INTEROPERABILITY • PORTAL • WEB SERVICES • JAVABEANS

WebSphere® DEVELOPER'S JOURNAL

The World's Leading Independent WebSphere Developer Resource

WebSphereDevelopersJournal.com

OCTOBER VOLUME 2 ISSUE 10

DISPLAY UNTIL DECEMBER 31, 2003
$8.99US $9.99CAN
10
0 09281 03422 3

SYS-CON MEDIA

WEBSPHERE STRATEGY FOCUSES ON TRIED-AND-TRUE

INTERVIEWED BY JACK MARTIN PAGE 38

CANDLE CORPORATION

WWW.CANDLE.COM/WWW1/AXA1_WSDJ

CANDLE CORPORATION

WWW.CANDLE.COM/WWW1/AXA1_WSDJ

FROM THE EDITOR

Profit Without Honor

BY JACK MARTIN

The latest way to make a buck is to get your company some very cheap labor using L-1 visas. You can pay the new workers less than minimum wage – and it's completely legal.

The way the L-1 visa program works is simple. All L-1 employees must have been employed by the company outside of the U.S. for at least one of the three years preceding the transfer. It doesn't matter if the worker was directly employed by the sponsor, or paid through a personnel agency, or even on a freelance basis, provided the sponsor had management and control of the worker during the qualifying year.

The standard of proof for managers and executives is quite strict – they must generally supervise other professional or managerial staff and/or direct and control the day-to-day operations of a significant function, unit, or subdivision of the employer. Specialized knowledge workers, however, qualify relatively easily; any employee possessing familiarity with the employer's specific products, procedures, or methods can qualify.

L-1 visas allow companies to transfer workers from overseas offices to the U.S. for up to seven years, ostensibly to familiarize them with corporate culture or to import workers with "specialized knowledge."

It also lets companies continue to pay workers the same wage they earned in their country of origin. Indian workers receive roughly one-sixth the hourly wage of the average U.S. programmer, who makes about $60 per hour in wages and benefits.

These companies may provide L-1 workers with additional compensation so they can continue to eat in the U.S. and may also pay for housing, transportation, and medical care. Some even claim that the package ends up costing significantly more than hiring a U.S. worker. Ask anyone who works for one of these companies why they don't take this package, if it's such a good deal.

I wonder why the publicly traded companies that use L-1 workers won't disclose how many they import? Many bring in workers through consulting firms, usually Indian companies.

One executive of a public company stated, "I don't think this issue is restricted to 'cheaper' labor, but in many cases, cheaper and better. College graduates in the U.S. have two things going against them: their expectations are too high, and they're not motivated. I blame this largely on colleges and universities that instill an attitude of entitlement in our kids for four years."

Yes, dear executive, our children have come to expect the American dream of owning a home and a car, along with having freedom of choice and the right to pursue happiness. And – unlike L-1 workers – not to have to worry that their boss will wake up one day and decide that today is their last day with the company, at which point they can be quickly thrown out of the U.S. This threat brings the motivation to kiss up to your boss to a whole new level.

The State Department issued 28,098 L-1 visas in the past six months. Charlie Oppenheim, the State Department's chief of immigrant visa control, recently stated that the number of L-1 workers in the U.S. is likely much higher, as each visa allows a worker to enter the U.S. multiple times over several years. There is no limit on the number of L-1 workers companies may import each year.

The L-1 visa program is the single greatest state-sanctioned human injustice in the U.S. since the abolition of slavery. The L-1 visa program creates a class of residents in this country who have none of the opportunities that turned the U.S. into a beacon of freedom to the world. The freedom to do as well as one can – this is what attracted many of the greatest minds of the 19th and 20th centuries to come here – a better life for themselves and their families. It is exactly what made America great.

Think for a moment how different this country would be if the L-1 visa program had been in effect for the past 100 years.

My mother's parents came from Sicily in 1920. If they had come here on an L-1 visa they would have been sent back long before my mother was born, so she would have never met my father, and this editorial wold not exist. The very magazine in your hands would vanish, for the publisher of this magazine is also an immigrant. And probably more important to you – unless your lineage is exclusive to individuals who came to the U.S. before 1903 – you and your family would not exist either.

ABOUT THE AUTHOR... Jack Martin, editor-in-chief of ***WebSphere Developer's Journal***, is cofounder and CEO of Simplex Knowledge Company, an Internet software boutique specializing in WebSphere development. Simplex developed the first remote video transmission system designed specifically for childcare centers, which received worldwide media attention; and the world's first diagnostic-quality ultrasound broadcast system. E-MAIL... jack@sys-con.com

WebSphere® DEVELOPER'S JOURNAL

ADVISORY BOARD

Richard Arone, David Caddis, Mike Curwen, Devi Gupta, Lloyd Hagemo, Barbara Johnson, Shannon Lynd, James Martin, Doug Stephen, Victor Valle

Editor-in-Chief	JACK MARTIN
Executive Editor	GAIL SCHULTZ
Contributing Editor	PATTI MARTIN
Product Review Editor	JAY JOHNSON
Special Projects Editor	RICHARD ARONE
Managing Editor	JEAN CASSIDY
Editor	NANCY VALENTINE
Associate Editors	JAMIE MATUSOW JENNIFER VAN WINCKEL

WRITERS IN THIS ISSUE

Colette Burrus, Alex El Homsi, Girish Kulkarni, Harvey Gunther, Jay Johnson, Christina Lau, Didier Le Tien, Boris Lublinsky, Raju Mukherjee, Barry Nance, Jack Martin, Ashim Ranjitkar, Soutik Singha, Brad Van Horne, Colin Yu

SUBSCRIPTIONS

For subscriptions and requests for bulk orders, please send your letters to Subscription Department.
SUBSCRIBE@SYS-CON.COM
Cover Price: $8.99/Issue Domestic: $149/YR (12 Issues)
Canada/Mexico: $169/YR Overseas: $179/YR
(U.S. Banks or Money Orders)
Back issues: $15 U.S. $20 All others

PRESIDENT AND CEO	FUAT KIRCAALI
PRESIDENT, SYS-CON EVENTS	GRISHA DAVIDA
GROUP PUBLISHER	JEREMY GEELAN
TECHNICAL DIRECTOR	ALAN WILLIAMSON
SENIOR VP, SALES & MARKETING	CARMEN GONZALEZ
VP, INFORMATION SYSTEMS	ROBERT DIAMOND
VP, SALES & MARKETING	MILES SILVERMAN
PRODUCTION CONSULTANT	JIM MORGAN
FINANCIAL ANALYST	JOAN LAROSE
ACCOUNTS RECEIVABLE	KERRI VON ACHEN
ACCOUNTS PAYABLE	BETTY WHITE
ADVERTISING DIRECTOR	ROBYN FORMA
DIRECTOR OF SALES AND MARKETING	MEGAN MUSSA
ADVERTISING SALES MANAGER	ALISA CATALANO
ASSOCIATE SALES MANAGERS	CARRIE GEBERT KRISTIN KUHNLE
CONFERENCE MANAGER	MICHAEL LYNCH
NATIONAL SALES MANAGER	SEAN RAMAN
ART DIRECTOR	ALEX BOTERO
ASSOCIATE ART DIRECTORS	LOUIS F. CUFFARI RICHARD SILVERBERG TAMI BEATTY
WEB DESIGNERS	STEPHEN KILMURRAY CHRISTOPHER CROCE
CONTENT EDITOR	LIN GOETZ
CIRCULATION SERVICE COORDINATORS	NIKI PANAGOPOULOS SHELIA DICKERSON EDNA EARLE RUSSELL
CUSTOMER RELATIONS MANAGER	RACHEL MCGOURAN

EDITORIAL OFFICES

SYS-CON Publications, Inc.
135 Chestnut Ridge Road, Montvale, NJ 07645
Telephone: 201 802-3000 Fax: 201 782-9637
SUBSCRIBE@SYS-CON.COM
WebSphere® Developer's Journal (ISSN# 1535-6914)
is published monthly (12 times a year).
Postmaster send address changes to:
WebSphere Developer's Journal, SYS-CON Publications, Inc.
135 Chestnut Ridge Road, Montvale, NJ 07645

WORLDWIDE NEWSSTAND DISTRIBUTION
CURTIS CIRCULATION COMPANY, NEW MILFORD, NJ

FOR LIST RENTAL INFORMATION:
KEVIN COLLOPY: 845 731-2684, KEVIN.COLLOPY@EDITHROMAN.COM
FRANK CIPOLLA: 845 731-3832, FRANK.CIPOLLA@EPOSTDIRECT.COM

WILY TECHNOLOGY

WWW.WILYTECH.COM

Part 2: Enhance your productivity with EJB QL

Step-By-Step EJB 2.0 Inheritance in WebSphere

BY RAJU MUKHERJEE, ASHIM RANJITKAR, AND SOUTIK SINGHA

ABOUT THE AUTHOR

Raju Mukherjee is a senior architect at Noospherics Technologies, Inc., with wide experience in industry and government. He has worked extensively with J2EE, WebSphere, and databases; and has led various WebSphere-based projects for many enterprises. Raju contributes articles on the WebSphere family of products to various magazines.

E-MAIL

raj_mukherjee@yahoo.com

One of the vital principles of object-oriented programming is inheritance. Although not formally supported by the EJB specification, the need for inheritance in the EJB world has real importance.

WebSphere Studio Application Developer supports inheritance of EJBs using a direct single-table approach (single table mapping) or a parent-child approach in which foreign keys are used between the master detail tables (root/leaf-table mapping). EJBs can inherit attributes and methods from a non-EJB super class, and the remote interface can inherit ("implement" in terms of Java) an abstract interface that is a collection of the business methods needed. But these types of inheritance do not have any use from a database point of view because they do not reflect the inheritance in the database schema and the tables used for persistence. In practical use, one EJB should inherit CMP (container-managed persistence) fields, relationships, methods, etc., from another EJB given the limitation that forces every EJB involved in the process of inheritance (parent and children) to be packaged in the same EJB module. Key EJB fields are shared among all the EJBs involved in the inheritance model.

The EJB inheritance supported in WebSphere Studio is specific to the IBM extension of the EJB specification and is restricted to only Entity beans. WebSphere Studio supports three main types of mapping schemes in the EJB inheritance model. In the single table mapping scenario all classes involved in the model use the same table for mapping and one column is used for mapping purposes. In a root/leaf table mapping, one table is used for each EJB involved in the inheritance hierarchy model. Each child bean's table (a leaf) has a foreign key pointing to the primary key of the parent EJB table (the root).

Root/leaf table mapping is much more efficient than single table mapping because in the latter scheme introduction of a new EJB might change the structure of the single underlying table. In distinct table mapping, each class in the inheritance table maps to a different table but with no relationships associated. This poses the problem of data redundancy and increases the chances of data inconsistency. WebSphere Studio doesn't support this scheme for CMP beans.

Now let's take an example of EJB inheritance in which a financial institute offers its clients different types of credit cards such as gold and platinum. Let's start creating the EJBs. We will discuss each step in detail.

Create the EJBs and Inheritance

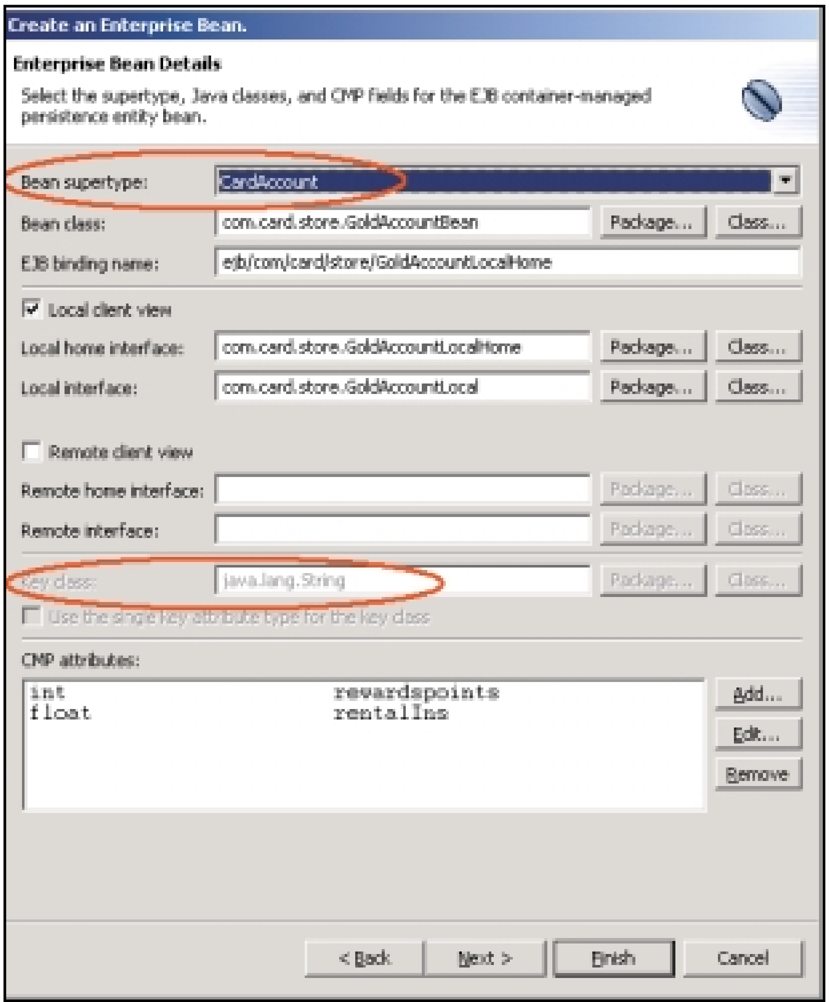
1. Create a new Enterprise Application Project "cardsonline" and create only the EJB module "cardsonlineEJB".
2. Switch to the J2EE perspective and select the J2EE Hierarchy tab. On the Project hierarchy, expand EJB Modules and select the cardonline-EJB module.
3. Right-click on the module selected, New > Enterprise bean. We will repeat the process to create three CMP 2.0 entity beans, Card-Account, GoldAccount, and PlatinumAccount.
4. Create CardAccount. Follow the same steps to create CardAccount, which will have four attributes, namely, cardnum and a key field of java.lang.String type, along with cLimit and apr of type float. Remember to choose the local interface for this bean.

   We will use CardAccount as our base class for creating the inheritance. Since both gold and platinum accounts are special types of card accounts, they will inherit the basic properties of a card account type, but they will also have their own business-specific properties. In other words, both are of card account type but individually they are different.

   There are some differences in creating gold and platinum account beans. For example, you need to select "CardAccount" as the bean supertype, and you should notice that while adding the attributes to the beans, the key field check box is disabled. Also notice that, in the Enterprise Bean details page of the bean creation wizard, the key class will automatically be assigned to the java.lang.String class, which is the type of the primary key attribute of the superclass (see Figure 1). This is because, you have selected CardAccount to be the superclass of these beans, and therefore its primary key is inherited by these beans, which are essentially subclasses of the CardAccount bean. Let's create these beans now.

5. Create GoldAccount. The GoldAccount CMP entity bean is a subclass of the "CardAccount" CMP bean, which has two attributes, *rewardpoints* and *rentalIns*, where rewardpoints is of type int, and rentalIns is of type float. While creating this bean using the wizard, remember to set up "CardAccount" as a bean supertype and check "Local Client View," as shown in Figure 1.
6. Create PlatinumAccount. The PlatinumAccount CMP entity bean is a subclass of the "CardAccount" CMP bean. It has three attributes: *miles* is of type int, *flightIns* is of type float, and *purchasewarranty* is of type boolean. Designate "CardAccount" as a bean supertype and also remember to check "Local Client View".

Once you have created all three beans, look at the EJB hierarchy pane. Notice that two subclasses of CardAccount are placed under it instead of directly under the EJB project (see Figure 2). Now that we have created our entity beans, we need to map them to the relational database tables, which are the persistent store for all the account data.


Create an Enterprise Bean.

Enterprise Bean Details

Select the supertype, Java classes, and CMP fields for the EJB container-managed persistence entity bean.

Bean supertype: CardAccount
Bean class: com.card.store.GoldAccountBean
EJB binding name: ejb/com/card/store/GoldAccountLocalHome
[✓] Local client view
Local home interface: com.card.store.GoldAccountLocalHome
Local interface: com.card.store.GoldAccountLocal
[ ] Remote client view
Remote home interface:
Remote interface:
Key class: java.lang.String
[ ] Use the single key attribute type for the key class

CMP attributes:

```
int        rewardspoints
float      rentalIns
```

FIG. 1: CREATE THE CMP ENTITY BEAN THAT WILL BE INHERITED

In this example we will create the tables after we have created the EJBs, and map the EJB attributes to table columns, which is known as the *top-down* approach to EJB-to-RDB mapping. If you have already created the database and table, you would instead use the *meet-in-the-middle* mapping approach. In our case we will create four tables, each represented by an EJB. But you might have a situation where the table is not normalized and all the data exists in just one table, Account. In that case you would use the meet-in-the-middle approach for mapping, even though you have created three beans to represent the account information.

EJB-to-RDB Mapping

1. Right-click to select the CardsOnline EJB project > Generate > EJB to RDB Mapping...
2. On the first page of the wizard, select any preexisting back-end folder, or just accept the default and click Next.
3. On the next page "top down" will be selected by default. Click Next.
4. On the next page, select the database type, which in our case is DB2. Type in the target database name, "cardline", where you want to create your tables. (If you don't have a database yet, create a database separately, since WebSphere Studio won't generate the DDL for creating a new database.) Give "account" as a schema name. Make sure that the Generate DDL check box is checked. Click Next.
5. Root/leaf mapping: On this page, you will create the root/leaf table mapping. Select all the children of the base CardAccount bean. Notice the discriminator column; this represents a discriminator column in the base cardaccount table for identifying the account type as gold or platinum. Click Finish
6. Upon successful creation of the mapping, WebSphere Studio will open the mapping editor. Save the mapping. To further your understanding, we recommend that you examine each column created, its properties, and the relationship created.

ABOUT THE AUTHOR

Ashim Ranjitkar is a computer engineer with a comprehensive knowledge of and experience in systems design and development in C, C++, and Java. Ashim is also Sun and IBM certified and has years of experience with the WebSphere family of products.

E-MAIL
ashim@noospherics.com

ABOUT THE AUTHOR

Soutik Singha is a senior enterprise architect and a certified systems expert at Noospherics Technologies. He has 10 years of experience in IT, working in several industries and in roles including mentor, tech lead, and architect. Soutik has written a number of articles on aspects of WebSphere.

E-MAIL
soutik@yahoo.com

Create Tables on the Database from WebSphere Studio

Select the J2EE navigator view under the J2EE perspective and expand the cardonlineEJB folder. Under ejbModule/META-INF/back-ends/DB2UDBNT_V72_1, right-click select table.ddl > Run on Database Server. In the wizard select all DDL statements to be executed. In the subsequent page create a connection that will be used to create a connection to the database. Click finish. This will create all required tables.

