max=double
 ranges=ArrayList

 min = ( Double Collections.min(nums)).doubleValue()
 max = ( Double Collections.max(nums)).doubleValue()


 ranges = ArrayList()
 ranges.add(Double(min))
 ranges.add(Double(max))
 ranges.add(Double(max-min))

 return ranges
```

Listing 6: getMean

```
method getMean(nums=ArrayList, sample=boolean 1) public stat-
ic returns double
  -- Define mean that finds two types of mean, namely:
  -- population mean and sample mean
 value=Double
 sum=0.0
 average=0.0
 iterator = nums.iterator()
 loop while iterator.hasNext()
  value = Double iterator.next()
  sum = sum + value.doubleValue()
 end

  -- Check to see if this is a sample mean
 if sample then
  average = sum / nums.size()-1
 else
  average = sum / nums.size()
 return average
```

Listing 7: getMode

```
method getMode(nums=ArrayList) public static returns double
 -- Find the number that repeats the most.

  -- make a duplicate copy of the nums argument
 duplicate = ArrayList(nums)

 Collections.sort(duplicate)
 highest_count =double -100
 mode =double -100


 iterator = nums.iterator()

  -- iterate through nums removing each item out of the duplicate
  -- calculate the highest_count and the mode
 loop while iterator.hasNext()

  count =double 0
  item = iterator.next()

   -- Count the number of times the item occurs in the list
   -- If Count is 0 go to the next iteration
  count = countMode(item, duplicate)
  if count == 0 then iterate

   -- determine the highest count. The highest counted item
   -- is the mode.
  if count > highest_count then
  do
   highest_count = count
   mode = (Double item).doubleValue()
  end
 end

method countMode(object=Object, list=ArrayList) private stat-
ic returns int
 index=int 0
 count=int 0
 loop until index >=0
  index = Collections.binarySearch(list, object)
  if index >=0 then list.remove(index)

  count = count + 1
 end

 return count
```

Listing 8: getMedian

```
method getMedian(nums=ArrayList) public static returns double
 -- Find the Median number

  -- create a duplicate since we are going to modify the sequence
 seq = ArrayList(nums)

  -- sort the list of numbers
 Collections.sort(seq)

 median = 0.0 -- to hold the median value

 length = seq.size() -- to hold the length of the sequence
 index=0

  -- Check to see if the length is an even number
 if ( ( length % 2) == 0) then
 do
   -- since it is an even number
   -- add the two middle number together
  index = length / 2
  m1 = (Double seq.get(index-1)).doubleValue()
  m2 = (Double seq.get(index)).doubleValue()
  median = (m1 + m2) /2.0
 end
 else
 do
   -- since it is an odd number
   -- just grab the middle number
  index = (length / 2)
  median = (Double seq.get(index)).doubleValue()
 end

 return median
```

Sic Corporation

www.sic21com

RT-SET Acquires Evans & Sutherland's Virtual Studio Technology and Products

(*New York, NY, and Salt Lake City, UT*) – RT-SET (Real Time Synthesized Entertainment Technology) Ltd and Evans & Sutherland Computer Corporation have finalized the sale of the E&S Digital Video Division and the entire MindSet Virtual Studio Systems' product line (including the in-process E&S PC-based platform technology development) to RT-SET.

The transaction will provide RT-SET, a leading provider of fully integrated broadcast graphics solutions, with a complete 3D PC-based, open-platform virtual studio solution that complements its existing line of solutions.

www.rtset.com

Open Market and ILOG Collaborate

(*Burlington, MA, and Mountain View, CA*) – Open Market, Inc., and ILOG have announced that the ILOG JRules Java rule engine was selected to provide enhanced personalization and marketing support features for Open Market's new Marketing Studio. Introduced as a component of Open Market's new e-business suite, Marketing Studio is a data analysis and marketing management product.

www.openmarket.com / www.ilog.com

Netmosphere Announces Enact Enterprise System 4.0

(*Palo Alto, CA*) – Netmosphere has unveiled Enact Enterprise System 4.0, an Internet-based product targeted at real-time, enterprise-wide collaboration. The system incorporates Netmosphere's flagship products, ActionPlan and Project Home Page, into a single, unified solution that provides everyone in the company, including executive management, team members and project planners, with the tools they need to effectively manage their complex projects in real time.

www.netmosphere.com

Sun Microsystems, NARUS in Alliance

(*Palo Alto, CA*) – Sun Microsystems, Inc., and NARUS, Inc., the pioneer and leading provider of Internet Business Infrastructure (IBI) solutions, have announced a wide-ranging agreement to bring NARUS IBI solutions, powered by Sun computing, and software infrastructure solutions to IP service providers worldwide. Analysts estimate the IBI solutions market to be as much as $8 billion by the year 2004.

www.sun.com / www.narus.com

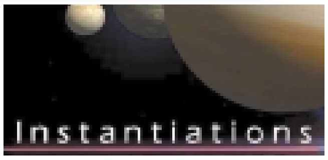
GE Fanuc Launches New EnterpriseRT Software

(*Charlottesville, VA*) – GE Fanuc Automation has launched new EnterpriseRT software, which combines and visualizes real-time data from multiple company systems including ERP, supply chains, customers, product management, manufacturing and control. Based on a technology foundation developed by IndX Software Corporation, EnterpriseRT uses Web technologies to provide a single portal for real-time monitoring and reporting of various systems and processes throughout an enterprise and its global universe. This improved combination of information enables users to make informed decisions about processes as they occur.

www.gefanuc.com

Instantiations Announces Products for VisualAge Development Environments

(*Portland, OR*) – Instantiations, Inc., has announced VA Assist Enterprise, a product family that will complement IBM's VisualAge brand of development tools.



The VA Assist Enterprise family includes products that complement IBM's VisualAge Generator (both VisualAge for Java and VisualAge Smalltalk versions) and VisualAge Smalltalk Enterprise. Developers may download a free 30-day evaluation copy of the available products at www.instantiations.com.

Flashline.com Unveils '11 Steps to Component-Based Development'

(*Cleveland, OH*) – A new resource section of Flashline.com, "11 Steps to Component-Based Development," guides IT professionals through the strategy, design and implementation of business systems developed with a component-based (CBD) architecture. The steps help developers, IT managers and business decision-makers realize the bottom-line benefits of CBD, including faster time-to-market and simplified development.

www.flashline.com

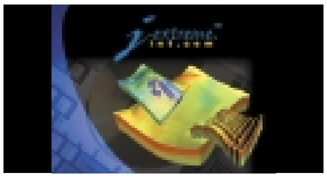
Advanced Software Technologies Names Marketing Director

(*Littleton, CO*) – Advanced Software Technologies Inc. has expanded its management team with the addition of Joshua Horwitz as director of marketing. Horwitz comes to Advanced Software with a broad range of marketing and management experience that includes a previous position as product marketing manager for Lotus.

www.advancedsw.com

INT Launches 100% Pure Java Graphics Tool

(*Houston, TX*) – Interactive Network Technologies, Inc., introduces an enhanced version of J/Carnac, its free graphics toolkit based on Java 2D.



J/Carnac offers a powerful API to develop state-of-the-art graphical applications such as scientific graphs and diagrams, maps, real-time charts and more. The bytecode development license for J/Carnac is also available for free, and there is no run-time fee for embedding J/Carnac inside applications or applets.

J/Carnac 1.0 boasts a variety of new features including improved performance and handling, new modes for shape editing and improved text rendering.

j-extreme.int.com

IONA and Progress Software Announce Agreement

(*Boston, MA*) – IONA Technologies and Progress Software Corporation have announced the signing of a multimillion dollar agreement in which the companies will engage in joint development efforts and exchange best-of-breed Java 2 Enterprise Edition technologies.

Under the license agreements IONA will integrate Progress SonicMQ Internet messaging server into the iPortal Application Server, and Progress Software will license IONA's iPortal Application Server technology and incorporate it into Progress Software's next-generation application server.

www.IONA.com / www.progress.com

Metamata, Inc.

www.metamata.com

Northwoods Releases JGo V2

(*Nashua, NH*) – Northwoods Software has announced a major new release of its JGo diagram class library. JGo is a critical component of applications such as network or hierarchy diagrams, flowcharts and CASE or workflow tools.

JGo V2 supports Java 2, Swing, Java 2D, zooming, panning, selection, drag and drop, cut/paste and printing. Creating custom nodes and connecting links is enabled by subclassing of classes provided in JGo.

www.nwoods.com

Netegrity Enhances Security in BEA E-Commerce Transaction Platform

(*Waltham, MA*) – Netegrity, Inc., and BEA Systems, Inc., have announced an alliance that will provide e-businesses with a secure, manageable and scalable solution for J2EE. Netegrity's SiteMinder 4.0 will support BEA WebLogic Server 5.1 and BEA WebLogic Enterprise 5.1, which will provide BEA customers with a comprehensive platform for securely managing J2EE applications and components. Netegrity and BEA have also entered into a longer-term agreement to extend SiteMinder support across the BEA product line, including added support for BEA Tuxedo.

www.bea.com / www.netegrity.com

UniBar Launches e-BARCODE 2000

(*Rochester Hills, MI*) – UniBar Inc. introduces an e-commerce version of its label printing software, e-BARCODE 2000, a server-based label-printing software running on UNIX, Linux and Windows NT systems. e-BARCODE 2000 enables customers to view and print bar codes created from a Web application via an Internet browser.

A demo can be viewed by accessing the Unibar Web site at www.unibar.com.

Everypath, Ericsson in Strategic Relationship

(*Los Angeles, CA*) – Everypath Inc. has announced an agreement with Ericsson to share technology to develop secure mobile e-commerce applications.

Everypath announced the availability of a service that can rapidly translate existing Web site content that can be accessed on all WAP (wireless application protocol) cellular telephones. As the Everypath solution doesn't store "persistent" data on Everypath's server, the privacy and security of users' credit card information and data is ensured.

www.everypath.com

Macromedia and ATG Form Strategic Partnership