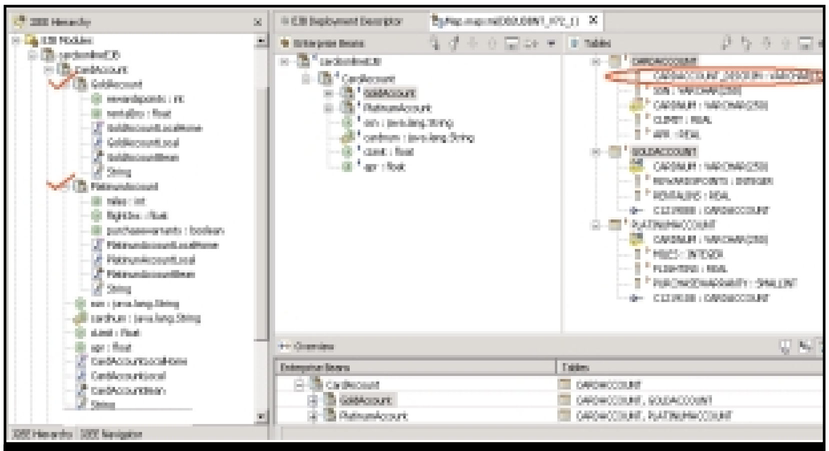


FIG. 2: INHERITANCE HIERARCHY AND CORRESPONDING RDB MAPPING

EJB QL

The EJB 2.0 specification defines a query language, EJB QL, to define the finder and select methods in the CMP entity beans. The specification also provides ways to navigate from one CMP entity bean to others through relationships, inheritance, etc. The three basic clauses are SELECT, FROM, and WHERE. The specification supports parameterized queries in which you can supply parameters as values to be used during execution. EJB QL also provides many other SQL features, including the use of clauses such as DISTINCT, NULL, AND, OR, NOT, AS, IN, BETWEEN, LIKE, OF, AS, IS, TRUE, FALSE, etc., and functions such as LENGTH, SUBSTRING, CONCAT, and LOCATE, along with string and arithmetic functions. In our example of card accounts we will use EJB QL to write a finder method within cardAccount bean to find all of the accounts. Later we will define a query in which we want to find all accounts of type gold.

1. Open the EJB Deployment descriptor and select the Bean tab. From the bean list select Customer, since the return type is customer. Scroll down to the Queries section on the same page and add a new query. This will bring up the Enterprise JavaBean 2.0 Query Wizard.
2. Select Method as New and Method Type as find method. Give the findermethod name as findAccounts. Now we need to add the parameter name. Add three parameters as apr – type float. Select Return Type as java.util.Collection. Click Next.
3. In the EJBQL page, select a sample query dropdown listbox. This is the supplied template you can choose from for the query. We will select the "Find All Query" template. This will create a query statement like "select object(o) from CardAccount o". We need to edit the query for the report we have described earlier. The edited query should be:

```
SELECT OBJECT(a) FROM
CardAccount a WHERE a is of type
(GoldAccount) and a.apr >?1
```

Click Finish. You will see the newly created query under the queries section in the descriptor. Save the EJB Deployment descriptor (see Figure 3).

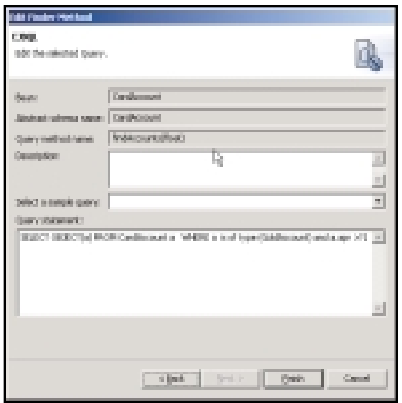


FIG. 3: DEFINING THE EJB QL QUERY AS A FINDER METHOD FOR CARDACCOUNT BEAN

Test the EJB Using a Test Client

1. Create a new WebSphere 5.0 Server and Server configuration. Then create a datasource for the cardline database with the JNDI name "jdbc/cardline". Open the EJB deployment descriptor and under "JNDI - CMP Factory Connection Binding" again type in "jdbc/cardline". Save the deployment descriptor.
2. Generate the EJB deployment code by selecting the EJB module, "cardonlineEJB", right-click > Generate > Deploy > RMIC Code.
3. Right-click on the EJB module and select Run on Server. Select the created server to run on. Upon successful server startup and EJB load, the universal test client will automatically open.
4. Using the test client you can enter data into the tables and then invoke findAccounts(float) in the CardAccountLocalHome object. You could also pass the "apr" value as parameter to get all the accounts which are of type gold.

Conclusion

As you can see, the introduction of a standard query language has made the EJBs portable, so now the developers don't need to worry about SQL portability. At the same time you can see that EJB inheritance will enhance productivity by allowing reuse of business logic, ease of maintenance, and extendibility.

PROLIFICS

WWW.PROLIFICS.COM/WEBSERVICES

Quick and easy development using the new Visual Editor for Java

Building Applications with IBM WebSphere Studio and JavaBeans

BY COLETTE BURRUS

In *Building Applications with IBM WebSphere Studio and JavaBeans*, my coauthor, Steph Parkin, and I take you on a guided tour of developing applications with the WebSphere Studio Visual Editor for Java and JavaBeans. With the Visual Editor, new with WebSphere Studio Version 5, you'll see how quick and easy it is to construct graphical user interfaces (GUIs) and use JavaBeans to build applications. This article gives you an introduction to the Visual Editor and shows examples of some of the applications you'll build with it when you follow the guided tour.

ABOUT THE AUTHOR

Colette Burrus retired from IBM after more than 20 years of programming and project management. Her last assignment was project manager for the IBM alphaBeans project, working with the "JavaBeans Around the World" team. She is coauthor of *VisualAge for Java for Non-Programmers* and *Building Applications with IBM WebSphere Studio and JavaBeans*.

E-MAIL

beans@pobox.com

Visual Editor for Java

The Visual Editor works very much like other visual design tools, such as VisualAge for Java's Visual Composition Editor. As you can see in Figure 1, the editor itself is composed of three parts: a palette of beans along the left edge of the window, a design surface where you visually work with beans, and just below that, a source pane with the source code for your program. The bean palette contains all the standard Java widgets for the Abstract Window Toolkit (AWT) and Swing, as well as a Choose Bean option, which you can use to select any custom beans that aren't already built into the palette.

To create a program visually in the Visual Editor, select the bean you want from the palette and drop it on the design surface, placing visual beans within the visual surface for your application and placing nonvisual beans in the free-form area. After adding the beans you want to the design surface, you can customize their properties to set the values you want to use, such as setting a button's label. As you work with beans by resizing them on the design surface or modifying their properties, the Visual Editor automatically generates the underlying code and presents it in the source pane. You can also modify code directly in the source pane, and changes that you make there are automatically reflected in the design surface and Properties view. Once you've added the beans you want and customized their properties, you can use the Visual Editor's code-assist features, such as its event-handling templates, to add the code to connect the beans together.

That's all there is to it – select the beans you want, add them to the design surface, customize their properties, and add the code to connect them together. And, if you want to test your application at any point along the way, just click the Run icon in the toolbar. The Visual Editor will launch your application, so you can test out the functions you've added so far.

JavaBeans

JavaBeans are the Java components you'll use to build applications during the guided tour. You'll use visual beans such as buttons and text fields to construct the GUI for your application and nonvisual beans to perform complex functions such as sorting data, working with databases, and working with XML documents. As you'll see, JavaBeans simplify the work of application programming because they provide prebuilt components with simple interfaces that you can use to customize their functions and connect them to your applications. You'll even learn how easy it is to build your own beans using WebSphere Studio.

When you use the Visual Editor, a hierarchy of all the beans in your visual composition appears in the Java Beans view, as shown in Figure 2. You can use the Java Beans view to rename, delete, or reorganize beans within your application, and any changes you make there are automatically reflected in the Visual Editor.

Creating the ToDoList Application

As you can see from the preceding figures, the design surface can contain both visual and nonvisual beans. For the ToDoList application, you will use visual beans such as buttons and text fields, as well as nonvisual beans such as ReadFile and WriteFile beans, to provide the logic for reading and writing files. Let's take a closer look at some of the steps to build the ToDoList application.

Typically, the first step to visually building an application is to add user interface elements to the design surface. In this case, you select a button, a text field, and a list box from the bean palette, and drop each of them onto the design surface. Once you drop the elements onto the design surface, you use the mouse or the Visual Editor's convenient alignment icons to resize the visual elements to whatever sizes you want. Then you use the Properties view to customize bean properties such as setting the label for the button bean, so that the application's visual surface looks like Figure 3.

Next, you add the code to connect the beans together. If you're an old VisualAge for Java programmer like me, your first instinct will be to try to connect the elements by wiring them together on the design surface. The Visual Editor doesn't support this type of visual connection, but you'll see that once you get used to it, the Visual Editor's content assist feature for adding event handlers is just about as easy. For example, when the button is clicked, you want to add the text in the text field to the list, clear the text field to blanks, and set the focus to the text field, so you'll place this logic in an action event handler for the button. WebSphere Studio includes event-handling templates for all the common events you might want to use, such as adding an ActionListener to listen for button clicks. In the source pane, enter the start of the line of code to add an ActionListener (button.addA), and press Ctrl+Space to show the content assist window, as shown in Figure 4.

When you double-click addActionListener – addActionListener to a Component, the code for an ActionListener inner class is automatically added to your source code, as shown in Listing 1.

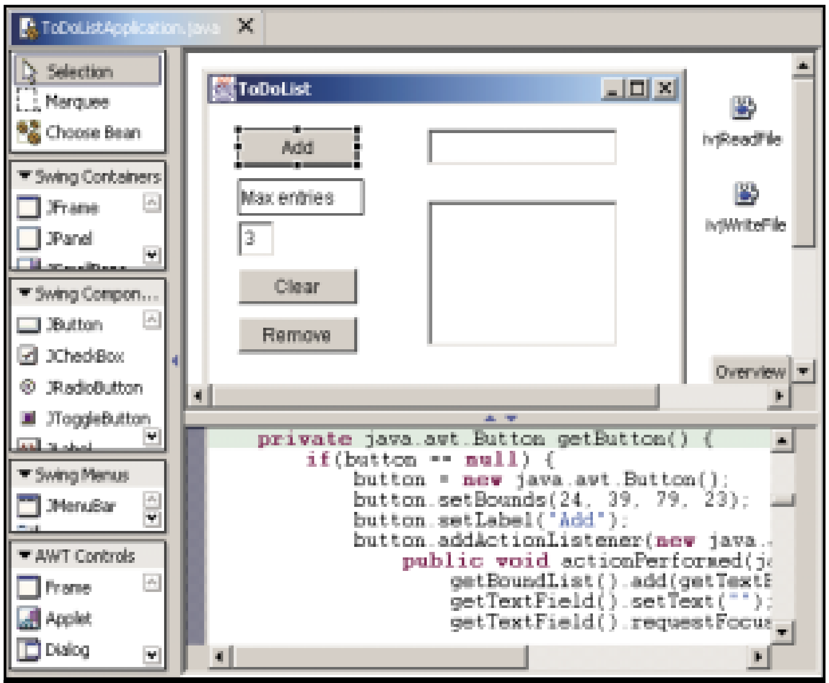
Finally, you need to add the appropriate logic to the ActionListener inner class to specify the actions that you want to take place when the button is clicked (add the text to the list, clear the text field, and set the focus to the text field), as shown in Listing 2.

To run the application, click the Run icon in the toolbar, and the Visual Editor will launch your application, as shown in Figure 5, so you can test it.

As you continue building the ToDoList application, you'll add more functionality, such as a Clear button to remove all entries from the list, a Done button to remove a selected entry from the list, and a text field to control the number of entries in the list. In each case, you drop the visual elements you want on the design surface, customize their properties, use content assist to add event handlers, and add the code for the actions you want to occur in the event handlers, such as clearing the list when the user clicks the Clear button. And for those of you who, like me, can't type Java code very well, the Visual Editor includes more code assist features to help you enter what little code you do need to write.

When you add functionality to read the list from a file and write the list to a file, you will need your first nonvisual beans (ReadFile and WriteFile), but the process is still the same. You select the beans you want, but this time you will use the Choose Bean option in the palette and drop the beans onto the free-form area of the design surface, as shown in Figure 6.

After dropping the nonvisual beans onto the design surface, you can customize their properties just as you did for visual beans, use content assist to add the appropriate event handlers, and add code for the actions you want to occur in the event handlers. When you're finished, you have the final working application shown in Figure 7, all built in about an hour – thanks to the power of the Visual Editor and JavaBeans.



```
private java.awt.Button getButton() {
    if(button == null) {
        button = new java.awt.Button();
        button.setBounds(24, 39, 79, 23);
        button.setLabel("Add");
        button.addActionListener(new java.
            public void actionPerformed(ja
                getBoundList().add(getTextF
                getTextField().setText("");
                getTextField().requestFocus
```

FIG. 1: THE TODOLIST APPLICATION IN THE VISUAL EDITOR

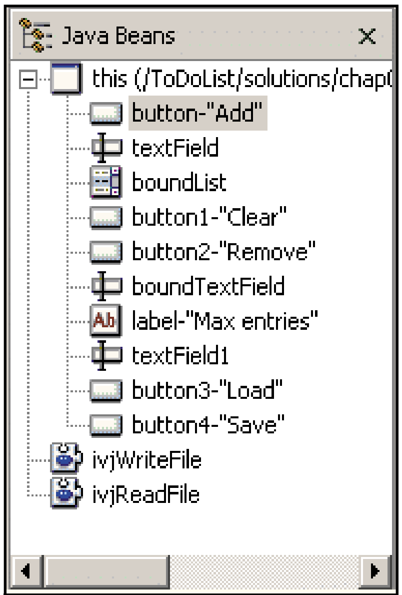


FIG. 2: THE JAVA BEANS VIEW FOR THE TODOLIST APPLICATION

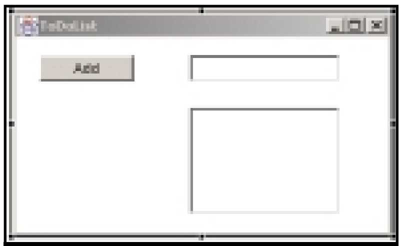
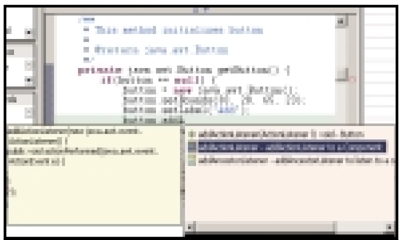


FIG. 3: THE DESIGN SURFACE WITH THE INITIAL USER INTERFACE ELEMENTS FOR THE TODOLIST APPLICATION



```
/**
 * This method initializes button
 *
 * @return java.awt.Button
 */
private java.awt.Button getButton() {
    if(button == null) {
        button = new java.awt.Button();
        button.setBounds(30, 28, 65, 23);
        button.setLabel("Add");
        button.add
```

FIG. 4: USING THE CONTENT ASSIST WINDOW TO ADD AN ACTIONLISTENER

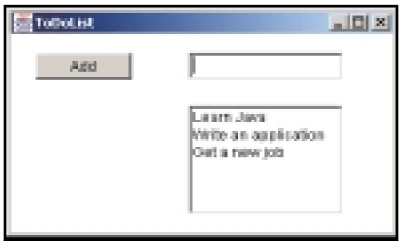


FIG. 5: THE INITIAL VISUAL ELEMENTS FOR THE TODOLIST APPLICATION IN ACTION

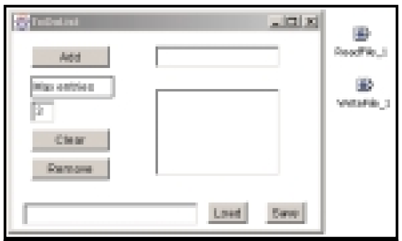


FIG. 6: NONVISUAL BEANS THAT HAVE BEEN DROPPED ONTO THE DESIGN SURFACE

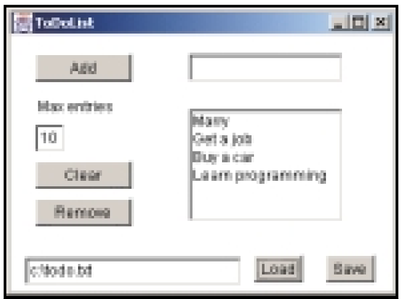


FIG. 7: THE FINAL TODOLIST APPLICATION IN ACTION

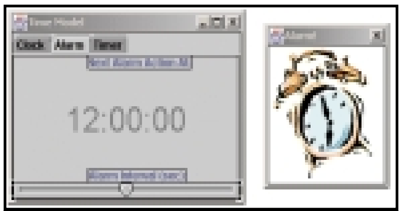


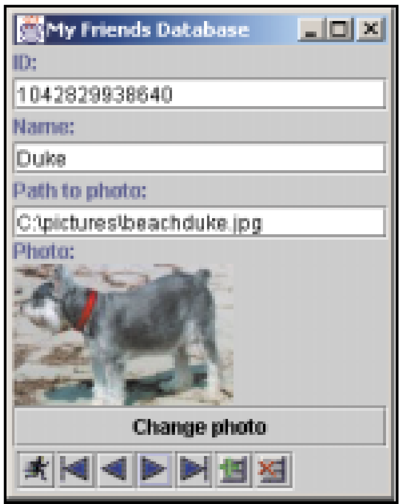
FIG. 8: THE DESIGN SURFACE FOR THE ALARM TAB IN THE TIMEMODEL APPLICATION

Want to See More?

The ToDoList application is obviously just a simple application to get you started on learning how to use the Visual Editor. Each chapter in the book covers a different topic and the complexity of the applications that you build increases as you proceed through the guided tour, so at the end you're an expert at using the Visual Editor and building applications with JavaBeans. Here are some more examples of the applications you can build.

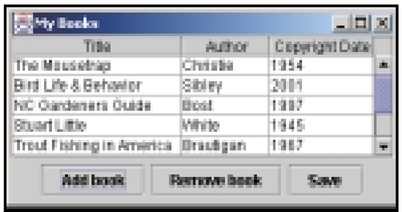
The ToDoList application shown above uses visual beans from Java's Abstract Window Toolkit, but the book also covers many Swing topics. Figure 8 shows a Swing application that uses the Swing JTabbedPane, JSlider, and JProgressBar components, and teaches the Swing Model-View-Controller architecture.

Most modern applications have some form of database access, and the book will show you how easy it is to create a database application with the Visual Editor and WebSphere Studio's DB beans for database access, as shown in Figure 9.



ID:
1042829938640
Name:
Duke
Path to photo:
C:\pictures\beachduke.jpg
Photo:

FIG. 9: A DATABASE APPLICATION THAT KEEPS TRACK OF FRIENDS



Title	Author	Copyright Date
The Mousetrap	Christie	1954
Bird Life & Behavior	Sibley	2001
NC Gardeners Guide	Bost	1997
Stuart Little	White	1945
Trout Fishing in America	Brautigan	1967

FIG. 10: AN XML APPLICATION THAT KEEPS TRACK OF BOOKS

These days, just about all data is stored in XML format, and you'll learn how easy it is to use WebSphere Studio to create DTDs and XML Schemas, to generate beans from a DTD or XML Schema file, and then to use those beans to build an XML application in the Visual Editor. Figure 10 shows an XML application that uses custom-built beans to present XML data in a Swing JTable component.

Conclusion

The Visual Editor, new with WebSphere Studio Version 5, is a great tool for quickly and easily constructing GUIs, and combined with the right JavaBeans, a truly powerful tool for building applications.