(*Los Angeles, CA*) – Macromedia, Inc., and Art Technology Group, Inc. (ATG), have announced a strategic partnership to integrate Macromedia's comprehensive Web Content Lifecycle products for content creation, analysis and real-time recommendations with the ATG Dynamo Product Suite for personalized e-commerce Web applications. The companies will enable Global 2000 and leading dot-com companies to shorten time-to-market for their strategic online business initiatives and maximize the effectiveness of their online customer relationships. This strategic partnership will involve collaborative development efforts as well as joint marketing and field-level sales activities.

www.macromedia.com

JavaScript 1.5 Available in New Netscape 6 Browser

(*Mountain View, CA*) – Netscape Communications, a subsidiary of America Online, Inc., has announced the availability of JavaScript 1.5, the latest version of its industry-leading scripting language. Netscape 6 Preview Release 1, now available for free public download, features full support for JavaScript 1.5, allowing Web developers to create powerful new Web applications using this popular scripting language.

http://home.netscape.com

HP, Everypath to Co-Market Mobile E-Services

(*Palo Alto, CA, and Los Angeles, CA*) – Hewlett-Packard and Everypath, Inc., a Santa Clara, California-based mobile application services provider, have agreed to co-develop and co-market a new wireless Internet service that will make it easy for companies to deliver personalized e-services to mobile-device users.

The new service will allow users of mobile devices, such as smart phones and Palm Inc.'s Palm VII organizer, to connect at any time to their most important Web services – including e-banking, stock trading and online auctions – and complete e-commerce transactions directly from the device.

www.everypath.com / www.hp.com

SPSS Ships Customer Relationship Management Software

(*Chicago, IL*) – SPSS Inc. has come out with SmartScore, new software that applies data mining technology to help companies customize interactions at all customer touchpoints in real time.

These models, derived from historical and sampled data, score new cases or customers in real time based on available information. People and systems use the scores to decide which products and services to present. Additional examples of how SmartScore can be used include online promotional offerings, fraud detection and risk prevention.

www.spss.com

SIlverstream Acquires eObject

(*Burlington, MA*) – SilverStream Software, Inc., has signed a definitive agreement to acquire eObject, a developer of advanced rules-based personalization technologies. SilverStream will integrate eObject's personalization technology into the company's upcoming Portal solution – a key part of SilverStream's eBusiness Platform.

www.silverstream.com

HiT Ships Direct OLE DB Middleware for IBM DB2

(*San Jose, CA*) – HiT Software, Inc., announces the availability of HiT OLEDB Server/DB2. HiT OLEDB, an SSL-enabled Windows OLE DB middleware product that provides direct access to IBM DB2 databases including DB2 UDB for OS/390.

HiT OLEDB technology supports ADO v2.1 and later application environments such as Microsoft IIS/ASP, SQL Server 7 and the Windows 2000 OS family.

HiT OLEDB/DB2 is available immediately in both client and server versions. A preview of the Developer Edition ASP Toolkit is available at www.hit.com/hit-web/daccess/dchome.htm.

Neoware Systems Introduces NeoLinux

(*King of Prussia, PA*) – Neoware Systems, Inc., has introduced the NeoLinux operating system – the first embedded Linux distribution designed specifically for business-to-business information appliances.

Based on official Red Hat Linux, NeoLinux includes important features designed specifically for information appliances. Combined with Linux software from other companies, NeoLinux is designed to power a new generation of dedicated B2B information appliances such as cash registers, firewalls, routers, interactive Web kiosks, thin clients, security devices and wireless appliances.

www.neoware.com

Pramati

www.pramati.com/j2ee.htm

JDJ NEWS

Netmosphere Announces Enact Enterprise System 4.0

(*Palo Alto, CA*) – Netmosphere has unveiled Enact Enterprise System 4.0, an Internet-based product targeted at real-time, enterprise-wide collaboration.

The system incorporates Netmosphere's flagship products, ActionPlan and Project Home Page, into a single, unified solution that provides everyone in the company, including executive management, team members and project planners, with the tools they need to effectively manage their complex projects in real time.

www.netmosphere.com

Q-Research Tools Let Users Enhance and Edit Digital Images Online

(*North Bend, WA*) – Q-Research Inc. has launched a suite of tools that allows users to enhance and manipulate images online in real time. VisualGenetics may be licensed individually or as a package to Internet content providers. VisualGenetics also offers image-editing tools such as cropping, rotation, flipping, painting, warping, red-eye reduction and various forms of image layering and blending.

www.q-res.com

ClickAction Releases ERM 5

(*Palo Alto, CA*) – ClickAction Inc., a leader in permission-based e-mail marketing, has released ClickAction Email Relationship Management (ERM) 5, the company's next-generation e-mail marketing system. ClickAction ERM 5 provides enhanced database integration, targeting and personalization capabilities.

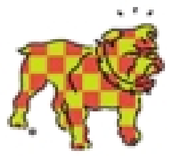
Features include data exchange with external sources, self-serve campaign management, in-bound e-mail processing, real-time tracking and reporting, and permission profiling.

www.ClickAction.com

Bulldog Announces Bulldog Two.Six

(*Toronto, ON*) – The Bulldog Group Inc. has announced the next version of its content management software, Bulldog Two.Six.

Bulldog Two.Six provides the cost-saving benefits associated with the reuse and repurpose of media content, and creates an infrastructure for new media distribution opportunities including e-commerce and interactive services.



Features include management of all media types including video, audio, text and images; sophisticated registration, searching and retrieval tools; and media version control and metadata management.

www.bulldog.com

Microsoft Claims Small Victory in Sun Legal Battle

(*Seattle, WA*) – According to Microsoft, it has scored a small victory in its legal battle with archrival Sun Microsystems Inc. over its license for a computer programming technology developed by Sun.

In a recent ruling Judge Ronald Whyte of the U.S. District Court in San Jose, California, rejected Sun's interpretation of its contract with Microsoft concerning updated versions of its Java technology.

Microsoft spokesman Jim Cullinan said that the pretrial ruling confirmed a tentative decision on the issue made by Whyte last June. Sun was not immediately available for comment.

www.microsoft.com / www.sun.com

Epic Edge

www.epiedge.com

Geek Cruises

www.geekcruises.com

North-
woods

www.nwoods.com

Visualize

www.visualizeinc.com

SHOP ONLINE AT **JDJSTORE.COM** FOR BEST PRICES OR CALL YOUR ORDER IN AT **1-888-303-JAVA**

Take a look at our specials this month!

JDJStore.COM

BUY THOUSANDS OF PRODUCTS AT GUARANTEED LOWEST PRICES!

Guaranteed Best Prices

JDJ Store Guarantees the Best Prices.

If you see any of our products listed anywhere at a lower price, we'll match that price and still bring you the same quality service.

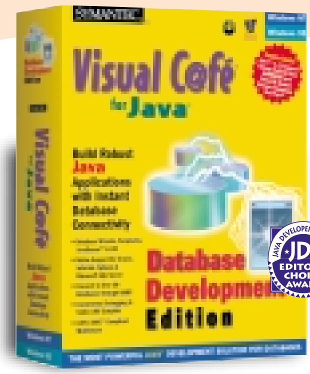
Terms of offer:

- Only applicable to pricing on current versions of software
- August issue prices only
- Offer does not apply towards errors in competitors' printed prices
- Subject to same terms and conditions

Prices subject to change.

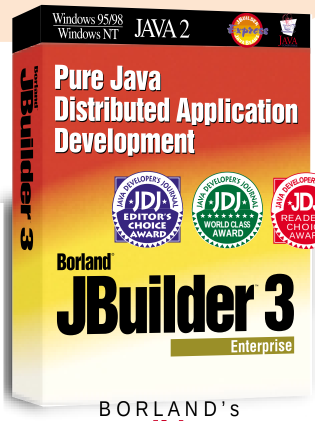
Not responsible for typographical errors.

Attention Java Vendors:
To include your product in JDJStore.com, please contact jackie@sys-con.com



IBM's
VisualAge for Java 2.0 Enterprise Edition

JDJStore.com Price . .	$2,498.99
COMPARE...	
Programmer's Paradise Price	$2,499.00
Beyond.com Price	$2,525.00



SYMANTEC's
Visual Café Professional Edition

JDJStore.com Price . .	$258.99
COMPARE...	
Programmer's Paradise Price	$259.95
Beyond.com Price	$299.75

BORLAND's
JBuilder 3 Professional New User

JDJStore.com Price . .	$499.99
COMPARE...	
Programmer's Paradise Price	$545.99
Beyond.com Price	$548.99

ORDER TODAY! 888-303-JAVA

XML DevCon 2000
Spread

www.xmldevcon2000.com

XML DevCon 2000
Spread

www.xmldevcon2000.com

Servlet to Servlet Communication

A look at the options you have

WRITTEN BY ALAN WILLIAMSON

Java has brought much in the way of programming advancement within easy reach of developers. Powerful constructs such as multithreading and advance communication are now relatively easy for you to use in your programs. Java has also brought a strong sense of object orientation. The ability to reuse code and not reinvent the wheel each time you program is a great improvement over alternative programming languages.

But for a developer to reuse code, the language has to support the mechanisms that make it easy to build complex systems by simply plugging objects together. Java has these mechanisms – the servlet API is no stranger to object reuse. This article outlines the different methods by which servlets can communicate data to one another. This communication promotes the concept of solving a problem only once and then applying the solution to many different areas.

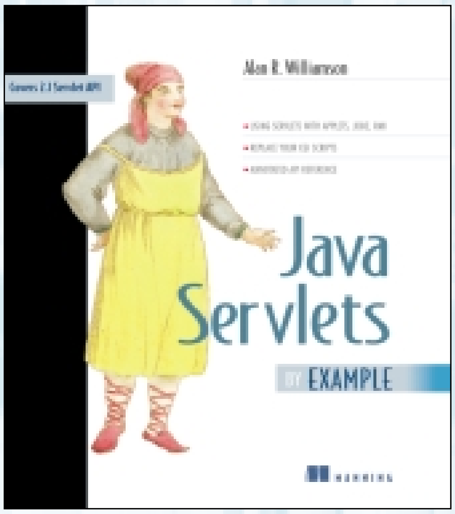
As this article will illustrate, you, as a servlet developer, have a lot of choices on how you can best reuse your servlet classes.

Common Base Class

A servlet is run in response to a client connection. It is designed to be lightweight and to process the client request as quickly as possible, thus freeing up the server so it can immediately process the next request that comes in. A servlet should never try to do everything itself. In a large system it is better to produce many smaller servlets, as opposed to a small number of large servlets. The longer a servlet spends servicing a request, the fewer clients the servlet can process in a given time.

If smaller servlets are to be built, rather than using the one-servlet-solution-fits-all approach, the ability to share information between the servlets is paramount.

Last in a series of articles adapted from *Java Servlets: By Example* by Alan R. Williamson, reproduced here by permission of Manning Publications.



One of the easiest ways for a servlet to share information (this works for classes as well) is to share common objects through inheritance – where the servlets are extended from a class other than HttpServlet or GenericServlet. To illustrate this concept, consider the example shown in Figure 1.

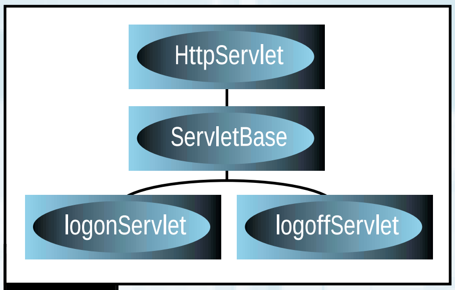


FIGURE 1 Inheritance

Suppose we had two servlets in our system: one to log a user onto our system and the other to log him. Let's assume that the reason we want a user to log on and log off again is to enable us to connect to a specific database given the username and password. Now, let us assume that they will share various methods, such as a simple check to see if the user is already logged on. We can develop methods for this simple check and place a copy of them in each servlet. However, since it's a database connection, let's also assume that we will be sharing database connections between the servlets.

Many different methods are available to facilitate method sharing and data pooling. For this section, we will use the simplest method. Instead of each servlet extending the HttpServlet, each will extend the capabilities of ServletBase, which is a custom class we will build. This class will itself extend the HttpServlet class, but it will also add other methods to allow the servlets to intelligently handle the database connections.

Listing 1 shows the basic skeleton implementation for the ServletBase class. You'll notice that it looks very similar to a normal servlet, except it doesn't override the service(...) method or any of the doXXX(...) methods. It does, however, override the init(...) method. In this instance it needs to open up a connection to the database which will be shared through the static Connection variable. This will be performed when the servlet first initializes.

However, please note a very important fact. The init(...) method will be called multiple times if multiple servlets inherit this class because each particular servlet needs to be initialized and needs to go through the complete servlet life cycle. Since we don't want to have additional database connections open, we'll check to see if it has been opened or not, and if so, we skip over that section.

The two additional methods, isLoggedOn(...) and logUserOn(...), are support functions that are accessible by all servlets that extend this class.

So now that we have developed the new base class, let's look at how we can use it. Listing 2 shows the logonServlet that will log a user onto the system. This class, instead of extending the HttpServlet class, extends the ServletBase class. Notice that this class does not override the init(...) method because the base class is taking care of this for us.

The API doesn't require the init(...) method to be overridden like the previous versions insisted it should be, so it may be left out without any adverse effects.

The logonServlet can now call methods from the underlying base class without having to wrestle with any object references. This allows class to easily share common methods and data without keeping a reference to a collection class.

This ability is not hard, and it's not new. It is used throughout Java to create groups of logical units, and it can easily be used in the world of servlets, where the need to share data is even more important.

Sharing Data

If the previous method seems a little clunky and too much of a hassle, the 2.1 version of the servlet API introduces a way to share objects among servlets. It operates along the same lines as the session management functionality, but it is done completely on the server side.

In the session management, you associated an object with a given client session. The session manager attempted to return the same object to you each time that particular client made a request. This was done using cookies or URL rewriting. Objects were stored using a unique string to reference them.

Servlets can now benefit from this technology through the ServletContext interface. Each servlet runs within some sort of environment. This environment describes various parameters associated with the servlet environment, such as

KL Group Inc.

www.klgroup.com/course

the document root and any mappings that may exist. This is known as the ServletContext. A servlet belongs to only one ServletContext and the server's Web administrator controls this.

For example, a server running virtual hosts would have a different ServletContext for each virtual host. This has the advantage of logically grouping servlets while maintaining different security models for each type.

The ServletContext can store objects on behalf of servlets running within its context. There are methods that allow a servlet to add and remove objects and to determine what the objects are that are stored. These objects can be retrieved and modified by other servlets running within the context.

Let's now look at how we could share a database connection among servlets using the ServletContext mechanisms. Listing 3 shows a simple example where we create a new database connection from some method and ask the ServletContext to store it for us under the name of "database.connection" using the setAttribute(...) method.

Retrieving the database connection again is just as easy as setting it. Listing 4 shows another servlet, running within the same context, which attempts to retrieve the object back.

If the object has not been stored, the getAttribute(...) method will return null. As you can see, the whole storage and retrieval system operates very much like the cookie system – so this can be thought of as server-side cookies!

Listing 5 illustrates another useful method from the ServletContext interface. The ability to discover which objects are in existence is a very handy feature; for example, it enables you to discover whether any initialization has been performed. The getAttributeNames() method returns an Enumeration to the list of stored objects, which can then be returned using the getAttribute() method.