LISTING 1: THE CODE IN BOLD IS ADDED BY CONTENT ASSIST

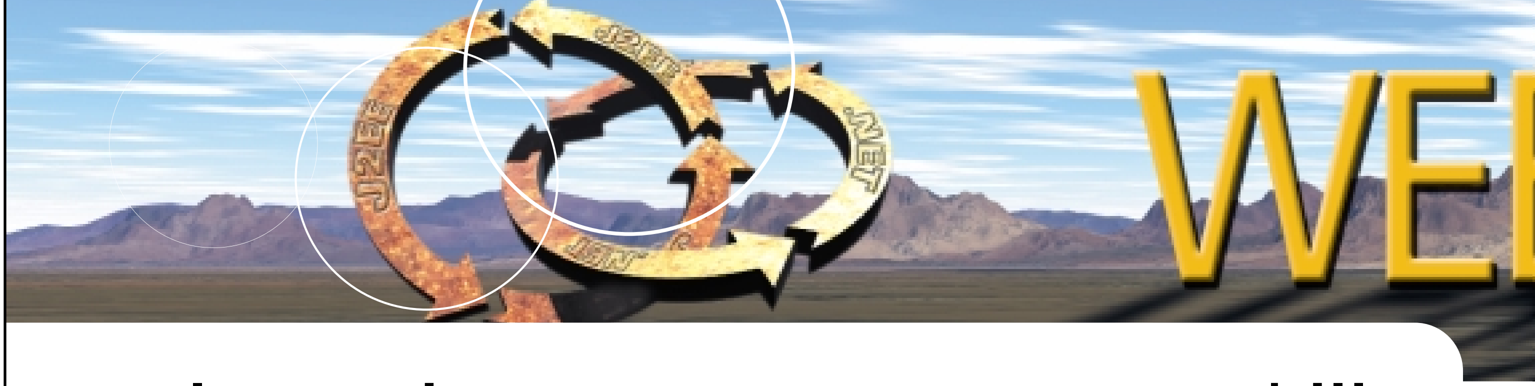
```
private java.awt.Button getButton()
{
if(button == null) {
    button = new java.awt.Button();
   button.setBounds(30, 28, 65, 23);
   button.setLabel("Add");
   button.addActionListener(new
   java.awt.event.ActionListener() {
         public void
actionPerformed(java.awt.event.Action
Event e) {

         }
   });
}
return button;
}
```

LISTING 2: ADD THE BOLDED CODE TO PROCESS ACTION EVENTS FOR THE BUTTON

```
private java.awt.Button getButton() {
if(button == null) {
   button = new java.awt.Button();
   button.setBounds(30, 28, 65, 23);
   button.setLabel("Add");
   button.addActionListener(new
java.awt.event.ActionListener() {
         public void
actionPerformed(java.awt.event.Action
Event e) {

getBoundList().add(getTextField().get
Text());

getTextField().setText("");

getTextField().requestFocus();
         }
   });
}
return button;
}
```

KENETIKS

WWW.KENETIKS.COM



Implementing J2EE-.NET Interoperability Using WebSphere MQ

Part 1: New SupportPac allows use of the same messaging middleware on both platforms

– BY BORIS LUBLINSKY AND DIDIER LE TIEN –

ABOUT THE AUTHOR

Boris Lublinsky is an enterprise architect at CNA Insurance, where he is involved in the design and implementation of CNA's integration strategy, building application frameworks and implementing service-oriented architecture for the company. Prior to joining CNA he was director of technology at Inventa Technologies, where he oversaw EAI and B2B integration implementations and development of large-scale Web applications. Boris has over 20 years of experience in software engineering and technical architecture.

E-MAIL

boris.lublinsky@cna.com

It is today's reality that most companies are using both J2EE and .NET environments for their software implementation. Until recently, the prevalent solution for integration of these two environments has been HTTP-based Web services.

Although this solution works well in many cases, it suffers from the following drawbacks:

1. Most implementations today are synchronous and based on the synchronous nature of the underlying HTTP protocol.
2. Reliable messaging is difficult to implement using HTTP.
3. Load balancing in an HTTP-based environment requires a client-side load-balancing implementation, which is usually quite difficult when compared to server-side load balancing.
4. Client and server application life cycles have to be synchronized in order for Web services–based communications to work.

Some of these problems can be easily alleviated by using messaging software that provides guaranteed message delivery, asynchronous communications, and ease of server-side load balancing.

So far, while the J2EE world has standardized on JMS, the .NET world usually relies on MSMQ and MSMQ bridges provided by third-party vendors. The release of the WebSphere MQ support for .NET – SupportPac MA7P (www-3.ibm.com/software/integration/support/supportpacs/individual/ma7p.html) – provides a new integration mechanism by using the same messaging middleware on both platforms.

We will discuss a simple interoperability example between a .NET client (via WebSphere MQ SupportPac MA7P) and a J2EE server (using WAS v5). Sample source code is provided for both environments.

Overall Implementation Architecture

The use of message queuing for communications between applications requires the introduction of additional components – queues (see the Queue Manager sidebar) – to serve as a main communication vehicle.

As shown in Figure 1, usage of queuing for application-to-application communication leads to a very loose coupling between applications. In this case applications don't talk to each other, but rather use intermediary queues. The consumer application puts a request message into the (request) queue, which is picked by the producer application actually fulfilling the request. When the request is processed, the producer application puts the result message into the (reply) queue, which is picked up by the consumer application.

Instead of depending on locations and interfaces of participating applications, queue-based communications are dependent only on the format of the messages and the queue names. In this approach, the producer does not need to know who and where the consumer is, and vice versa. It also allows unlimited scalability by providing the ability to add additional producers, all of which will listen on the same queue (see the Queue Clustering sidebar).

As stated above, the drawback of this otherwise very flexible architecture is the necessity for the consumer and producer to agree on the names of the queues they will use. In the simplest implementation, these queue names are hard-coded in both the consumer and producer; the more advanced implementations use configuration files. Although the notion of logical (or alias) queue names versus physical queue names, supported by WebSphere MQ, can further improve the solution by allowing the applications to use logical queue names that are mapped to different physical queue names, this approach still creates a fairly tight coupling between applications and queue names.

JMS, the Java implementation of messaging, overcomes this issue by introducing JNDI for storing information about queues and queue connection factories (queue managers), which allows for centralized control over queue topology. Instead of knowing the names of the queues, components need to know where in the JNDI tree this information is located. A centralized queue repository – usually LDAP based – becomes, in this case, a central point for the infrastructure configuration, allowing for queue topology changes without impacting code implementation.

Additionally, the message sender has the ability to take advantage of the notion – standardized by JMS (native to MQ) – of "Reply-to" queues, which allows it to specify the queue in which it wants to receive replies. This provides even more flexibility for overall queue-based communications.

Both JMS and WebSphere MQ provide support for two types of queues:

- ***Permanent queues:*** Queues created as part of the infrastructure; these queues exist regardless of whether or not the application exists.
- ***Temporary queues:*** Queues created by the application as part of its execution. Temporary queues can be either explicitly deleted by the application, or can be deleted by WebSphere MQ automatically when the application disconnects from the MQ server.

Request queues are usually permanent queues because they are used for initializing communications and therefore have to be known by both parties. As for the reply queue, in this case either type can be used. When choosing permanent versus temporary queues, the following should be considered:

- Temporary queues don't require infrastructure creation and maintenance for the applications to communicate correctly. Adding more client applications does not require any infrastructure activities.
- Temporary queues require a better programming discipline from the client implementation; creation of temporary queues for each request may overwhelm the queue managers and lead to worse overall performance.
- In the WebSphere MQ environment, the overall system health is often determined by monitoring the communication queue depth. It is usually much more difficult to monitor temporary queues than it is to monitor permanent queues.
- Usage of temporary queues might not be applicable in the case of asynchronous communications because the queue could be destroyed before the reply comes back from the server. If the system needs to support both synchronous and asynchronous communications, usage of temporary queues will usually lead to duplication of the overall MQ infrastructure.

Based upon the above considerations, we chose permanent queues for our implementation.

Figure 2 shows multiple consumers, each of which can invoke any of the producers. The sequence of steps for the consumer application is as follows:

1. Query LDAP to retrieve reply queue (specific to the consumer application) information using, for example, the consumer name. This queue will be used by the consumer application to receive replies from all producers.
2. Query LDAP to retrieve request queue (specific to the producer application type) information using, for example, the producer name.
3. Place a request message on the request queue. If a reply is expected, the consumer's reply queue is placed in the "Reply-to" field of the message.
4. Retrieve the reply (if required) from the reply queue.

The sequence of steps for the producer application is as follows:

1. Query LDAP to retrieve request queue (specific to the producer application type) information using, for example, the producer name.
2. Listen on the request queue for all incoming requests. In the case of "cloned" producers, all of them are competing for the message on the same queue.
3. Process the received message.
4. Check the "Reply-to" field of the request message and if a reply destination is specified, send a reply to the appropriate queue.

Queue Managers

WebSphere MQ introduces the notion of queue managers. Queue managers play a vital role in the WebSphere MQ framework, as they are not only responsible for the management of the queues but also for the routing of messages from consumers to providers. To use a queue, the consumer/provider needs to connect to the queue manager first; it then is allowed access to the queue. In JMS, this is done by establishing a connection with a queue connection factory, which enables the creation of a sender/receiver and allows access to the queue. A full description of the queue manager responsibilities can be found at www-3.ibm.com/software/ integration/wmq.

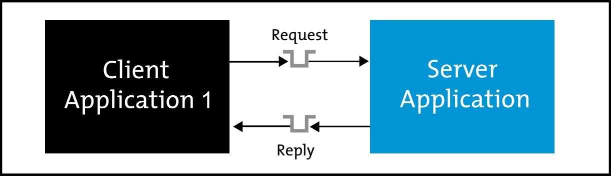


FIG. 1: THE USE OF QUEUING TO INTEGRATE APPLICATIONS

ABOUT THE AUTHOR

Didier Le Tien is an enterprise architect at CNA Insurance, where he is involved in the design and implementation of CNA's integration strategy, building application frameworks and implementing service-oriented architecture for the company. He holds a PhD in computer science from the University of Evry, in France.

E-MAIL

didier.letien@cna.com

The above architecture allows a very simple implementation for both failover and load balancing. It also allows consumers to define their own programming model (request/reply versus "fire and forget") without impacting the producer implementation. The producer implementation should always be prepared to send the reply back, whereas the consumer implementation dictates the actual interaction model.

The last issue that must be resolved for queue-based communications is the format of the messages. WebSphere MQ supports a multiplicity of message formats, including object, binary, map, and text messages. For queue-based communications between applications implemented in different languages and different operating systems, text messages (the least common denominator) are usually used as the messaging format. Combined with XML, text messaging provides a very powerful mechanism for inter-application communications.

For our example we used the architecture shown in Figure 3. We developed a request/reply consumer implementation in C# on the .NET platform, using IBM's SupportPac MA7P. The producer was implemented using message-driven beans (MDBs) in WebSphere Application Server 5. Both the consumer and the producer use IBM's Directory Server 5.1 to access queue/queue manager information. While our simple example uses text messaging, we recommend XML-based messaging for production implementations.

Queue Clustering

The architecture described in this article can be further improved by the use of the queue-clustering capability provided by WebSphere MQ. Clustering enables the physical distribution of the providers by using the same queue name on clustered queue managers. WebSphere MQ is then responsible for handling the routing and load balancing of messages to instances of the queue. By default, the round-robin algorithm is used for load balancing.

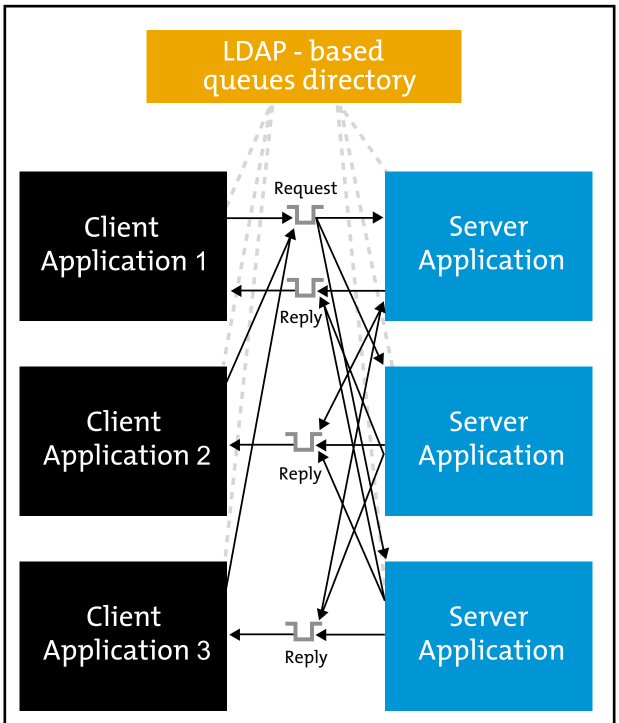


FIG. 2: CONFIGURATION WITH MULTIPLE PRODUCERS AND CONSUMERS

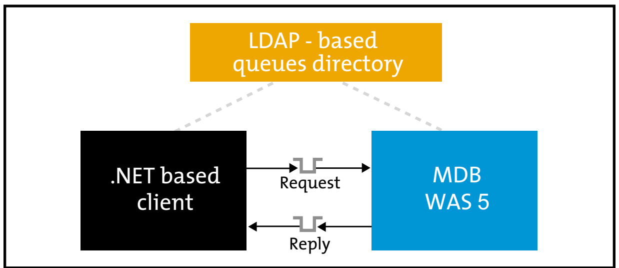


FIG. 3: EXAMPLE ARCHITECTURE

Setting Up the Execution Environment

When setting up a WebSphere MQ environment, we first decide on a client/server connection to the WebSphere MQ server. The client connection is implemented using TCP/IP–based communication with the WebSphere MQ server via a connection channel. The server connection directly binds executable code with the server software running on the same machine. Direct binding leads to better overall performance and transactional support but requires installation of the WebSphere MQ server on every machine participating in MQ interactions and thus usually leads to the use of MQ clustering. The client connection requires only WebSphere MQ client installation but leads to some performance degradation and worse overall quality of service due to additional TCP/IP communications with the server (sharing of the client connection channels leads to a further performance degradation).

The three connection options supported by the .NET implementation are implicit server binding, implicit client binding, and explicit client binding. The type of implicit binding (client/server) depends on the software installed on the given machine as checked by the WebSphere MQ implementation. In the case of the client connection, at least an address and port of the server machine must be specified in the MQEnvironment class for the connection to work. Explicit client connection allows overwriting of default behavior by specifying a server connection string (machine name/port number pair) and channel name. When using the MQ connection API, if both parameters are null, an implicit connection takes place. In our implementation we use the explicit connection APIs, thus supporting all possible connection options controlled by the input parameters.

J2EE supports the same three connection options, controlled by the definition of the queue factory object in the JNDI.

For our simple example, we installed WebSphere MQ server, a .NET client, and WAS 5 on a single machine. This installation uses server bindings for both .NET and Java.

Administrating the WebSphere MQ Objects

For our example only a handful of WebSphere MQ objects in the LDAP tree need to be defined:

- The queue manager that is shared between the consumer and provider applications
- The provider queue (request queue)
- The consumer queue (reply queue)

DIRIG SOFTWARE

WWW.EBIZPERFORM.COM

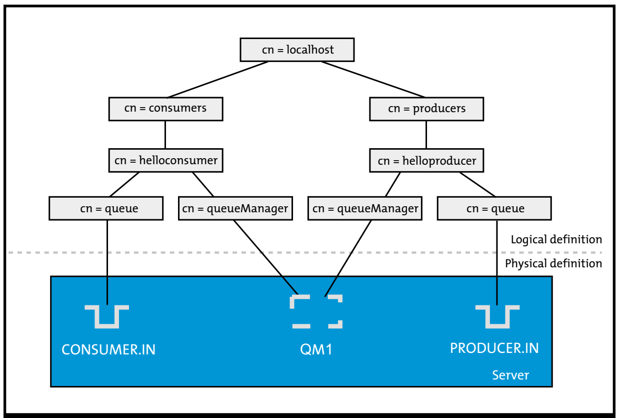


FIG. 4: LOGICAL AND PHYSICAL LDAP LAYOUT

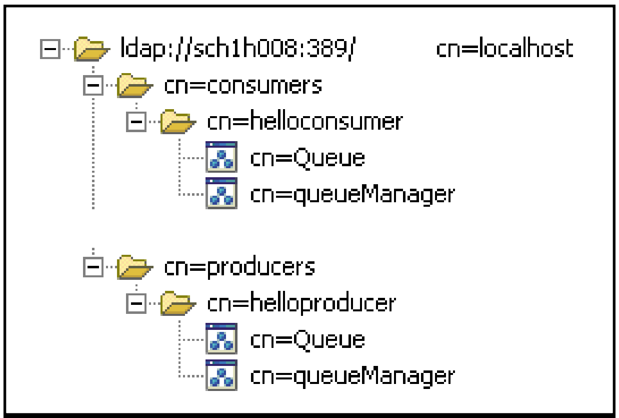


FIG. 5: LDAP DIRECTORY TREE

The logical LDAP architecture shown in Figure 4 is designed to support both our and more complex deployments.

As seen in Figure 4, the logical LDAP architecture allows for multiple consumers and producers. Each consumer/producer is defined by the queue (request for producers and reply for consumers) and queue manager it connects to. This allows every application in the system to bootstrap with the proper MQ information (as long as it knows its own name and functionality). This also allows consumer applications to refer to the required producer applications by their respective application names and resolve queue information using LDAP.

The logical layout maps to our physical layout by mapping queue names onto two different physical queues (PRODUCER.IN and CONSUMER.IN) and mapping two logical queue managers into a single physical queue manager, QM1.

WebSphere MQ object creation is done using the JMSAdmin tool. A full overview of this utility can be found in the WebSphere MQ manual, *Using Java* (http://publibfp.boulder.ibm.com/epubs/pdf/csqzaw11.pdf).

References to WebSphere MQ objects (such as queue manager and queues) are stored in the directory as Java objects. Each object stores the information needed to locate and communicate with a specific WebSphere MQ element. The objects required for our application were defined through the following steps:

1. Define a JMSAdmin configuration file to point to the LDAP directory. A typical configuration file includes information related to the directory access method (file-based, LDAP-based, or WebSphere AppServer–based), the directory URL, and the credential. A sample JMSAdmin configuration file is presented in Listing 1.
2. Use a script (see Listing 2) to automate the object creation process. Although the same queue manager is used for both the consumer and producer, they are defined slightly differently:
   - Normal queue manager for consumer
   - Transactional queue manager for the producer (MDB requirement)

Once the script is executed, the LDAP tree should look like the one shown in Figure 5.

Conclusion

We have discussed the reasons for using WebSphere MQ for a J2EE-.NET interoperability overall implementation architecture and environment setup. In the second part of this series we will discuss implementation of the proposed architecture on both the J2EE and .NET platforms.