We have now seen methods for storing and retrieval of data. One additional method allows you to remove a given object from the ServletContext. This method, removeAttribute(...), ensures that the ServletContext no longer holds a reference to the object.

So the ability to share data among servlets running within the same context is straightforward enough. But what if the servlet isn't running within the same context? What if a servlet running on another machine or another virtual host has data that the current servlet needs? Have no fear; help is at hand.

The 2.1 servlet API introduces a new interface that allows servlets to get an external reference to a context. This is done through an overloaded version of getServletContext(...) which takes in the URL of the servlet you wish to retrieve the context for. An example of this call is shown in Listing 6.

Assuming the remote host allows this to be performed, the ServletContext will be returned. However, if it is not, then the method will return null. In calling this method, it will not invoke the servlet running in the other context, nor will it interpret any processing that that remote servlet may be doing at that point.

Using this new mechanism from the servlet API, you can easily share objects with other developers, including other contexts.

Servlets Working Together

In the previous section, we looked at a number of ways to share objects between servlets. In most instances this is used to share common data resources such as a database connection. This can allow for more efficient operation within the server. Now, thanks to the new addition to the servlet API, servlets can work together in yet another way.

In past version releases, a technique known as servlet chaining was possible. In servlet chaining, the output of one servlet fed the input to another servlet. The second servlet, or the next one in the chain, could perform additional processing before sending the result out to the client. Although this was useful, it was a bit clunky to operate. To begin with, the chaining mechanism had to be set up by the Web administrator, and it could not be performed dynamically. Also, the servlet had no control over whether it should be in a chain. From the developer's point of view, no API control was possible.

The new API has made this feature much more accessible. You have the ability to insert the contents of one servlet within the output of another, and you can pass a request on to another servlet for processing.

The following sections will show you how you can use these features in your servlets.

Forwarding a Request

You may find a time when it's better to pass a request on to another servlet for processing. For example, consider a servlet that acts as a front-end processor for a search engine. Depending on the result of the query, different servlets could be used to generate the actual output. The first servlet would take in all the search parameters, run the query, and pass the results on to a specific servlet for output. The advantage over the conventional servlet-chaining model is that the servlet can decide at runtime which output servlet will be used as opposed to relying on a Web administrator to set up the mapping. This is particularly useful in an ISP environment where the developer has to rely on a third party to get this right.

To handle this scenario and the one in the next section, a new interface was developed. The RequestDispatcher interface is the glue that controls the flow between servlets and output streams such as HTTP clients, files, or even JavaServer Pages (JSP).

Before you can send control to another servlet, you must first get the necessary RequestDispatcher for that particular servlet. This is controlled through the method call getRequestDispatcher(...) from the ServletContext interface. You specify the URI to the servlet, and assuming no security violations incur, the necessary RequestDispatcher reference is returned.

From here, you can make a call to the forward(...) method, which passes on the request to the servlet represented by theRequestDispatcher. Let's look at an example of this in action.

Listing 7 shows a typical use of this mechanism. Assume that the forwardServlet shown takes in a search query from the user. It first retrieves an instance to the context it is running within, then using this reference a call to retrieve the RequestDispatcher for the xyzServlet that will be used to generate our output for us.

Before we pass on the request for processing, we run the query. The next servlet will be used to process the results for this query, so we need some mechanism to pass the results to the next servlet.

As demonstrated in the previous section, sharing objects between servlets is not that big a problem; we simply ask the ServletContext to store the data for us. But this is not satisfactory, as these objects take no account of the client request. Storing the results this way would be very dangerous since only one copy per ServletContext would exist, as opposed to one copy per client request.

You could use session management to get around this problem, but since the data hasn't actually made it back to the client, it's not an ideal solution.

Thanks to the ServletRequest object, help is available. There are methods that allow servlets to ask that the ServletRequest object store objects for it on its behalf, ensuring that the object remains intact per client session. The first servlet can then pass data to the second servlet without worrying about multiple-client sessions.

As you'll see in Listing 7, the method for storing data is exactly the same as was demonstrated for storing objects using the ServletContext. The equivalent getAttribute(...) methods exist to allow the servlet to retrieve the objects again.

When you are forwarding a request to another servlet for processing, the first servlet is not permitted to do anything to the output, including setting any HTTP status fields. As soon as the first servlet makes an attempt to retrieve an output stream, then the call to forward(...) will fail. The calling servlet is permitted to continue running after the call, but it must not make any attempt to communicate back to the original client.

Inserting a Request

The previous section looked at when the complete request is passed on to another servlet for processing. Although this procedure is very useful, it is restrictive – only one servlet can produce the final output. There are times where it would be nice if servlets could cooperatively work together to create the output. For example, you could have one servlet conduct or control the flow calling other servlets to insert content as and when.

Iona

www.iona.com

Fortunately the API supports this very type of use. The previous section introduced the forward(...) method from the RequestDispatcher interface. To insert content into the current output stream, the insert(...) method is available. This method operates in exactly the same way as forward(...) in that it accepts a ServletRequest and ServletResponse as parameters. Only this time, the called servlet is not meant to set any headers. If it does (for example, any cookie setting), then it is not guaranteed to be successful. As before, data can be passed to the servlet using the ServletRequest interface that was shown in the previous section.

Listing 8 shows an example of the insert(...) method. As before, it retrieves a reference to the RequestDispatcher of the servlet we wish to use. We create some output from the original servlet and then call the secondary servlet 10 times so it can insert its output to the original client request.

There is no limit to the number of times or number of servlets you can ask to insert output for. Using this technique, you have the ability to create sophisticated servlet systems that can all work together closely, thus reducing the total number of servlets that the server must develop and handle.

Summary

This article introduced the features that are available to you for sharing data between servlets. This functionality, as well as the ability to have servlets share the responsibility of processing the client request by collaborating on the output generation, was a major step forward in servlet ideology.

alan@sys-con.com

Listing 1: ServletBase

```
public class ServletBase extends HttpServlet{
  static Connection databaseConnection = null;

  public void init(ServletConfig _config) throws ServletException{
    super.init(_config);
    if ( databaseConnection == null )
      //- Open up the database connection
  }

  protected boolean isLoggedOn( String _username ){
    return true;
  }

  protected boolean logUserOn( String _username ){
    return true;
  }
}
```

Listing 2: Using the New ServletBase Class

```
public class logonServlet extends ServletBase{
  public void service(HttpServletRequest _req, HttpServletRe-
                      sponse _res) throws ServletException{
    if ( isLoggedOn( _req.getParameter("USERNAME") ){
      //- Display a message indicating they are already logged on
    }else{
      logUserOn( _req.getParameter("USERNAME") );
    }
  }
}
```

Listing 3: Storing an Object

```
public class logonServlet extends HttpServlet{
  public void service(HttpServletRequest _req, HttpServletRe-
                      sponse _res) throws ServletException{
    ServletContext thisContext = getServletContext();

    //-- Assume some method creates a new connection class
    Connection newConnection = createConnection();

    thisContext.setAttribute( "database.connection", newConnection );

    //-- Return some output to the client
  }
}
```

Listing 4: retrieving an Object

```
public class logoffServlet extends HttpServlet{
  public void service(HttpServletRequest _req, HttpServletRe-
                      sponse _res) throws ServletException{
    ServletContext thisContext = getServletContext();

    //-- Assume some method creates a new connection class
    Connection newConnection = thisContext.getAttribute(
                          "database.connection");
    if ( newConnection == null )
      //- Database has not been opened yet


    //-- Return some output to the client
  }
}
```

Listing 5: Looking at All the Objects

```
public class allServlet extends HttpServlet{
  public void service(HttpServletRequest _req, HttpServletRe-
                      sponse _res) throws ServletException{
    ServletContext thisContext = getServletContext();

    //-- Assume some method creates a new Connection  class
    Enumeration E = thisContext.getAttributeNames();
    while ( E.hasMoreElements() ){
      String name = (String)E.nextElement();
      System.out.println( "Object: " + name );
    }
  }
}
```

Listing 6: Retrieving Remote Contexts

```
public class otherServlet extends HttpServlet{
  public void service(HttpServletRequest _req, HttpServletRe-
                      sponse _res) throws ServletException{
    ServletContext otherContext =
getServletContext("http://<otherdomain>/servlet/allServlet");

    //-- Assume some method creates a new Connection class
    Enumeration E = otherContext.getAttributeNames();
    while ( E.hasMoreElements() ){
      String name = (String)E.nextElement();
      System.out.println( "Object: " + name );
    }
  }
}
```

Listing 7: Forwarding a Request

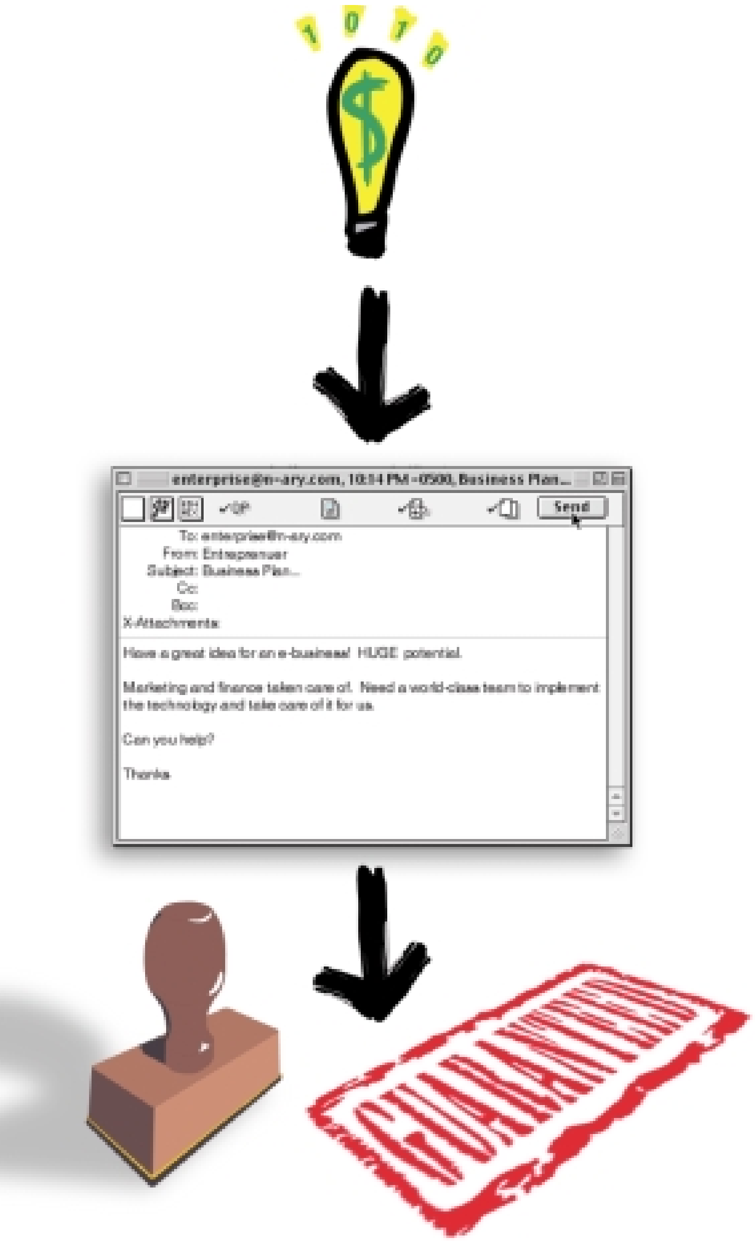
```
public class forwardServlet extends HttpServlet{
  public void service(HttpServletRequest _req, HttpServletRe-
                      sponse _res) throws ServletException{
    ServletContext xt = getServletContext();
    RequestDispatcher xyzServlet =
xt.getRequestDispatcher("http://<domain>/servlet/xyzServlet");