LISTING 1: SAMPLE JMSADMIN CONFIGURATION FILE

```
# Configured to use an LDAP directory
INITIAL_CONTEXT_FACTORY=com.sun.jndi.ldap.LdapCtxFactory
#
#LDAP provider URL
PROVIDER_URL=ldap://myhosttest/cn=localhost
#
#  The following line specifies the security authentication
#  model in use,
#  and may be 'none' (for anonymous authentication), 'sim
#  ple', or 'CRAM_MD5'.
#
SECURITY_AUTHENTICATION=simple
#
# LDAP Directory root/passwd
PROVIDER_USERDN=cn=theroot
PROVIDER_PASSWORD=password

# The following line specifies a prefix to add to names when
# carrying out operations such
# as lookup/bind.
NAME_PREFIX=cn=
```

LISTING 2: SCRIPT FOR CREATION OF MQ OBJECTS IN LDAP

```
def ctx(consumers)
chg ctx(consumers)
def ctx(helloconsumer)
chg ctx(helloconsumer)
def q(Queue) queue(Reply.Q)
def qcf(queueManager) qmanager(TestQM)


chg ctx(=INIT)
def ctx(producers)
chg ctx(producers)
def ctx(helloproducer)
chg ctx(helloproducer)
def q(Queue) queue(Service.Q)
def wsqcf(queueManager) qmanager(TestQM)

end
```

QUEST SOFTWARE

HTTP://JAVA.QUEST.COM/JPROBE/WDJ

Overcome the complex challenge of handling various releases

Managing Production Deployment of Java Applications

BY BRAD VAN HORNE

Companies that have adopted WebSphere as their enterprise development platform are creating business-critical production applications. These projects include new, stand-alone solutions and modernized front ends for legacy applications and are being deployed to any eServer platform that supports IBM WebSphere Application Server.

ABOUT THE AUTHOR
Brad Van Horne brings nearly 20 years of enterprise IT sales and product management experience to his role of Integrations Product Manager at MKS (www.mks.com), an enterprise software configuration management vendor. In this role, he is responsible for the integration of MKS's core products with other industry-leading software development tools.

E-MAIL
bvanhorn@mks.com

Development teams and production control personnel are facing complex challenges. Production deployment processes and tools in this area have not matured as they have in legacy iSeries and zSeries environments. There is a need to reliably manage various releases of applications that often contain thousands of source members. Management needs a repeatable, dependable process, while developers need to be free to do their job without being burdened by external tools.

Enterprises need answers to these questions:

- How can a project team develop new functionality while providing maintenance for the production version?
- How can managers guarantee that absolutely no unplanned or unwanted changes are introduced into a production application when fixes are required?
- How can a WebSphere development team reduce the amount of time spent on manual source code merging (which can destabilize the application) and unproductive Ant makefile scripting?
- How can any WebSphere team member rebuild the application, on any machine, exactly the way it exists today?

A solution should enable production deployment teams to:

- Reliably reproduce the exact set of source code for any version of a production application
- Ensure that only desired changes have been introduced into a new build
- Control and automate deployment to test and production
- Provide a complete audit trail

The solution must allow development teams to:

- Work seamlessly from within their development IDE
- Easily follow a repeatable managed development process
- Work efficiently without distractions from external tools

The ideal solution will consist of a software configuration management/version control system; a change management/process and workflow system; and a reliable, repeatable build process. The software configuration management (SCM) system should be a secure, client/server, cross-platform application. It must seamlessly integrate with the WebSphere Studio family of products. It must provide a 100% reliable method of re-creating a past version; simple labeling is too prone to errors.

A change management application should allow for the definition of a variety of flexible development processes and be tightly integrated with SCM systems on distributed iSeries and zSeries platforms. A robust command line and event trigger mechanism should allow the change management system to trigger and control external actions in the development process. The build application should ensure that every member of the team builds the application the same way every time. Unproductive Ant XML scripting should be reduced.

Let's review a real-world scenario. In this case we'll be referring to specific functionality found in MKS Source Integrity Enterprise Edition, an enterprise SCM solution. Baselining in Source Integrity Enterprise is achieved by an action called *checkpointing*. The checkpoints are versioned, meaning that an earlier baseline can never be corrupted. Corruption is possible with solutions that rely on labeling, however, once a project has been checkpointed, new development can carry on, with full confidence that any previous release can be re-created for maintenance work.

Figure 1 shows a graphical view of a member history that a developer can use to visualize the status and history of a source file, and also to perform actions such as merge on it. This view is also available for project history.

MKS provides a Ready for WebSphere Certified version control plug-in for the WebSphere family of products. A key aspect of this integration is its unique ability to dynamically update file status decorators that provide WebSphere developers with real-time status updates. This reduces the amount of code merging required and increases the visibility of project activities, allowing developers to make informed decisions. Conflict avoidance versus conflict resolution results in superior-quality applications and increased developer productivity.

Figure 2 shows the file status indicators and the file decorators added by the MKS Source Integrity Enterprise Edition plug-in. They are updated in real time without any user action or input.

MKS Integrity Manager is a flexible, configurable development process and workflow engine. It is tightly integrated with MKS Source Integrity Enterprise for distributed platforms and MKS Implementer for iSeries, and can be integrated with zSeries-based SCM systems. This provides management with a single Web-based change management system to oversee development and to link source code changes with issues across all platforms. MKS has introduced a Build & Deploy module as part of Integrity Manager. This module provides the production deployment staff with the ability to automate, control, and report on all build and deploy activities.

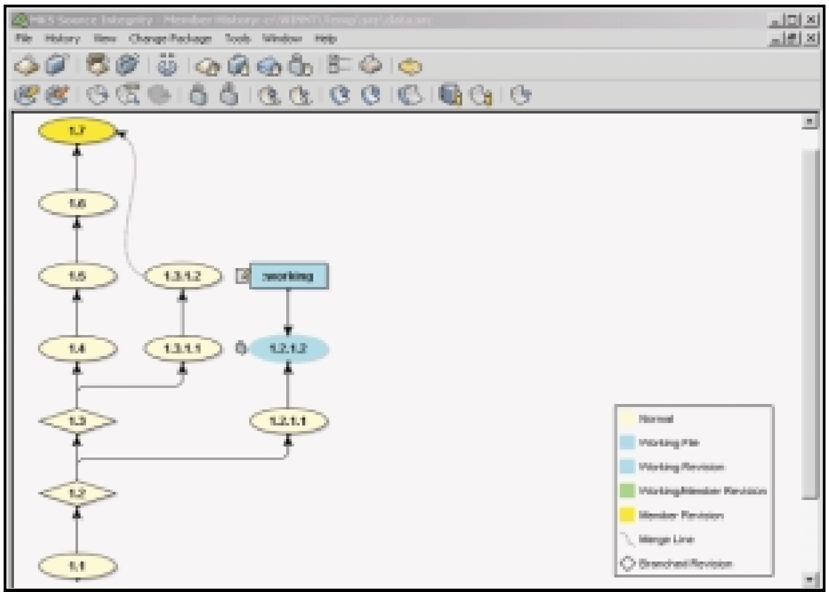
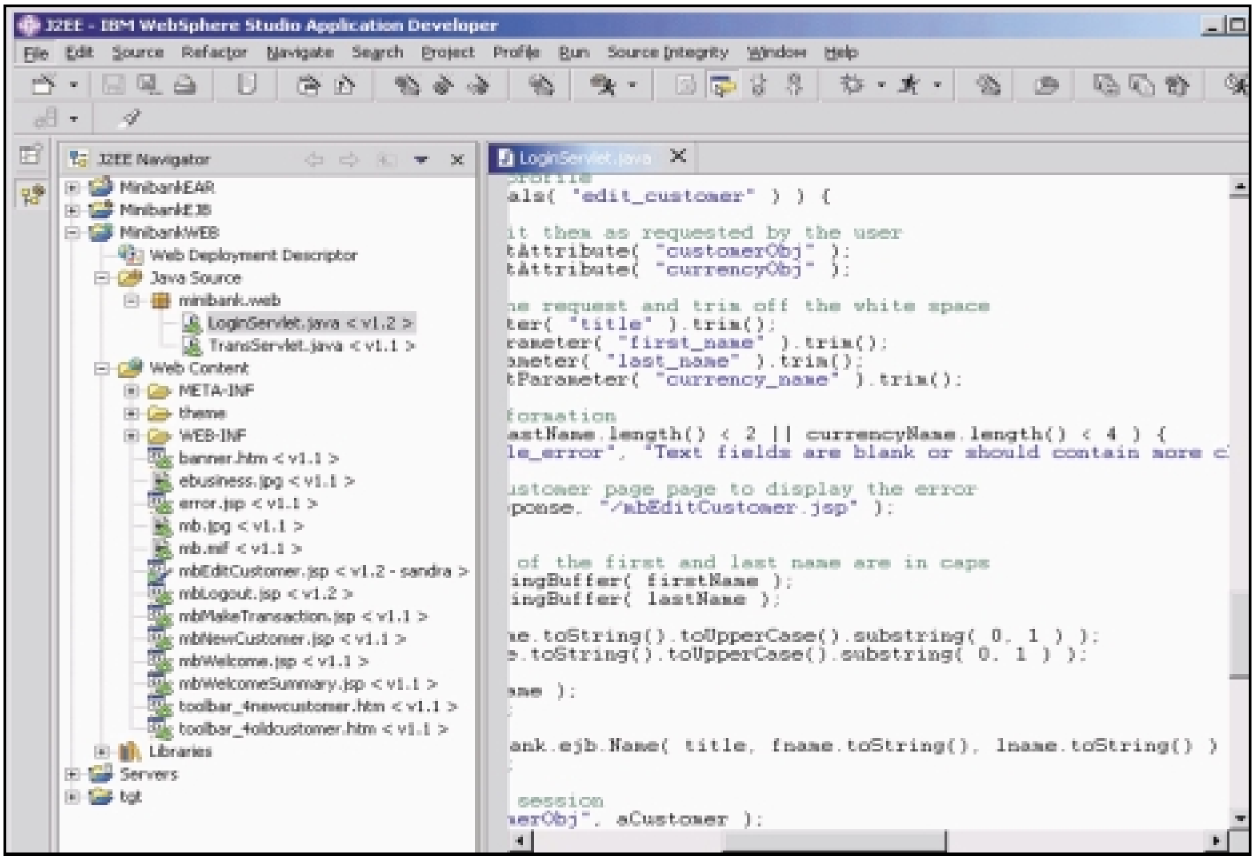


FIG. 1: GRAPHICAL MEMBER HISTORY SCREEN



J2EE - IBM WebSphere Studio Application Developer
File Edit Source Refactor Navigate Search Project Profile Run Source Integrity Window Help
J2EE Navigator
LoginServlet.java

```
MinibankEAR
MinibankEJB
MinibankWEB
  Web Deployment Descriptor
  Java Source
    minibank.web
      LoginServlet.java < v1.2 >
      TransServlet.java < v1.1 >
  Web Content
    META-INF
    theme
    WEB-INF
    banner.htm < v1.1 >
    ebusiness.jpg < v1.1 >
    error.jsp < v1.1 >
    mb.jpg < v1.1 >
    mb.wmf < v1.1 >
    mbEditCustomer.jsp < v1.2 - sandra >
    mbLogout.jsp < v1.2 >
    mbMakeTransaction.jsp < v1.1 >
    mbNewCustomer.jsp < v1.1 >
    mbWelcome.jsp < v1.1 >
    mbWelcomeSummary.jsp < v1.1 >
    toolbar_4newcustomer.htm < v1.1 >
    toolbar_4oldcustomer.htm < v1.1 >
  Libraries
Servers
tgt
```

```
profile
als( "edit_customer" ) ) {

it them as requested by the user
tAttribute( "customerObj" );
tAttribute( "currencyObj" );

he request and trim off the white space
ter( "title" ).trim();
rameter( "first_name" ).trim();
ameter( "last_name" ).trim();
tParameter( "currency_name" ).trim();

formation
astName.length() < 2 || currencyName.length() < 4 ) {
le_error", "Text fields are blank or should contain more c

ustomer page page to display the error
ponse, "/mbEditCustomer.jsp" );


 of the first and last name are in caps
ingBuffer( firstName );
ingBuffer( lastName );

me.toString().toUpperCase().substring( 0, 1 ) );
e.toString().toUpperCase().substring( 0, 1 ) );

ame );
;

ank.ejb.Name( title, fname.toString(), lname.toString() )
;

 session
merObj", aCustomer );
```

FIG. 2: WSAD SCREEN

Openmake from Catalyst Systems is an award-winning application build process automation tool. It provides the ability for any developer to do full or incremental builds reliably on any machine the same way every time, while eliminating the need for manual makefile or Ant XML scripting. Upon completion of a build, Openmake creates a Bill of Materials (BOM). This BOM includes information on every component that went into the EAR file.

The following scenario features a standalone J2EE application developed with WebSphere Studio Application Developer 5.0 and deployed to WebSphere Application Server 5.0 running on AIX 5.2

Version 1.0 of the application contains 1,200 Java source files and has gone through complete systems integration and user acceptance testing; it has been in production for a month. To ensure version 1.0 can always be re-created, it was checkpointed with Source Integrity Enterprise. It was built using Openmake and moved into production using the Integrity Manager's Build & Deploy module. After the application was built by Openmake, both the EAR file and the BOM were added to a Source Integrity Enterprise project for later use. Since that time, significant development has been done, working toward version 1.1. The work has resulted in 300 new files and changes to 50 existing files.

A serious bug has just been reported and must be fixed immediately. Investigation of the problem has been completed and development work assigned. Issues are created in Integrity Manager and assigned to three developers. It is clear from the Build & Deploy request that the production version (1.0) refers to Checkpoint 1.0. This information is provided to the developer in the issue. Each developer will work on a variant project, based on Checkpoint 1.0. In this way, none of the changes made since the release of 1.0 will be in their WebSphere project workspace.

An administrative option for this environment has been set that requires developers to add all work they do to a Change Package. This Change Package must be associated with an issue that has been assigned to them in Integrity Manger. As they work on their changes, the dynamically updating status indicators will clearly show the effect other developers may have. They can choose to bring in the other changes to do their unit testing. Once complete, they close their Change Package and mark their issue as complete. Once all developers have completed their unit testing and have closed their change packages, it's time to move to the next stage of the process.

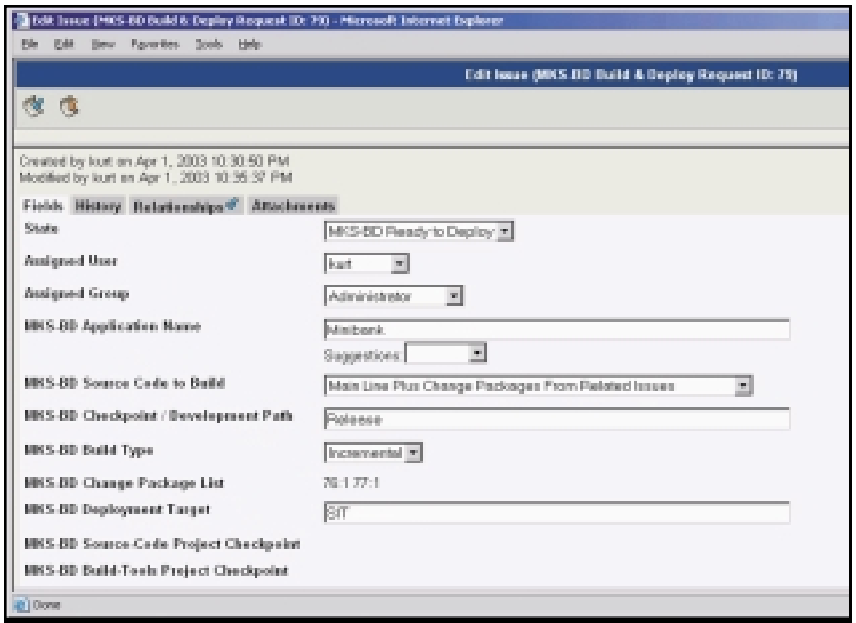
A Build & Deploy request will now be created in Integrity Manager to move the work through to production. The first action is to associate with the new Build & Deploy request all of the issues where work was attached. All of the changes will be automatically applied to Checkpoint 1.0. Once applied, the request will be moved to a ready-to-build state, which triggers Openmake to perform the build. The new EAR file and BOM will be added to a Source Integrity Enterprise project and a new checkpoint created.

In this view we see a Build & Deploy request. The State indicates the EAR file is ready to deploy and the "Source Code to Build" field shows the Change Packages from the related issues have been applied to the appropriate development path.

The Build & Deploy request will now call the WebSphere Application Server command line to install and deploy the EAR file on the designated systems integration test server. The QA technician will receive notification that work is assigned and ready for testing. Keep in mind there was a detailed BOM created by Openmake for both the production version of the EAR file and the current EAR file just deployed for testing. As part of the process, the technician can use Source Integrity Enterprise to differentiate between these two BOMs to be certain only the desired changes from the three developers were introduced. They can now QA-test those changes with confidence.

Upon successful testing, the EAR file can be deployed for another test phase or be promoted directly into production. This process is flexible and can be set up to meet the team's specific business needs.

Now that the issue is fixed in production, another important step is to ensure it is not re-introduced when version 1.1 goes live. The same Change Packages created to fix the problem in 1.0 can be seamlessly applied to the version 1.1 line of development to ensure this does not happen.


Edit Issue (MKS-BD Build & Deploy Request ID: 79) - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Edit Issue (MKS-BD Build & Deploy Request ID: 79)
Fields History Relationships Attachments
Done

Created by kurt on Apr 1, 2003 10:30:50 PM
Modified by kurt on Apr 1, 2003 10:36:37 PM

State	MKS-BD Ready to Deploy
Assigned User	kurt
Assigned Group	Administrator
MKS-BD Application Name	Minibank Suggestions:
MKS-BD Source Code to Build	Main Line Plus Change Packages From Related Issues
MKS-BD Checkpoint / Development Path	Release
MKS-BD Build Type	Incremental
MKS-BD Change Package List	76:1 77:1
MKS-BD Deployment Target	SIT
MKS-BD Source-Code Project Checkpoint	
MKS-BD Build-Tools Project Checkpoint	

FIG. 3: BUILD & DEPLOY REQUEST

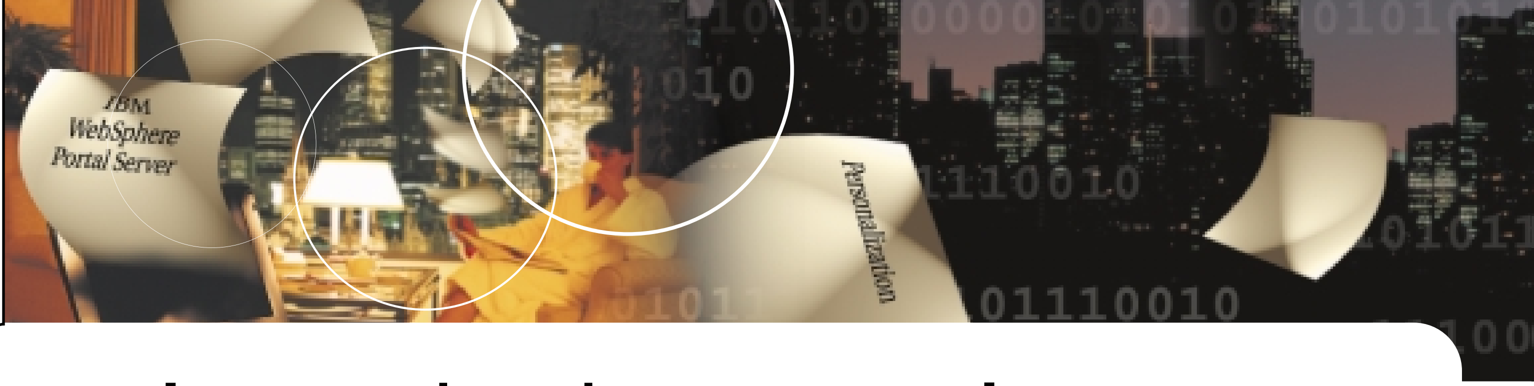
Conclusion

I have discussed how a tightly integrated SCM and change management system, combined with a mature build process, can address many of the complex issues facing today's enterprise Java development teams. There is a system that can provide the process and management control needed without inhibiting developers' creative talents.