    //- Do any preliminary processing
    _req.setAttribute( "database.results", new Results() );

    xyzServlet.forward( _req, _res );
  }
}
```

Listing 8: Inserting Content

```
public class insertServlet extends HttpServlet{
  public void service(HttpServletRequest _req, HttpServletRe-
                      sponse _res) throws ServletException{
    ServletContext xt = getServletContext();
    RequestDispatcher xyzServlet =
xt.getRequestDispatcher("http://<domain>/servlet/xyzServlet");

    PrintWriter Out = _res.getWriter();
    Out.println( "This is from the insertServlet " );
    for(int x=0; x < 10; x++ )
      xyzServlet.insert( _req, _res );

    Out.println( "This is the end of the print servlet " );
  }
}
```

Concentric Network

www.concentrichost.net



enterprise@n-ary.com, 10:14 PM -0500, Business Plan...

To: enterprise@n-ary.com
From: Entrepenuer
Subject: Business Plan...
Cc:
Bcc:
X-Attachments:

Have a great idea for an e-business! HUGE potential.

Marketing and finance taken care of. Need a world-class team to implement the technology and take care of it for us.

Can you help?

Thanks

World class development guaranteed. We have a proven track record in taking peoples ideas and turning them into million dollar winners. For a small stake in your company, we will become your complete development department. With world class Java skills from reknowned figures in the Java community, you can be confident of your idea becoming a reality. **Visit our website or e-mail us at enterprise@n-ary.com for more details.**

nary

www.n-ary.com

ADVERTISER INDEX

ADVERTISER	URL	PH	PG
4TH PASS	WWW.4THPASS.COM	877.484.7277	91
BUZZEO	WWW.BUZZEO.COM		29
CAREER CENTRAL	WWW.CAREERCENTRAL.COM/JAVA	888.946.3822	73
CAREER OPPORTUNITY ADVERTISERS		800.846.7591	110-125
CHICAGO INTERNET WORLD 2000			79
CIMMETRY SYSTEMS, INC.	WWW.CIMMETRY.COM	800.361.1904	86
COMPUTER JOBS.COM	WWW.COMPUTERJOBS.COM		11
COMPUTERWORK.COM	WWW.COMPUTERWORK.COM		32
COMPUWARE	WWW.COMPUWARE.COM/NUMEGA	800.4.NUMEGA	15
CONCENTRIC NETWORK	WWW.CONCENTRICHOST.NET	800.476.0196	107
DEVELOPMENTOR	WWW.DEVELOP.COM/COURSE/IJAVA.HTM	800.699.1932	86
ELIXIR TECHNOLOGY	WWW.ELIXIRTECH.COM/DOWNLOAD/	65 532.4300	35
EMBARCADERO	WWW.EMBARCADERO.COM/ADMINISTER		85
EMBARCADERO	WWW.EMBARCADERO.COM/DESIGN		87
EMBARCADERO	WWW.EMBARCADERO.COM/DEVELOP		89
EPIC EDGE	WWW.EPICEDGE.COM	888.756.1665	98
EVERGREEN INTERNET, INC.	WWW.EVERGREEN.COM		75
FIORANO SOFTWARE, INC.	WWW.FIORANO.COM	408.354.3210	61
FLASHLINE	WWW.FLASHLINE.COM	800.259.1961	41
GEEK CRUISES	WWW.GEEKCRUISES.COM	650.327.3692	98
GEMSTONE	WWW.GEMSTONE.COM/WELCOME		17
HOTDISPATCH.COM	WWW.HOTDISPATCH.COM		47
IAM CONSULTING	WWW.IAMX.COM	212.580.2700	59
INETSOFT TECHNOLOGY CORP	WWW.INETSOFTCORP.COM	732.235.0137	20,77
INFORMATION ARCHITECTS	WWW.IA.COM		25,27
INSIGNIA SOLUTIONS	WWW.INSIGNIA.COM/JEODE	800.848.7677	13
INTUITIVE SYSTEMS, INC	WWW.OPTIMIZEIT.COM	408.245.8540	51
IONA	WWW.IONA.COM		105
JAVACON2000	WWW.JAVACON2000.COM		80-81,83
JAVAONE	WWW.JAVA.SUN.COM/JAVONE/	888.886.8769	109
JDJ STORE	WWW.JDJSTORE.COM	888.303.JAVA	99
KL GROUP INC.	WWW.KLGROUP.COM/INTERFACE	888.328.9596	23
KL GROUP INC.	WWW.KLGROUP.COM/COURSE	888.328.9596	103
KL GROUP INC.	WWW.KLGROUP.COM/COLLECT	888.328.9596	128
METAMATA, INC.	WWW.METAMATA.COM	510.796.0915	95
MICROSOFT	MSDN.MICROSOFT.COM/TRAINING		4
MODIS SOLUTIONS	WWW.IDEA.COM	703.821.8809	57
N-ARY	WWW.N-ARY.COM		108
NEW ATLANTA	WWW.SERVLETEXEC.COM	678.366.3211	31
NO MAGIC	WWW.MAGICDRAW.COM		5
NORTCHWOODS SOFTWARE CORPORATION	WWW.NWOODS.COM	800.226.4662	99
POINTBASE	WWW.POINTBASE.COM/JDJ	877.238.8798	37
PRAMATI	WWW.PRAMATI.COM/J2EE.HTM	914.876.3007	97
PROGRESS SOFTWARE	WWW.SONICMQ.COM/AD1.HTM	800.989.3773	2
PROSYST	WWW.PROSYST.COM	678.366.5075	55
PROTOVIEW	WWW.PROTOVIEW.COM	800.231.8588	3
QUICKSTREAM SOFTWARE	WWW.QUICKSTREAM.COM	888.769.9898	88
SEGUE SOFTWARE	WWW.SEGUE.COM	800.287.1329	19
SIC CORPORATION	WWW.SIC21.COM	822.227.398801	93
SILVERSTREAM	WWW.SILVERSTREAM.COM		127
SLANGSOFT	WWW.SLANGSOFT.COM/CODE/SPIRUS.HTML		43
SOFTWARE AG	WWW.SOFTWAREAG.COM/BOLERO	925.472.4900	63
SOFTWIRED	WWW.SOFTWIRED-IN.COM/IBUS		39
STARBASE	WWW.STARBASE.COM	888.STAR700	71
STERLING SOFTWARE	WWW.COOLJOECHALLENGE.COM		64-65
SYBASE INC.	WWW.SYBASE.COM/PRODUCTS/EASERVER	800.8.SYBASE	45
THE OBJECT PEOPLE	WWW.OBJECTPEOPLE.COM	613.569.8855	49
TIDESTONE TECHNOLOGIES	WWW.TIDESTONE.COM	800.884.8665	33
TOGETHERSOFT LLC	WWW.TOGETHERSOFT.COM	919.772.9350	6
UNIFY	WWW.EWAVECOMMERCE.COM		21
VISICOMP, INC.	WWW.VISICOMP.COM	831.335.1820	53
VISUALIZE INC.	WWW.VISUALIZEINC.COM	602.861.0999	99
VSI		800.556.VSI	67
WINTERTREE SOFTWARE	WWW.WITNERTREE-SOFTWARE.COM	800.340.8803	28
XML DEVCON 2000	WWW.XMLDEVCON2000.COM		100-101
YOUCENTRIC	WWW.YOUCENTRIC.COM/NOBRAINER	888.462.6703	69

JavaOne

www.ava.sun.com/javaone/

Career Opportunities

Career Opportunities

Career Opportunities

Career Opportunities

Career Opportunities

Career Opportunities

Career Opportunities

Career Opportunities

Career Opportunities

Career Opportunities

Career Opportunities

Career Opportunities

Career Opportunities

Career Opportunities

Career Opportunities

Career Opportunities

'Hello Internet!'

WRITTEN BY ARTHUR VAN HOFF

When Edison invented electric lighting, his company was simultaneously selling generators, switches and lightbulbs. At the time he invented it, there was no preexisting infrastructure for electricity, so he had to sell his customers every component of his direct-current electricity system. Many early electricity customers were the pioneers of their day and spent countless hours building, deploying and debugging these systems...only to find out a few years later that *alternating* current electricity systems would become the eventual standard.

In today's world, many years later, electricity has become ubiquitous – but not without having gone through major transformations (no pun intended). Gigantic investments in infrastructure systems had to be made to bring electricity to our doorsteps. The electricity grid has become a nationwide network and is powered by enormous centralized plants making use of exotic power sources such as nuclear, hydro and coal. The electricity network has become highly decentralized and very reliable, and has many redundant components. The business model has evolved from the selling of hardware to the selling of the electricity itself. Today, when you turn on a light, a microbilling system will automatically charge you a couple of cents more on your monthly bill. Electricity has become a service; it has become mainstream.

The Internet is only a few years old and still has a long way to go before it reaches the same maturity as the electricity networks. We often delude ourselves into thinking that the current protocols and infrastructure are here to stay. Yet, in truth, we need to make many more mistakes before Internet services can achieve the ubiquity and ease of use needed for true consumer acceptance. What's the Internet equivalent of AC power? One thing's for sure: many more years of infrastructure evolution are needed to enable new high-bandwidth consumer services.

It can take many years to establish even the simplest protocols. When the telephone was first invented by Alexander Graham Bell, the technology was so new that nobody knew how to answer a ringing phone. People would stare at it, pick up the receiver, listen for a while, then hang up, frustrated when nothing could be heard. Bell tried to overcome this by training people to call out "Hoy-Hoy" when answering a phone. It sounds incredible, but it wasn't until years later that Edison's marketing machine successfully introduced the word "Hello" instead. (He promoted the use of the word "Hello" by making his telephone operators wear a badge saying: "Hello, my name is...")

Standardizing Internet protocols and infrastructure is a great challenge, but it represents the first step toward true global ubiquity. Technology now progresses so rapidly that it's often outdated before it can become widely accepted. Once the rate of change slows down, it will become possible to create the great power companies of the Internet. Today, if you visit the great data centers of the Internet, you'll see huge bundles of cables disappearing into a forest of cages, filled with equipment from many different suppliers with various degrees of compatibility. Building a service based on today's technology is like a Tinkertoy problem, but the pieces don't necessarily always fit.

The current rate of change has led to the flowering of a lively industry providing the latest gadget to make your network work for you, but the result is a continuous tug-of-war between centralization and decentralization, replication and caching, software and appliance, insourced and outsourced services, thick and thin computing, low and high bandwidth. The challenges involved in building Internet infrastructure have created a great need for tools to manage complex distributed infrastructure systems on a scale never seen before. This trend is likely to continue. It is still early days for the Internet. Many years of change and rapid growth lie ahead of us. But first we need to figure out the Internet equivalent of "Hello."

AUTHOR BIO
Arthur Van Hoff is chief technical officer of Marimba, Inc., located in Mountain View, California.

avh@marimba.com

Silverstream

www.silverstream.com

KL Group Inc.

www.klgroup.co/collect