In Part 2 of this series, Adam Terrenzio of MKS Inc. will discuss ways SCM plug-ins can help developers with day-to-day activities in his article, titled "Teamwork in WebSphere Studio."

VERSATA

WWW.VERSATA.COM/BUSINESSLOGICDESIGNER



Using WebSphere Portal to Manage an Order Tracking Process

Enhance your business and improve employee productivity

– BY CHRISTINA LAU AND COLIN YU –

More than ever, companies need to model and manage their business processes in a way that can integrate systems and people throughout the enterprise, as well as connect with customers and partners. The IBM WebSphere product family delivers those key capabilities to help companies respond with speed to any customer demand, market opportunity, or external threat.

As the foundation of the WebSphere software platform, WebSphere Application Server provides a rich e-business application deployment environment and offers full J2EE 1.3 specification support. The WebSphere Application Server Enterprise Process Choreographer can be used to choreograph all kinds of business processes. The types of business processes can vary greatly, ranging from Web services navigation to business transaction support. Business processes can be automatic, recoverable, or can require human interaction.

WebSphere Studio Application Developer Integration Edition provides J2EE developers and application integration specialists with an integrated development environment for building, testing, integrating, and deploying J2EE applications, business processes, and Web services. With this development tool, developers can model business processes from the Process Editor and test implementations in the WebSphere Test Environment using the embedded Application Server.

WebSphere Portal Server allows people to interact in a personalized way with the on-demand world. It acts as a simple, unified access point to Web applications, permits personalization of Web-based content, and makes it accessible to any device.

ABOUT THE AUTHOR
Christina Lau is a senior technical staff member at the IBM Toronto Lab. Christina is an architect on the On Demand Development team focusing on the next generation of WebSphere platform technologies and products.

E-MAIL
clau@ca.ibm.com

In this article we will discuss how to use these products to build, deploy, unit-test, and manage an order tracking process. First, we will create a business process using the Process Choreographer tool in WebSphere Studio Integration Edition, and show how to deploy and unit-test the process in the unit-test environment. Users can use the ready-to-use Process Choreographer Web client to start new processes or to claim and complete an activity in their work list. However, user interfaces for business process applications often require customization, such as adapting the user interface to fit a corporate look and feel, adding content, or introducing new functionality. Hence we will demonstrate how you can easily build a portlet that can integrate with your company portal as well as the order tracking process.

Creating a Part Order Tracking Process

A business process is a multistep operation. The part order workflow will implement the following activities:

1. Receive a part order request from a customer.
2. Check the credit card information for the customer.
3. If the customer has good credit, accept the order request and initiate shipping.
4. Otherwise, reject the order request.

To create a business process, we first need to create a service project using WebSphere Studio Integration Edition. In the Business Integration perspective, use the Service Project wizard to create a new service project called PartOrder. You can then create a business process using the Process Editor. A typical way to create a business process is to use a WSDL file that describes the business process interface. WebSphere Studio Integration Edition includes a WSDL editor that you can use to create or modify a WSDL file. Using the WSDL editor, create an order.wsdl file that contains the following processPartOrder one-way operation.

```
<portType name="partOrderPT">
    <operation name="processPartOrder">
```

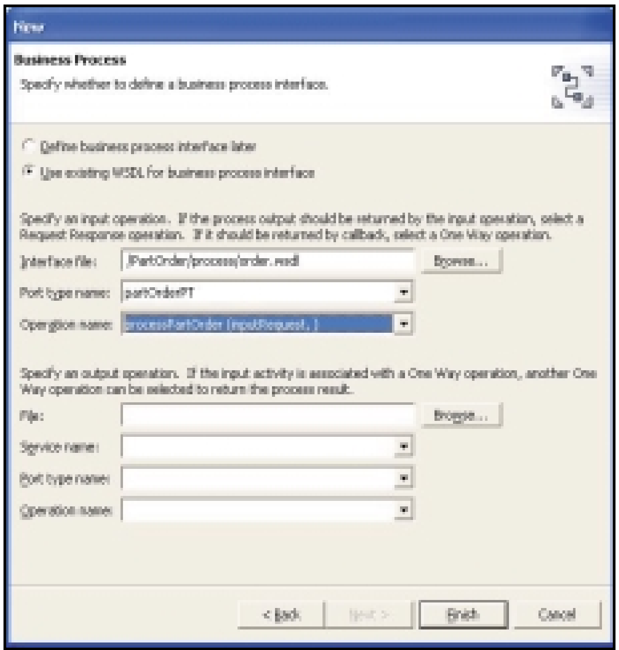
```
        <input name="inputRequest"
        message="tns:partOrder"/>
    </operation>
</portType>
```

The processPartOrder operation takes an input message partOrder that specifies the following input parameters:

```
<message name="partOrder">
    <part name="customerName" type="xsd:string"/>
    <part name="creditCardNumber" type="xsd:string"/>
    <part name="expiryDate" type="xsd:string"/>
    <part name="item" type="xsd:string"/>
    <part name="quantity" type="xsd:int"/>
</message>
```

With the order.wsdl file in hand, you can invoke the Business Process wizard to create the business process, as shown in Figure 1.

The part order business process has a number of activities. An activity represents an operation and can be implemented in many different ways such as an external Web service, a Java class, an EJB, or an EIS. To keep our business process simple, we will implement it using some custom Java code by dragging three Java snippets from the palette and dropping them onto the process. These three Java snippets are shown in Figure 2 as calculatePrice, initiateShipping, and rejectOrder.


New

Business Process

Specify whether to define a business process interface.

( ) Define business process interface later
(•) Use existing WSDL for business process interface

Specify an input operation. If the process output should be returned by the input operation, select a Request Response operation. If it should be returned by callback, select a One Way operation.

Interface file: /PartOrder/process/order.wsdl
Port type name: partOrderPT
Operation name: processPartOrder (inputRequest,)

Specify an output operation. If the input activity is associated with a One Way operation, another One Way operation can be selected to return the process result.

File:
Service name:
Port type name:
Operation name:

FIG. 1: CREATING A BUSINESS PROCESS FROM THE ORDER.WSDL FILE

Because the part order workflow requires human interaction to call the credit card company to check the customer credit, a Staff activity, checkCreditRating, is added to stop the execution of the process and to solicit human input before proceeding. If the customer has enough funds to pay for the part, the initiateShipping activity will be called to schedule the shipping; otherwise, the rejectOrder activity will be called to send an e-mail to notify the customer.

Unit Test the Business Process

After creating the business process and developing the implementation code, you can use the wizards to generate deployment code. WebSphere Studio Integration Edition provides a built-in unit-test environment that you can use to unit-test your business process. On the server instance on which you have deployed your business process, select Run Process Web client to start the Business Process Execution Web client. Using the Web client you can create a new instance of the business process. As shown in Figure 3, the default Web client prompts you for the input parameters to your business process. These input parameters are obtained from the partOrder message definition in the order.wsdl file.

When you click on the Start Process action, the part order business process will be created, the calculatePrice activity will be executed, and you can use the default screen to enter a "yes" to call the initiateShipping activity or a "no" to call the rejectOrder activity.

Although the default Web client is very useful for quickly testing your business process, most likely it will not be invoked in the same way the business process is invoked in a production environment. Custom applications can be developed to create the business process using one of the generated facades from WebSphere Studio Integration Edition. For example, if you want to launch the business process using a JMS message, you can generate a message-driven bean and send a message to start the business process. Alternatively, an application can also be developed directly using the Process Choreographer APIs. In the next section, we will show how to use this approach to develop customized portlets that can create, claim, and complete activities in the workflow.

Accessing the Business Process Using WebSphere Portal

WebSphere Portal allows people to interact and transact in a personalized way with diverse business resources. It provides a single point of personalized interaction with applications, content, and processes, enabling real-time collaboration. An online retailer can use WebSphere Portal to provide

ABOUT THE AUTHOR
Colin Yu is a technical designer on the WebSphere Business Scenarios Development team at the IBM Toronto Lab. Colin helps define integration requirements for WebSphere platform products through the development of scenarios.

E-MAIL
coliny@ca.ibm.com

"WebSphere Portal Server allows people to interact in a personalized way with the on-demand world. It acts as a simple, unified access point to Web applications, permits personalization of Web-based content, and makes it accessible to any device"

on-the-glass integration and to streamline business processes. For example, a customer can use a part order portlet such as the one shown in Figure 4 to place an order.

When the order is received from the customer, an instance of the business process that we described earlier is created. The first step in the business process is to calculate the final price for the order. This is handled by the calculatePrice activity that was shown in Figure 2.

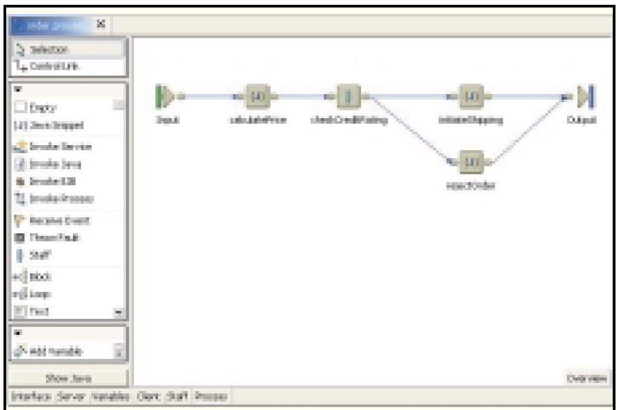
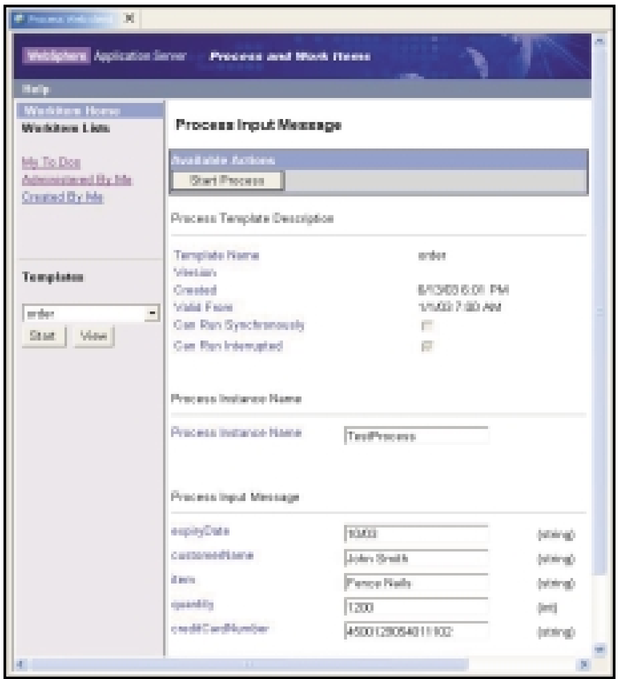


FIG. 2: PART ORDER BUSINESS PROCESS



Process Input Message

Process Template Description

Template Name	order
Version	
Created	8/13/03 6:01 PM
Valid From	1/1/03 7:00 AM
Can Run Synchronously	☐
Can Run Interrupted	☑

Process Instance Name

Process Instance Name: TestProcess

Process Input Message

expiryDate	10/03	(string)
customerName	John Smith	(string)
item	Fence Nails	(string)
quantity	1200	(int)
creditCardNumber	4500128054011102	(string)

FIG. 3: USING THE DEFAULT WEB CLIENT TO UNIT-TEST THE BUSINESS PROCESS

Once the final price is calculated, a clerk on the retailer side must call the credit card company to make sure the credit card information supplied by the customer is valid. The clerk can use an approval portlet such as the one shown in Figure 5, which shows the list of pending order requests.

The clerk can call the credit card company, validate the customer account, and then approve or reject the order. If the order is approved, the initiateShipping task will be called to send the order to the customer.

Integration Using Web Services

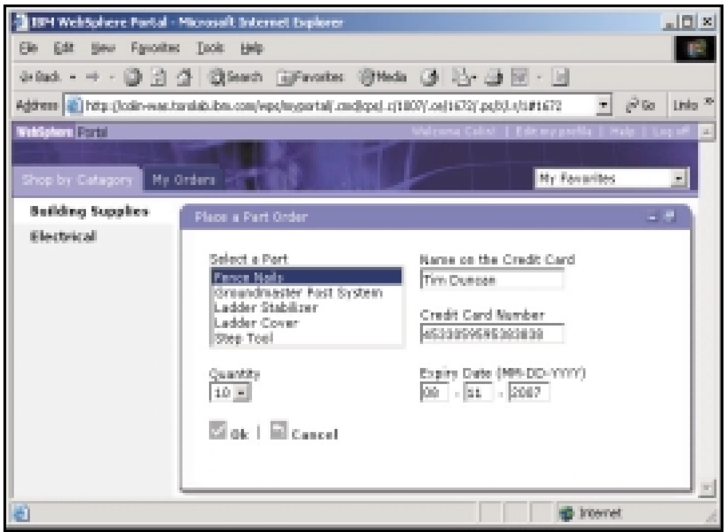
The Process Choreographer component provides a programming interface called the Business Process Service API for developing applications that can be used to control the processes and manage pending work items. The Business Process Service API is a set of methods available through a session bean facade of the Process Choreographer component.

The portlets that we developed run on WebSphere Portal Server 4.2. The business process that we developed runs on WAS Enterprise v5.0. We use Web services to enable seamless integration between the two systems. We developed a JavaBean WFServiceAdapter to wrap the Business Process Service API and use the Web Service wizard to deploy the bean as a Web service on the WAS Enterprise from WebSphere Studio. We also use WebSphere Studio to generate the proxy code used by the Part Order Portlet and the Approval Portlet to invoke the Web service, as shown in Figure 6.

In order to achieve a personalized list of work items in the Approval Portlet, user credentials are required to be passed from the WebSphere Portal Server to the WebSphere Application Server. We enable authentication for the RPC Router on the WebSphere Application Server and pass the username and password to the HTTP transport layer from the portlets. To guarantee the confidentiality of the SOAP message, SOAP over HTTPS is used to send the message from the portal server to the application server.

Conclusion

The WebSphere software platform is organized into three areas of functionality often shown as a pyramid: foundation and tools for building, running, and deploying applications; business integration for integrating internal business processes; and business portals for providing a single point of personalized interaction with applications, content, processes, and people. We have shown you how to use these capabilities together to enhance your business and improve employee productivity.


IBM WebSphere Portal - Microsoft Internet Explorer

Address: http://colin-was.torolab.ibm.com/wps/myportal/.cmd/ps/.c/1807/.ce/1672/.p/5/.s/1#1672

Place a Part Order

Select a Part: Fence Nails; Groundmaster Post System; Ladder Stabilizer; Ladder Cover; Step Tool

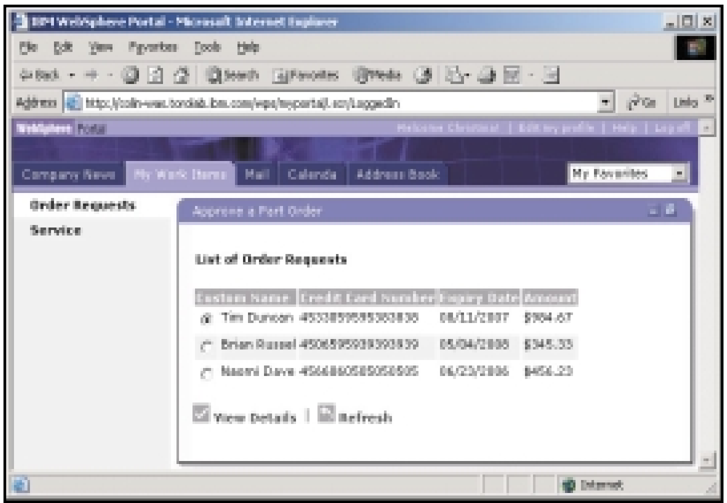
Name on the Credit Card: Tim Duncan

Credit Card Number: 4533099595383839

Quantity: 10

Expiry Date (MM-DD-YYYY): 08 - 11 - 2007

FIG. 4: A PART ORDER PORTLET


IBM WebSphere Portal - Microsoft Internet Explorer

Address: http://colin-was.torolab.ibm.com/wps/myportal/.scr/Loggedin

Approve a Part Order

List of Order Requests

Custom Name	Credit Card Number	Expiry Date	Amount
Tim Duncan	4533859595383838	08/11/2007	$904.67
Brian Russel	4506595939393939	05/04/2008	$345.33
Naomi Dave	4564860585050505	06/23/2006	$456.23

FIG. 5: AN APPROVAL PORTLET

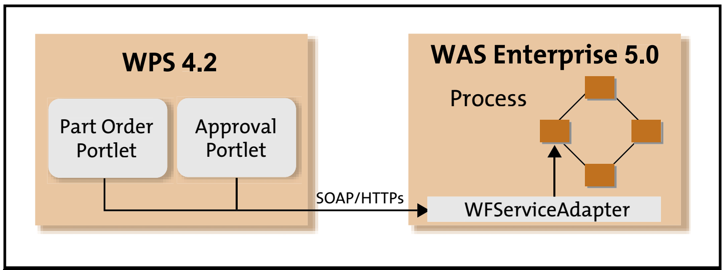


FIG. 6: TOPOLOGY

References

- *WebSphere Application Server Enterprise Edition:* www-3.ibm.com/software/webservers/appserv/enterprise
- *WebSphere Studio Application Developer Integration Edition:* www-3.ibm.com/software/awdtools/studiointegration
- *WebSphere Application Server Enterprise Process Choreographer Concepts and Architecture:* www7b.boulder.ibm.com/wsdd/library/techarticles/wasid/WPC_Concepts/WPC_Concepts.html
- *WebSphere Portal Server:* www-3.ibm.com/software/genservers/portal
- *Business Process Service API:* http://publib7b.boulder. ibm.com/wasinfo1/en/info/ee/javadoc/ee/com/ibm/bpe/api/BusinessProcessService.html

MQSOFTWARE

WWW.MQSOFTWARE.COM/QNAMI

Combines modeling technology with a component generator

IBM Rational Rapid Developer

BY JAY JOHNSON

As a long-time Rational Rose user, I'm happy with what I've seen lately. IBM is making its acquisition of Rational pay off with the creation of what seems to be the ultimate end-to-end modeler/code generator. Rational Rapid Developer (RRD) combines the features of Rational Rose UML modeler and the latest in component generators. This adds a UML modeling dimension to the RAD approach many vendors are addressing with code generators alone. From the Rational side of its heritage, the new product includes support for deployment to an impressive variety of application servers: Rapid Developer supports J2EE application servers as well as offering deprecated support for Microsoft DNA.

ABOUT THE AUTHOR
Jay Johnson is a J2EE architect with Tier Technologies, which combines deep domain expertise with technical capabilities to solve problems for their clients in government, health care, utilities, and insurance. In his spare time Jay works as the product review editor for *WebSphere Developer's Journal.*

E-MAIL
jay2ee@hotmail.com

For Java environments, the Rational platform supports all J2EE application servers, providing automated deployment to most leading application servers, including IBM WebSphere. Rapid Developer–generated applications will manually deploy to all other J2EE application servers.

Fans of other Rational tools, such as the Rose Modeler, will notice that RRD doesn't have quite the same look and feel, and that occasionally the documentation refers to "NeuArchitect." This is because much of RRD is based on a tool Rational purchased, then reworked extensively in-house. This is definitely not a version 1.0 product. It is loaded with more wizards than Hogwarts School of Magic, including generators for connectivity, transaction modeling, persistence, Web apps, and JSPs, in addition to UML capabilities. Similar in concept to WSAD/Eclipse, RRD provides many perspectives of an application, such as database/persistence definition (see Figure 1) and class diagrams (see Figure 2).

Although the RRD model builder is different from Rational Rose and Rational XDE, it can import model files from these tools. According to IBM, developers can import existing UML class models into Rapid Developer as business objects using XML Metadata Interchange (XMI). Rapid Developer also provides import and iterative synchronization with Rational Rose and Rational XDE class models, leveraging Rational's Extensibility APIs. IBM Rational has also delivered an IBM Rational Unified Process (or RUP) plug-in for Rational Rapid Developer that makes it easier to configure the RUP for use with the tool.

Rational Rapid Developer has the ability to iteratively import and synchronize class models from IBM Rational Rose and IBM Rational XDE using the Rational Extensibility API. (Rational Rapid Developer can also use XMI to import class models from Rational Rose and Rational XDE, and other IBM and non-IBM products that support an XMI export.) The synchronization utility provides the ability to capture changes to the original models on an iterative basis, along with the ability to select and deselect model elements for import and synchronization.

The Rational platform supports popular browsers, including versions of Microsoft Internet Explorer and Netscape Navigator. The HTML template languages supported include Microsoft's Active Server Pages and JavaServer Pages. It also supports Java applets, Shockwave, and Flash.

Rapid Developer supports J2EE and allows users to automatically generate Enterprise JavaBeans as part of the constructed application. Rapid Developer also supports the leading EJB standard application servers, including IBM WebSphere.

Rational Rapid Developer supports multiuser development with integrated support for IBM Rational ClearCase as well as other commercial version control systems that support the SCC API. The interface to Rational ClearCase is integrated into Rational Rapid Developer so developers use Rational Rapid Developer controls and dialogs while leveraging Rational ClearCase under the hood.

Java IDE and Debugging Facilities

Rational Rapid Developer provides a plug-in for Eclipse that enables it to link with Eclipse-based IDEs such as WebSphere Studio. Rational Rapid Developer can be launched from WebSphere Studio, can automatically create project files in WebSphere Studio, and can use WebSphere Studio for detailed (line-by-line) debugging of Rational Rapid Developer–generated applications.

Where other RAD tools support only a programming view of a Web container, Rapid Developer provides support for senior Java developers and architects as well. From the IBM side of its heritage, RRD offers wizards to generate legacy-interface code. The Rational platform directly supports IMS, VSAM (CICS and MVS applications) and, via partner adapters, access to a broad set of other data sources. RRD also natively supports IBM DB2, Microsoft SQL Server, Access, Oracle, and Sybase. In addition, RRD offers generic driver support for other SQL 92 data sources via JDBC.

In addition to the UML models Rational Rose supports, the Rapid Developer Modeling System also includes models for database, process, security, and rules; user interface models for theme, style, page, site interaction (linking business objects to data); and integration models for external (B2B) and internal (EAI) systems using database, XML, API, and Web services.

The combination of a modeling tool and an application generator is a powerful one, and no other tool has yet gone to the lengths of RRD to provide not only class generators but a WYSIWYG JSP builder. Going beyond creating skeleton J2EE components, Rapid Developer creates the set of supporting classes and configuration files needed for generation, optimization, and deployment of an application.

ObjectSpace, a transaction modeling tool included with RRD, has especially interesting potential. Modeling transactions is a particularly challenging area of application development, and a graphical builder can provide real advantages. Transaction modeling has become even more challenging due to the increased complexity of *n*-tier architectures and the multitude of delivery channels, including Web (HTML), wireless (WML), messaging (XML), and Web services (SOAP). ObjectSpace uses a visual environment for transaction modeling and ties the models created directly to the business objects and databases on one side, and to various presentation formats on the other.

Once applications are generated, modifications are made at the model level, not to the code. This dramatically reduces the maintenance effort because the maintenance of models is significantly easier than the maintenance of code. The code and model stay in sync. After the changes have been made to the models, Rapid Developer regenerates the application and the old code is overwritten.

Unlike Rational XDE and Rational Rose Modeler, Rapid Developer cannot reverse-engineer existing Java code and create a model from it. However, it is possible to reuse existing Java code in a Rational Rapid Developer application, expose the code as services or APIs, and package them in a JAR file that you include in Rational Rapid Developer. Your application can then call these services from your business logic methods.

Rapid Developer allows the developer to preview automatically generated code. If the developer wants to modify, delete, or change any of this code, he or she can make modi-

MQSOFTWARE

WWW.MQSOFTWARE.COM/QNAMI

fications to the code within the tool. The code can then be generated with the desired modifications. These modifications are made persistent for subsequent regenerations of the application.

Rational does not recommend that developers make changes to the code outside of Rapid Developer, as modifications to the generated code will require future maintenance of the code rather than the models. This makes sense from a process point of view, but it also means that all future maintenance must be done using RRD, locking a project into a single vendor. This is certainly not a unique strategy for Rational/IBM. On the other hand, Rational says the tool leaves it open for developers to integrate code from other sources, but this code will not be included in the RRD model.

Rapid Developer offers a wide range of functionality for developing many different types of applications. Your applications may provide services that can be used only by other applications, or you can create a complete end-to-end business application. It opens the J2EE development paradigm up to a variety of application developers, since only limited knowledge of Java is required to create most applications.

However, greater automation brings less flexibility. Developers who want to dynamically model a system would do to well to use Rational XDE in conjunction with WSAD, which provides a much greater variety of UML diagrams. Rational XDE can be used for full UML modeling, whereas Rational Rapid Developer uses UML much more judiciously (primarily class diagrams). The good news is the products can coexist – syncing class models from Rational XDE to RRD, as well as publishing and consuming components.

In its next release, scheduled for mid-October, Rational Rapid Developer will provide automated construction of portlets that run on IBM WebSphere Portal Server. This support will be seamlessly integrated into the existing Rational Rapid Developer interface. In addition, the next release will include support for J2EE Connector Architecture.

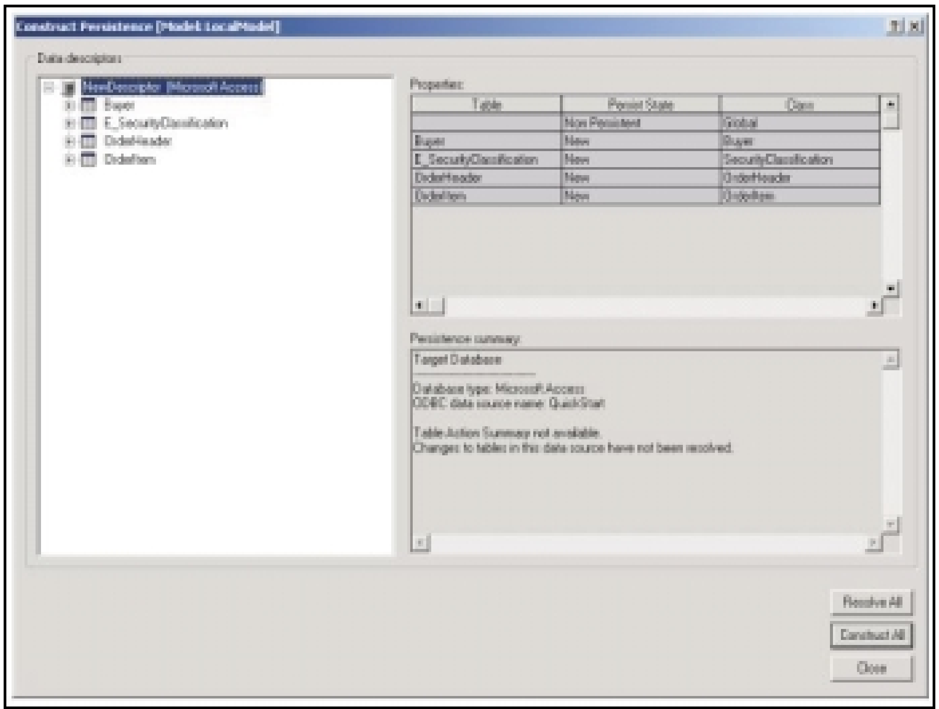


Table	Persist State	Class
	Non Persistent	Global
Buyer	New	Buyer
E_SecurityClassification	New	SecurityClassification
OrderHeader	New	OrderHeader
OrderItem	New	OrderItem

Target Database

Database type: Microsoft Access
ODBC data source name: QuickStart

Table Action Summary not available.
Changes to tables in this data source have not been resolved.

FIG. 1: CONSTRUCT PERSISTENCE VIEW

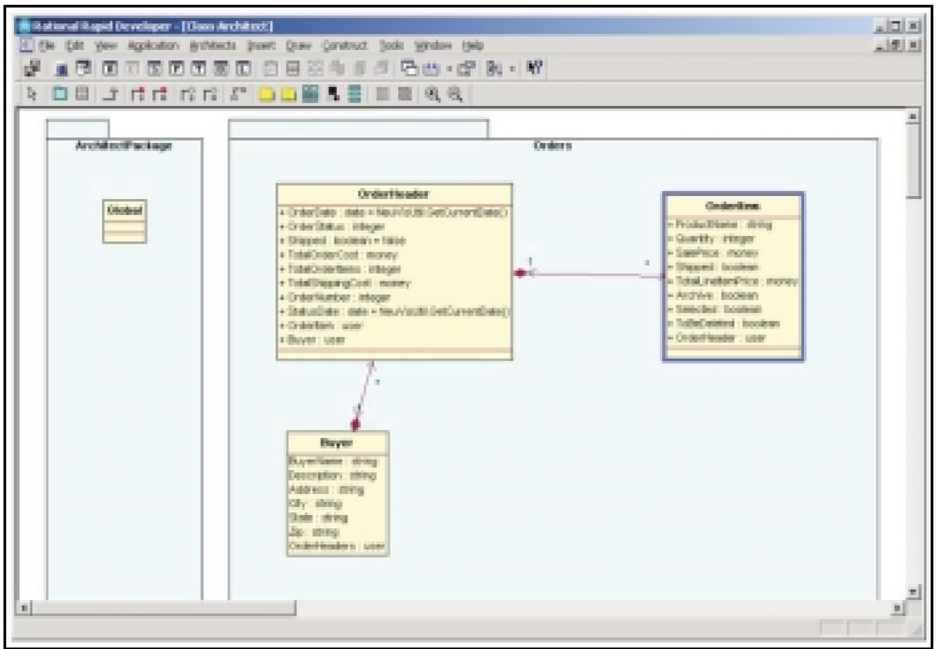


FIG. 2: CLASS ARCHITECT VIEW

IBM Rational Rapid Developer

COMPANY INFO:
Rational Software from IBM
18880 Homestead Rd.
Cupertino, CA 95014
Telephone: 800.728.1212
Web: www.rational.com
E-mail: www.rational.com/contact/request.jsp

SPECIFICATIONS:
Pentium III-based microprocessor (minimum 300MHz), 256MB of RAM (768MB for IBM WebSphere), 500MB disk space (minimum), a high-resolution display (minimum 1024 x 768) Windows NT 4.0 (Workstation or Server), Service Pack 5 (Admin rights on your NT system) *Note:* Internationalization features are not supported under Windows NT 4; Windows 2000, Admin rights on your NT system (Service Pack 1, 2, or 3); or Windows XP

MQSOFTWARE

WWW.MQSOFTWARE.COM/QNAMI

JMX can give you a view into your components' performance

Understanding and Optimizing Java Management Extensions

BY GIRISH KULKARNI AND BARRY NANCE

Developers are capitalizing on Java's open and dynamic properties to use the technology for seemingly limitless applications across the computing spectrum. To ensure that developers and businesses optimize Java performance in a variety of deployments, organizations must use an organized, standardized approach to looking inside – and sometimes even modifying – Java-based devices or processes.

ABOUT THE AUTHOR
Girish Kulkarni is a senior R&D manager at Candle Corporation. He has been involved in the design and development of several of Candle's middleware solutions. He is working with a team on the design of a comprehensive J2EE monitoring and management solution.

E-MAIL
girish_kulkarni@candle.com

Java Management Extensions (JMX) is a relatively new Java standard that provides developers with a standard template and process for viewing the performance of Java components.

Originally part of the Java 2 Platform, Enterprise Edition (J2EE) 2.0 standards maintained by the Java Community Process (www.jcp.org), JMX grew via the Java Service Request (JSR-3) standard into a separate specification in its own right: JMX 1.0. Several Java-based development environments and runtime components contain support for JMX. For instance, IBM's WebSphere Application Server version 5.0 implements JMX.

Using the JMX specification as a guide, Java programmers can more easily integrate their software with management protocols, platforms, and tools. More broadly, the specification describes the JMX architecture, provides template software on which Java programmers can model their code to become JMX-compliant, spells out the JMX application programming interfaces (APIs), and explains the application and network management services that JMX-enabled software can use.

JMX is particularly well suited to mission-critical business process software. For a Java application that automates a business function – as opposed to some other type of Java-based entity, such as a network router or mobile phone – you might think of JMX as an interface between the application and a management/monitoring software tool.

Architecture Overview

The JMX specification developed by the Java Community Process defines a framework, a set of APIs, and a collection of services for managing JMX-enabled devices and applications. Even more formally, JMX's architecture contains an Instrumentation Level, an Agent Level, and a Distributed Services Level.

The JMX Instrumentation Level, or application level, ties a Java-based computing resource to the JMX framework. Within the application, managed bean (also called MBean) components expose application-specific data for management and monitoring purposes. Software designers and programmers specify which information the MBeans make available to the outside world. MBeans, which are similar to JavaBeans, can reveal internal software structures, disclose operational statistics, announce the progress of various internal application functions, and even accept control input that can modify the application's behavior. For instance, through a dashboard entry you might be able to pause, shut down, or temporarily disable some internal functions within an application. Alternatively, if you deem it necessary, you might be able to throttle the application's forward progress.

The Agent Level describes how to instrument a Java resource with the sensors mentioned earlier and link the result to a management/monitoring tool. The Distributed Services Level, which is the most recent section of the JMX specification and the one perhaps most subject to change, discusses JMX in the context of existing network management standards. This article will focus on the JMX Instrumentation Level.

JMX Coding 101

The following examples illustrate interfacing an application with JMX. The examples underscore the simplicity of the JMX APIs. You can judge from these two simple examples how application programmers incorporate JMX into their software.

```
import javax.management.*;
import com.sun.jdmk.comm.*;
public interface ServiceMBean {
Public void serviceMethod();
}
public class Service implements
ServiceMBean {
public Service() {}
public void serviceMethod() {}
}
```

International Conference & Expo

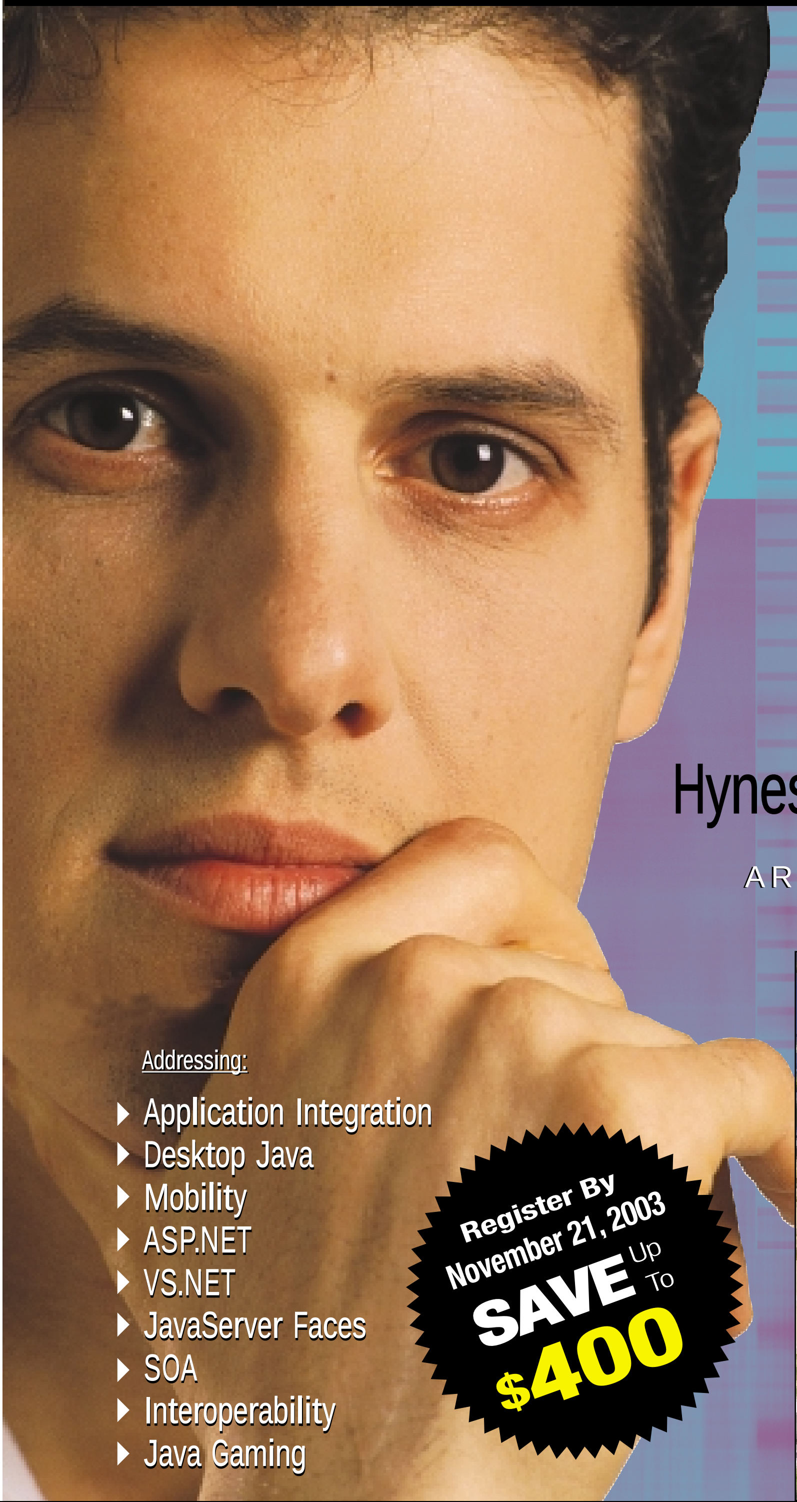
Edge 2004 EAST

Development Technologies Exchange

February 24-26, 2004

Hynes Convention Center, Boston, MA

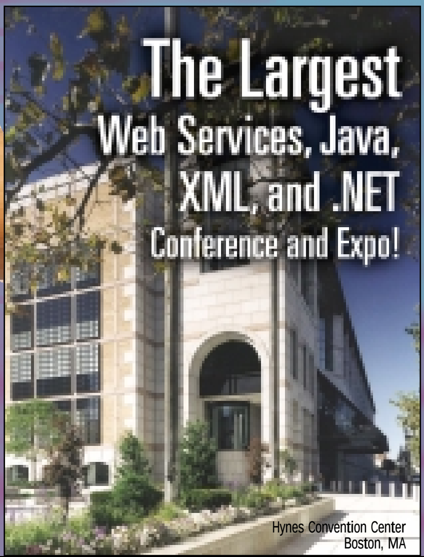
ARCHITECTING JAVA, .NET, WEB SERVICES, OPEN SOURCE, AND XML



Addressing:

- Application Integration
- Desktop Java
- Mobility
- ASP.NET
- VS.NET
- JavaServer Faces
- SOA
- Interoperability
- Java Gaming

Register By November 21, 2003 SAVE Up To $400



For more information visit
www.sys-con.com
or call
201 802-3069

Over 200 participating companies will display and demonstrate over 500 developer products and solutions.

Over 3,000 Systems Integrators, System Architects, Developers, and Project Managers will attend the conference expo.

Over 100 of the latest sessions on training, certifications, seminars, case-studies, and panel discussions promise to deliver real world benefits, the industry pulse and proven strategies.

Full Day Tutorials Targeting
- Java • .NET • XML
- Web Services • Open Source

Conference program available online!
www.sys-con.com/edgeeast2004

Contact information: 201 802-3069 • events@sys-con.com

This code snippet shows a few lines of Java programming language statements a programmer could use to JMX-enable an application. To the Service Class shown in the example, the programmer would add whatever Java statements are required to enumerate those data items he or she wants to expose via JMX. The following code snippet depicts Java programming statements for creating and registering JMX objects.

```
MBeanServer MBS =
MBeanServerFactory.createMBeanServe
r("AppAgent");
MBS.registerMBean(Object,
ObjectName);
```

Instrumentation Level Details

A Java-based device, such as a network router or switch, can use JMX to "instrument" itself with sensor-like behaviors. Via JMX, such a device could, for example, respond to Simple Network Management Protocol (SNMP) requests. SNMP is a pervasive protocol that network operations people use to monitor network devices. A business application can also use JMX to instrument itself for network management and monitoring. It can also offer control interfaces through which administrators can alter its behavior.

A majority of Java runtime environments contain or will soon include JMX implementations you can leverage. For example, to provide core control and basic management services, J2EE application server vendors – such as IBM in its WebSphere Application Server – are enthusiastically incorporating JMX in their products. JMX is becoming an essential building block within Java application servers, and JMX support will be mandatory in J2EE 1.4.

Developers and programmers JMX-enable a business application by inserting some specific programming statements in their Java programs, as shown in the Architecture Overview section. Alternatively, programmers can dynamically inject JMX code into their applications in a declarative fashion. A separate monitoring tool works with the Java runtime system to monitor the JMX-enabled application.

MBeans essentially turn the application into a managed Java entity. As noted, MBeans are similar to session beans and entity beans (i.e., Enterprise JavaBeans), which a WebSphere application already contains.

These statements are relatively harmless from a performance or development standpoint. The MBean programming statements identify and enumerate the application data structures and data variables the application's designers and programmers want to expose. A monitoring tool interfaces with JMX to observe the application and reveal the exposed application data. When a JMX-enabled application runs in the presence of a JMX-based monitoring tool, the tool can show capacity planners, administrators, troubleshooters, and developers application characteristics and even let them modify its behavior.

JMX Costs and Benefits

Because WebSphere already supports JMX, the incremental development cost of enabling JMX is nominal. The cost of becoming familiar with JMX concepts and techniques is far from exorbitant. Java programmers with intermediate skills typically find that a day or two of JMX familiarization makes them quite proficient.

Administering the JMX aspects of an application requires only a part-time effort. During development, programmers or a team's librarian might perform the few administrative chores, while later in production a network administrator might assume those duties. The monitoring tool you select should be robust, reliable, JMX-compliant, scalable, responsive, and flexible. It will become your company's window into the behavior and performance of the application. The cost of the tool will in all cases be far less than the cost of the ad hoc, do-it-yourself approach that many companies had to take in the past.

JMX alternatives lack the organization, standardization, and simplicity of JMX. For instance, developers who cobble together after-the-fact debugging code to display an application's intermediate results via the application's user interface are notorious for leaving their code in the application. Programmer oversight is so notorious that, in many cases, organizations simply overlook and ignore the extraneous displays. In rare cases, organizations may grow so tired of asking the developers to remove the extraneous displays that users may insert notes in the application documentation instructing new hires on which display output to ignore or automatically "click through."

The cost of using troubleshooting experts to analyze and solve an application performance problem can be exorbitant. The cost of the experts typically grows as efforts to solve a problem fail time and time again. Such costs are never budgeted at the outset of an application development project, but rather occur as a nasty surprise late in the development cycle or soon after the application is delivered to the business community.

Conclusion

JMX provides a means of controlling costs, ensuring application reliability, and preserving application integrity while still providing developers, troubleshooters, and IT managers with the information they require to solve performance and availability problems. The JMX-related costs for development resources, administration, and a good-quality, third-party monitoring and management tool pale in comparison to JMX's benefits and the overall expenses you incur in the development, operation, and administration of a significant, mission-critical business automation application.

By embracing JMX, developers, troubleshooters, and IT managers can access an application's internal structure and observe its status at any given point in time. Examining an application's current operations provides developers and IT managers with the information required to determine whether the application is behaving correctly and whether it is consuming excessive computing resources.

Software management and monitoring tools are a natural addition to the JMX architecture. In fact, you might consider a WebSphere application without an accompanying JMX monitoring tool to be like a car without a dashboard instrument panel.

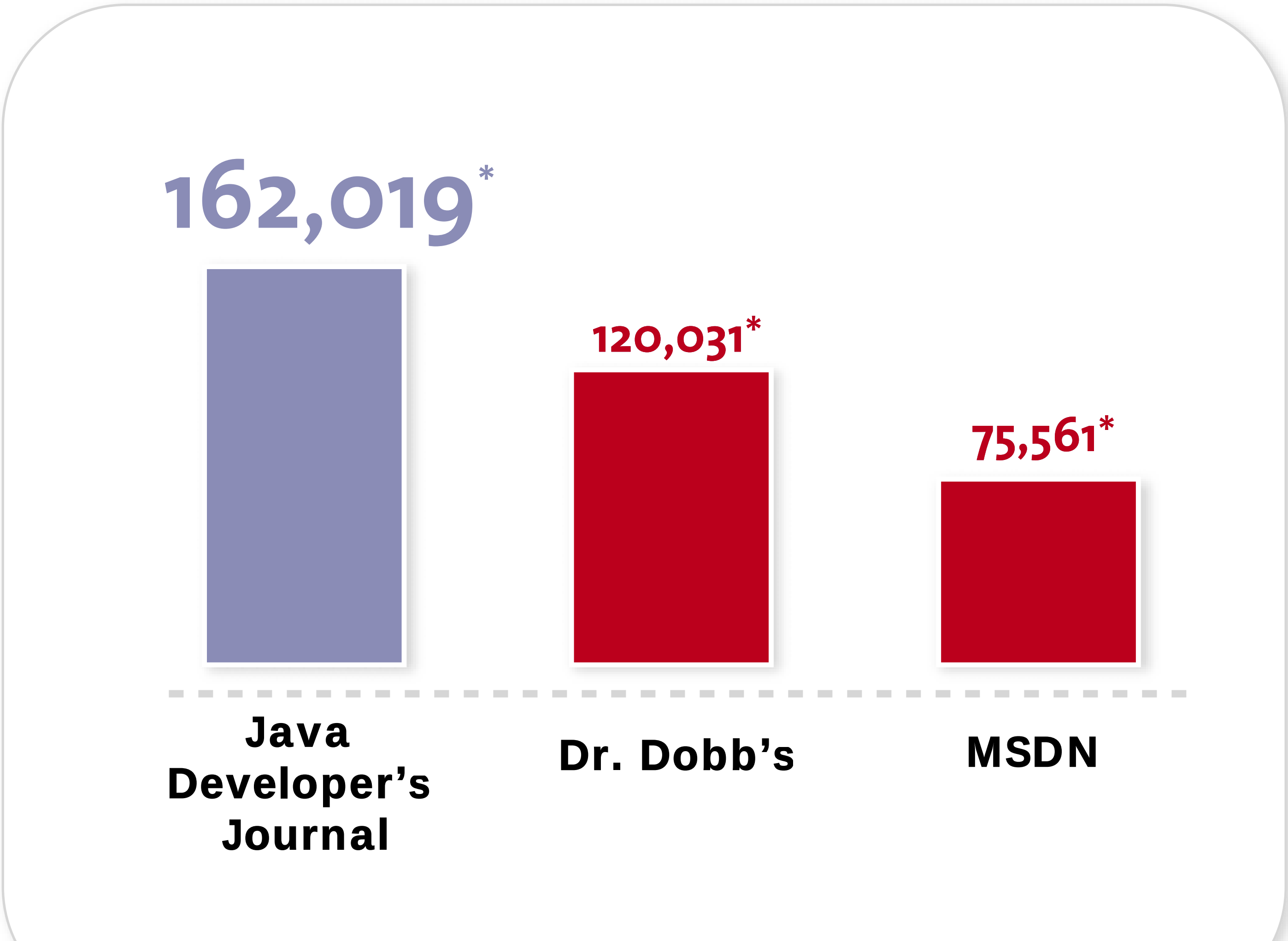
ABOUT THE AUTHOR

Barry Nance is a consultant for Candle Corporation. During his 29 years in the IT industry, he has designed Web-based e-commerce applications and developed a number of network diagnostic software utilities and special-purpose networking protocols. He has also written frequently for a number of technology publications and has authored three technical books.

E-MAIL
barryn@erols.com

#1 Circulation in the World!

JDJ is the highest circulation i-Technology magazine in the world!



*JUNE 2003 BPA AUDIT STATEMENTS *All circulation numbers are publishers' most current own data, six-month average circulation through June 2003.

SYS-CON MEDIA

Miles Silverman
VP Marketing

Carmen Gonzalez
Senior VP Marketing

Contact | Carmen Gonzalez: carmen@sys-con.com or 201 802.3021

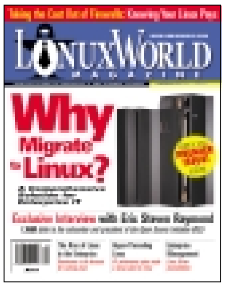
JAVA DEVELOPER'S JOURNAL

OFFER FROM SYS-CON Media

PLE MAGAZINES ONLINE

AND SAVE UP TO $400 AND RECEIVE UP TO 3 FREE CDs!

RECEIVE YOUR DIGITAL EDITION ACCESS CODE INSTANTLY WITH YOUR PAID SUBSCRIPTIONS



TO ORDER •Choose the Multi-Pack you want to order by checking next to it below. •Check the number of years you want to order. •Indicate your location by checking either U.S., Canada/Mexico or International. •Then choose which magazines you want to include with your Multi-Pack order.

Pick a 3-Pack, a 6-Pack or a 9-Pack

☐ 3-Pack	☐ 1YR	☐ 2YR	☐ U.S.	☐ Can/Mex	☐ Intl.
☐ 6-Pack	☐ 1YR	☐ 2YR	☐ U.S.	☐ Can/Mex	☐ Intl.
☐ 9-Pack	☐ 1YR	☐ 2YR	☐ U.S.	☐ Can/Mex	☐ Intl.

☐ **Linux World Magazine**

U.S. - Two Years (24) Cover: $143	You Pay: $79.99 /	Save: $63 + FREE $198 CD
U.S. - One Year (12) Cover: $72	You Pay: $39.99 /	Save: $32
Can/Mex - Two Years (24) $168	You Pay: $119.99 /	Save: $48 + FREE $198 CD
Can/Mex - One Year (12) $84	You Pay: $79.99 /	Save: $4
Int'l - Two Years (24) $216	You Pay: $176 /	Save: $40 + FREE $198 CD
Int'l - One Year (12) $108	You Pay: $99.99 /	Save: $8

☐ **Java Developer's Journal**

U.S. - Two Years (24) Cover: $144	You Pay: $89 /	Save: $55 + FREE $198 CD
U.S. - One Year (12) Cover: $72	You Pay: $49.99 /	Save: $22
Can/Mex - Two Years (24) $168	You Pay: $119.99 /	Save: $48 + FREE $198 CD
Can/Mex - One Year (12) $84	You Pay: $79.99 /	Save: $4
Int'l - Two Years (24) $216	You Pay: $176 /	Save: $40 + FREE $198 CD
Int'l - One Year (12) $108	You Pay: $99.99 /	Save: $8

☐ **Web Services Journal**

U.S.- Two Years (24) Cover: $168	You Pay: $99.99 /	Save: $68 + FREE $198 CD
U.S. - One Year (12) Cover: $84	You Pay: $69.99 /	Save: $14
Can/Mex - Two Years (24) $192	You Pay: $129 /	Save: $63 + FREE $198 CD
Can/Mex - One Year (12) $96	You Pay: $89.99 /	Save: $6
Int'l - Two Years (24) $216	You Pay: $170 /	Save: $46 + FREE $198 CD
Int'l - One Year (12) $108	You Pay: $99.99 /	Save: $8

☐ **.NET Developer's Journal**

U.S. - Two Years (24) Cover: $168	You Pay: $99.99 /	Save: $68 + FREE $198 CD
U.S. - One Year (12) Cover: $84	You Pay: $69.99 /	Save: $14
Can/Mex - Two Years (24) $192	You Pay: $129 /	Save: $63 + FREE $198 CD
Can/Mex - One Year (12) $96	You Pay: $89.99 /	Save: $6
Int'l - Two Years (24) $216	You Pay: $170 /	Save: $46 + FREE $198 CD
Int'l - One Year (12) $108	You Pay: $99.99 /	Save: $8

☐ **XML-Journal**

U.S. - Two Years (24) Cover: $168	You Pay: $99.99 /	Save: $68 + FREE $198 CD
U.S. - One Year (12) Cover: $84	You Pay: $69.99 /	Save: $14
Can/Mex - Two Years (24) $192	You Pay: $129 /	Save: $63 + FREE $198 CD
Can/Mex - One Year (12) $96	You Pay: $89.99 /	Save: $6
Int'l - Two Years (24) $216	You Pay: $170 /	Save: $46 + FREE $198 CD
Int'l - One Year (12) $108	You Pay: $99.99 /	Save: $8

☐ **WebLogic Developer's Journal**

U.S. - Two Years (24) Cover: $360	You Pay: $169.99 /	Save: $190 + FREE $198 CD
U.S. - One Year (12) Cover: $180	You Pay: $149 /	Save: $31
Can/Mex - Two Years (24) $360	You Pay: $179.99 /	Save: $180 + FREE $198 CD
Can/Mex - One Year (12) $180	You Pay: $169 /	Save: $11
Int'l - Two Years (24) $360	You Pay: $189.99 /	Save: $170 + FREE $198 CD
Int'l - One Year (12) $180	You Pay: $179 /	Save: $1

☐ **ColdFusion Developer's Journal**

U.S. - Two Years (24) Cover: $216	You Pay: $129 /	Save: $87 + FREE $198 CD
U.S. - One Year (12) Cover: $108	You Pay: $89.99 /	Save: $18
Can/Mex - Two Years (24) $240	You Pay: $159.99 /	Save: $80 + FREE $198 CD
Can/Mex - One Year (12) $120	You Pay: $99.99 /	Save: $20
Int'l - Two Years (24) $264	You Pay: $189 /	Save: $75 + FREE $198 CD
Int'l - One Year (12) $132	You Pay: $129.99 /	Save: $2

☐ **Wireless Business & Technology**

U.S. - Two Years (24) Cover: $144	You Pay: $89 /	Save: $55 + FREE $198 CD
U.S. - One Year (12) Cover: $72	You Pay: $49.99 /	Save: $22
Can/Mex - Two Years (24) $192	You Pay: $139 /	Save: $53 + FREE $198 CD
Can/Mex - One Year (12) $96	You Pay: $79.99 /	Save: $16
Int'l - Two Years (24) $216	You Pay: $170 /	Save: $46 + FREE $198 CD
Int'l - One Year (12) $108	You Pay: $99.99 /	Save: $8

☐ **WebSphere Developer's Journal**

U.S. - Two Years (24) Cover: $360	You Pay: $169.99 /	Save: $190 + FREE $198 CD
U.S. - One Year (12) Cover: $180	You Pay: $149 /	Save: $31
Can/Mex - Two Years (24) $360	You Pay: $179.99 /	Save: $180 + FREE $198 CD
Can/Mex - One Year (12) $180	You Pay: $169 /	Save: $11
Int'l - Two Years (24) $360	You Pay: $189.99 /	Save: $170 + FREE $198 CD
Int'l - One Year (12) $180	You Pay: $179 /	Save: $1

☐ **PowerBuilder Developer's Journal**

U.S. - Two Years (24) Cover: $360	You Pay: $169.99 /	Save: $190 + FREE $198 CD
U.S. - One Year (12) Cover: $180	You Pay: $149 /	Save: $31
Can/Mex - Two Years (24) $360	You Pay: $179.99 /	Save: $180 + FREE $198 CD
Can/Mex - One Year (12) $180	You Pay: $169 /	Save: $11
Int'l - Two Years (24) $360	You Pay: $189.99 /	Save: $170 + FREE $198 CD
Int'l - One Year (12) $180	You Pay: $179 /	Save: $1

3-Pack

Pick any 3 of our magazines and save up to $275⁰⁰
Pay only $175 for a 1 year subscription plus a FREE CD

- 2 Year – $299.00
- Canada/Mexico – $245.00
- International – $315.00

6-Pack

Pick any 6 of our magazines and save up to $350⁰⁰
Pay only $395 for a 1 year subscription plus 2 FREE CDs

- 2 Year – $669.00
- Canada/Mexico – $555.00
- International – $710.00

9-Pack

Pick 9 of our magazines and save up to $400⁰⁰
Pay only $495 for a 1 year subscription plus 3 FREE CDs

- 2 Year – $839.00
- Canada/Mexico – $695.00
- International – $890.00

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

www.sys-con.com/2001/sub.cfm

SYS-CON MEDIA

Joe Damassa
VP Marketing, WebSphere Infrastructure Software
Tom Inman
VP, WebSphere Foundation & Tools

WebSphere Strategy Focuses on Tried-and-True

Scalable, reliable, and robust transactional computing capabilities that are easy to use

– INTERVIEWED BY JACK MARTIN –

WebSphere Developer's Journal editor-in-chief Jack Martin recently chatted with Joe Damassa and the WebSphere marketing team, which features some new faces but is just as dedicated to the continued success of the WebSphere brand. In this exclusive interview the team discusses open standards, ISVs, Web services – and where WebSphere will go in 2004.

WSDJ: Joe, WebSphere is a mature product and is obviously the market leader. Since you're the new head of WebSphere marketing, I'd like to hear from you and your team about where the platform's been and where it's going, so our readers can understand how this new team sees WebSphere.

Joe: WebSphere is IBM's platform for the next generation of e-business computing. The application server, which launched in 1998, is the core product. The platform also includes WebSphere MQ messaging middleware; business integration software; portal and commerce software; WebSphere Studio, a set of tools that developers use to build applications; and our whole pervasive computing platform.

Perhaps our greatest strength in the marketplace, and the reason for our leadership, is that we have consistently defined and evolved the WebSphere platform to meet the requirements of our customers. We have done that time after time and have watched our competitors follow.

WSDJ: Where do you see WebSphere going in the next year? What should people be looking forward to?

Joe: We're going to continue delivering what our customers need in the way of middleware infrastructure for on-demand computing. That means a continued focus on tooling and integration, and the simplification of the application development process. It means continuing to leverage integration technologies and provide capabilities to build solutions beyond just the tech-heavy ways of doing it. For example, WebSphere Portal enables organizations to provide their customers, vendors, and partners with a single, Web-based access point to the people, processes, and applications that are important to them. We will continue to innovate with the portal to provide easy-to-use tools to access business-critical information via wireless and voice interfaces.

I think our overall mantra has two themes. First, simplification – to try to make it easier to understand and use the WebSphere portfolio, which is very rich and broad. Also, we focus on IBM's tried-and-true value proposition, which is scalable, reliable, and robust transactional computing capabilities. WebSphere has done that for the past five years and we will continue to do that moving forward with the industry's leading infrastructure platform.

Tom: We have always focused on what our customers need in order to be successful, and how to help them implement it. In the early days we started complementary education and services in

ABOUT JOE DAMASSA

Joe Damassa, vice president of Marketing in IBM's WebSphere Software division, is the chief marketing executive for WebSphere, the leading software platform for e-business on demand. He was a member of the team that defined IBM's initial Internet and e-business strategy and previously served as vice president of WebSphere Marketing Programs and Support, responsible for the definition and execution of worldwide marketing plans and programs for WebSphere, and establishing it as the industry's fastest growing e-business software platform.

"Perhaps our greatest strength in the marketplace, and the reason for our leadership, is that we have consistently defined and evolved the WebSphere platform to meet the requirements of our customers" –Joe Damassa

Stefan Van Overtveldt
Director, WebSphere Technical Marketing
Jamie Thomas
Director, WebSphere Strategy & Portfolio Management

best practices that we wrapped around the technology. The next step has been to help customers take advantage of those best practices and services through IBM business partners and IGS [IBM Global Services], and we will continue with that. I would argue that our solutions services support has been a critical element in our success and we will continue that going forward.

WSDJ: How do you see the service strategy evolving for 2004?

Tom: I think we'll further simplify what it takes to build and deploy. It will be easier for all sorts of companies, users, and partners to do it. We will extend the solutions service capabilities to more and more partners. We will concentrate on ensuring that we have the right services in the marketplace and that they are available through the right partner network. And we will continue to make sure that our own services business, IGS – as well as midmarket services geared to partners and integrators – are working to help customers with business transformation.

WSDJ: How do you see the open standards movement in conjunction with WebSphere?

Stefan: It's really inseparable. Obviously, we are very strong believers in open standards. Everything we do right now is centered on being open standards–based.

WSDJ: Like Eclipse?

Stefan: Like Eclipse. Like everything we are doing about Web services, portal technology, and IBM's next-generation integration platform. We are working together with the entire industry, not just IT vendors, but also customers, system integrators, etc., to define what these standards need to be, while obviously competing with the rest of the IT world on the implementation of those standards. I think WebSphere has become not just a gathering point for bringing together open standards, but also it's the way we drive open standards into the market.

"A product or a technology by itself won't win in the marketplace unless it has the right ecosystem of partners, customers, service providers, application providers, and application developers behind it" –Jamie Thomas

WSDJ: What you're saying is that there is going to be a big push down to the midmarket?

Tom: We have had phenomenal success with the WebSphere Express portfolio of products. Approximately 1,400 ISVs have signed up to create applications on WebSphere Express. It's been tremendous.

Jamie: This is about creating an ecosystem. A product or a technology by itself won't win in the marketplace unless it has the right ecosystem of partners, customers, service providers, application providers, and application developers behind it. And it's not just products. Part of building an ecosystem and providing industry leadership is setting standards that help the products become successful. IBM has repeatedly done that in terms of defining approximately 80% of the J2EE spec.

Stefan: Basically, the entire IBM software portfolio leverages WebSphere as a core runtime for building out capabilities. They're doing it on top of a common layer, which makes WebSphere the operating system for e-business. I would guess that just from the software perspective, we probably have more than 20,000 skilled Java developers. That doesn't take into account the developer teams that we have in our industry solutions teams.

WSDJ: Like what's happened with Linux.

Stefan: We see more and more implementation of WebSphere on Linux. It's a really good combination with Eclipse. One of the benefits of open source is that you know when the technical community likes something. Why have there been more than 12 million downloads of the Eclipse platform? Not because developers were told to do it. They want to do it because they believe in it.

Jamie: IBM's focus on industry standards is very appealing to a large constituency among our customer base. They are very interested in J2EE, the typical Web services standard, but they also want to understand that we support industry-specific technology like the UCCnet Standards for retail.

WSDJ: What are the UCCnet Standards?

Jamie: It's a set of standards that retailers use to communicate with their suppliers. So it's a common definition of how you do item synchronization – all the information that you need to acquire between retailers and suppliers.

WSDJ: Is that something that a Wal-Mart type of company would use?

Jamie: Exactly. UCCnet facilitates communication between a large company and all of its suppliers. Item synchronization is one of the biggest challenges within the retail industry.

ABOUT TOM INMAN

Tom Inman is vice president of Marketing for WebSphere Foundation & Tools, where he is responsible for product strategy, offerings, and strategic alliances for the WebSphere Foundation and Tools portfolio. Tom developed the initial business plan leading to the creation of the WebSphere portfolio and brand and helped launch WebSphere in 1998.

Another example is HIPAA and how the government defines standards for sharing information about health care records and maintaining patient privacy. HIPAA typically affects health care providers, insurance companies, doctor's offices, and hospitals.

All of these standards are supported in different ways throughout the WebSphere platform. So when we meet with customers, they want to know that we support traditional IT standards as well as industry standards, which continue to proliferate as we see regulation across different industries.

Joe: We are seeing that once you have the data standards, or the IT standards, then you can define content standards. So, if a patient record is defined as having these pieces as a common standard definition, when insurance companies and hospitals send you a patient record it's not in a proprietary format, so standards facilitate the flow of information.

WSDJ: I never realized the flexibility that WebSphere had, that you can bend around whatever standard a customer wants.

Joe: We are heading in a couple of directions – midmarket, which we talked about, and line-of-business initiatives by industry.

WSDJ: Is it a team that is focused on all of those, is it broken across the whole organization where the retail folks are focused on retail, or is it coming from WebSphere?

Joe: We have a strong industry-focused team coming out of WebSphere Business Integration because that is where a lot of the industry specialization has already taken place. They focus on solutions using the entire WebSphere portfolio – and even the entire IBM software portfolio – dedicated to specific industry solutions. Increasingly, we are moving toward assembling the parts into a customer solution to address customer needs and deliver business value to customers.

Tom: We've described the horizontal technology standards that permeate the WebSphere platform. We talked about industry-specific standards – retail and health care, for example – that organizations care about to solve their unique problems more quickly. Then we talked about having a team focused on understanding the marketplace and bringing together the capabilities of technology – knowing how the services and all the partner networks can solve this problem. All together, this allows us to more quickly solve the problems of our customers.

WSDJ: So if a business partner was going after a midsize health care provider, they've got all the tools in this kit to be HIPAA-compliant right out of the box?

Tom: Absolutely.

Jamie: In many cases, when we offer the solutions to a midsize retailer, for example, we are going into that deal jointly with our partner network. We have quite an extensive industry-based partner network, which allows us to find the right solution for the customer.

WSDJ: How big is the partner network right now?

Jamie: Overall, IBM software group has about 100,000 partners.

Joe: We are taking things down to a level of application server technology, integration technology, user interface technology, and it's surfacing in a lot of places. WebSphere's transaction engine is showing up under the Lotus product in WebSphere Portal and our dynamic workplace. It incorporates DB2, so there is a componentization of these assets across IBM software group.

That also helps our partners and makes customers and developers more efficient because they can reuse what they have. I can take the application server off the shelf and use it with WebSphere Portal to reach mobile workers. They build on each other and we can deliver solutions faster because we are reusing existing components and technology.

Tom: There's the horizontal integration of the technology, which is making us much more efficient and quicker to market. Then there is vertical integration. We take all these components and we deliver them as part of a vertical solution for health industries, for mobile worker force, for government applications. Now the middleware is part of a solution that's more focused on a business value problem. We are providing horizontalization and componentization of the technology, and verticalization of the value.

Jamie: If you really think about end-to-end application development, it can start with business process models. We acquired Holosofx and their business process modeling tool, which has become part of the WebSphere business integration tool portfolio. You can model an application from a business person's perspective to know what kind of result you are going to get at various steps in the business process.

WSDJ: So IBM's tooling strategy has gotten to the point where it's hands-down the best thing on the street?

Jamie: In effect, we took what were, independently, the best tools in their class – Rational Rose with its modeling testing configuration repository design tools, and WebSphere Studio tools that exploit the capabilities of runtime environments. We have taken the best in both instances and have an even broader solution that is the best in its class.

Stefan: And, by the way, it all gets back to our earlier discussion of open standards – and it is all based on Eclipse.

WSDJ: Jamie, what's your perspective on the Eclipse standard?

Jamie: I think it's been a huge asset to us. Through Eclipse and its enormous success – I think Stefan said we've had 12 million downloads over the past three years – we've created an ecosystem of developers out there with the tools that our customers need.

WSDJ: IBM is obviously the world leader in business integration. Where are you going with that?

Joe: That brings us to IBM's focus on helping customers become on-demand businesses. An on-demand business is flexible, agile,

“We have had phenomenal success with the WebSphere Express portfolio of products. Approximately 1,400 ISVs have signed up to create applications on WebSphere Express” –Tom Inman

ABOUT JAMIE THOMAS

Jamie Thomas is director of Strategy & Portfolio Management for WebSphere Infrastructure Software, responsible for strategic business plan development for the WebSphere family. Prior to this assignment she was director for the WebSphere Connectivity and Edge Solutions team, where she was responsible for the worldwide delivery and service of distributed software in the Edge and Host Access market places.

and can react quickly to changes in the marketplace. It is very dynamic. It is not the traditional company where I can't change my business processes because they are locked into old infrastructures.

This is all about providing tools and capabilities for our customers so they can adapt their business and flexibly build new business processes. They can tap into capacity so when a Web site faces peak usage only during the Christmas season, they don't need all that capacity on their floor; they can lease it or use it as a utility function on demand as needed.

On-demand presumes that there are existing assets and capabilities that you want to take advantage of, either within your own organization or in the ecosystem or partners that you deal with. Customers don't have the luxury of rewriting software, and they want to reuse IT investments made over the past three decades.

Stefan: And there's even more involved than reusing applications or the hardware that these applications run on. Customers have made massive investments in building operational procedures around these systems. They know what to do, how to make backups of those applications, how to keep them running 24 hours a day, 365 days a year. You want to be able to leverage that know-how – because if you were to take an application and rewrite it onto a different platform, it involves more than simply the effort of rewriting it; it's also the effort of making that entire set of capabilities operational around it.

One of the things that we realize is that, as Joe said, the characteristics of an on-demand business are variable and flexible when it comes to having a set of requirements for how to build out your IT infrastructure.

The notion of service choreography is that once I have all of my applications defined as business services, I can choreographically create logical flows between those different functions. I can pull from a list of functions and define how to interact and then possibly expose an entire workflow as the outer function that may be reused.

Unlike a lot of our competitors, IBM understands that integration means more than building new applications as modules that can merely be combined. That's just too simplistic for most businesses today.

WSDJ: Instead of doing an integration job with, say, SAP – is that what you're alluding to?

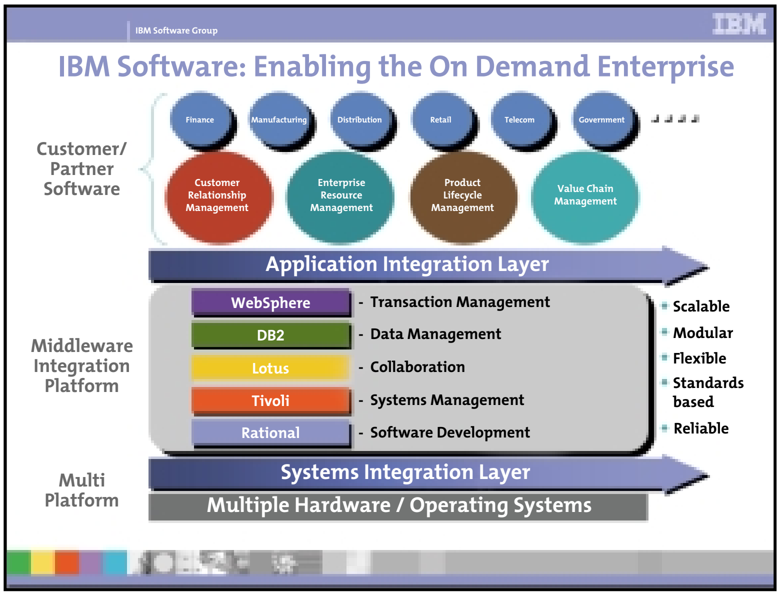
Stefan: SAP typically has, say, 15 or 20 functions. Now you can obviously say, "I need something new there, so I am going to build this into SAP."

Or you can say, "I have these 15 functions defined in SAP. I can define them as Web services as a way to take something from the customer system and from the ordering system and rapidly combine them. We're going to do this by combining the services versus going in and coding a new SAP."

WSDJ: This is Web services as a reality, as opposed to a concept.

Stefan: Exactly. The average insurance company has somewhere between 1,500 and 2,500 applications running. Every one of those was built to support a new function. Why don't they just identify the 400–500 core services and build out new applications by rapidly combining those services and wiring them together in a visual fashion? That's what service-oriented architecture enables.

You can only do that if you also have the technologies in-house to take existing applications, mainframe applications, Unix applications, and Windows applications, and bring them into this new forum.


IBM Software Group
IBM
IBM Software: Enabling the On Demand Enterprise
Customer/ Partner Software
Finance
Manufacturing
Distribution
Retail
Telecom
Government
Customer Relationship Management
Enterprise Resource Management
Product Lifecycle Management
Value Chain Management
Application Integration Layer
Middleware Integration Platform
WebSphere - Transaction Management
DB2 - Data Management
Lotus - Collaboration
Tivoli - Systems Management
Rational - Software Development
Scalable
Modular
Flexible
Standards based
Reliable
Multi Platform
Systems Integration Layer
Multiple Hardware / Operating Systems

WSDJ: What about Grid computing?

Jamie: Like all of on-demand, it's based on an operating environment, which is the software and hardware that you need to deploy and exploit to become an on-demand business.

WSDJ: How would you define an operating environment?

Jamie: An operating environment, in its simplest terms, is a set of functions that encompasses integration capabilities across your enterprise. Think of it in three terms: integration, virtualization, and automation.

Grid computing describes one way of virtualizing IT resources. In reality virtualization can be achieved by a cluster of WebSphere services.

WSDJ: Can you give me an example?

Jamie: Let's say I'm a customer with a cluster of WebSphere servers, and I need redundancy and all those things that maintain a 24/7 business. I have the capabilities within WebSphere Application Server today to manage workload across those servers. IBM will continue to enhance those capabilities to allow you to do even more intelligent, autonomic workload management across the cluster of servers.

WSDJ: I have one final question about the WebSphere platform. Where do you see all this at the end of 2004? The economy is coming together and there is a lot of adoption of WebSphere taking place. What do you think the world will look like?

Joe: The success that we've had, the focus on where we are going in the future, and the fact that we are staying ahead of the marketplace – that's all because we have a talented team of people who are skilled in what they do. They are helping execute in the present and define the future.

As the world becomes more open, it will be easier for our customers to choose the right technology to meet their business needs. Theoretically, the cost of building applications and deploying them will be cheaper for our customers. We will continue to drive that value proposition for our customers – and we do it better than our competitors.

ABOUT STEFAN VAN OVERTVELDT

Stefan Van Overtveldt, director of WebSphere Technical Marketing, manages the IBM WebSphere Internet Infrastructure Technical strategy. His area of expertise includes Internet/e-business technologies and infrastructure, and client/server architectures. He has been a key contributor to the creation of the IBM Application Framework for e-business, and is the author of several IBM whitepapers on e-business technology and e-business infrastructure.

WEB SERVICES

Understand the pressure points

Performance Best Practices for Using WAS Web Services

BY HARVEY GUNTHER

Web services performance comes of age in WebSphere Application Server (WAS) version 5.0.2, but just as with more traditional J2EE applications, the performance of Web services applications is largely determined by the design of the application and the database.

ABOUT THE AUTHOR

Harvey Gunther is a senior performance analyst in IBM's WebSphere Portal Server Performance team. Harvey has 16 years of server runtime performance analysis and a development background with WebSphere Portal Server, WebSphere Application Server, IMS, VisualAge Smalltalk, and Imaging.

E-MAIL

hgunther@us.ibm.com

This article considers the application design factors unique to Web services performance, including the most important: moving to WAS 5.0.2. I will examine the performance of WAS 5.0.2 Web services and establish some best practices for optimizing Web services performance on WAS 5.0.2.

Before discussing specific recommendations, we first need to understand the major performance characteristics or pressure points of Web services.

Web Services Performance Components or Pressure Points

There are three major pressure points to a Web services engine. These three pressure points define the performance "physics" of Web services. They are defined here and shown in context in Figure 1:

- ***Parsing (Input):*** On the receiving side of a Web service, the incoming Web services payload is parsed. There are two major performance components in parsing: (1) scanner, and (2) symbol or name identification.
- ***Deserialization (Input):*** As the document is parsed the XML payload is deserialized or marshaled into business objects that will be presented to the Web services as business object parameters. The Web services provider, either a JavaBean provider or EJB provider, has no awareness of its participation in a Web service.
- ***Serialization (Output):*** On output the inserted reply is marshaled or serialized into XML. Large documents or complex (in terms of the number of elements) objects put pressure on output serialization.

Web Services Best Practices

USE WAS 5.0.2 WEB SERVICES INSTEAD OF SOAP

WAS 5.0.2 introduces a better-performing Web services implementation than the SOAP implementation based on Apache SOAP in WAS 4.0 and iWAS 5.0. However, WAS 5.0.2 still includes support for the WAS 4.0 Apache SOAP implementation for legacy Web services applications. Figure 2 shows the difference in performance between the Apache SOAP and the newer Web services support in WAS 5.0.2. Figure 2 also shows performance with the same object payloads for the Web Services Technical Preview that was in WAS 5.0 and WAS 5.0.1.

Figure 3 shows how performance improved from WAS 4.0 SOAP, to the WAS 5.0–5.01 Web Services Tech Preview to WAS 5.0.2. The performance improvements were largely due to the reduction of parsing and XML-to-object deserialization.

KEEP XML PAYLOADS SMALL AND SIMPLE

This is the main Web services best practice. The performance of your Web service is directly related to the size and complexity of the XML payload transferred. Some of the Web services best practices that follow expand upon this one. Refer again to Figure 2, which shows the performance impact of the input payload's size and complexity in terms of the number of XML elements. Recall the three performance components or pressure points of Web services shown in Figure 1: Parsing, XML-to-object deserialization, and object-to-XML serialization. As input documents increase in size and/or number of elements, they require more processing for both parsing and deserialization. As output documents increase in size and/or number of elements, they require more processing for serialization. As I will explain in the next best practice, you shouldn't reduce the size and complexity of the XML payload by transferring smaller but more frequent payloads.

AVOID FINE-GRAINED MESSAGING

Every Web services request by definition is a remote request, typically involving the Web container or JMS plus the XML overhead of parsing and deserialization. Figure 4 shows the performance implications of fine-grained messaging. If you need to send or retrieve a 50K object that has 10 properties, each 5K long, you can retrieve all 50K as one Web service request, as 50 requests for 1K or some other combination, such as 10 requests for 5K each. Figure 4 shows the performance implications and comparisons. Transferring the 50K in one single Web services request is the best-performing alternative. Transferring 1K, 50 times is the worst-performing alternative. The overhead cost of latency is the major factor.

AVOID DEEP NESTING OF XML PAYLOADS

I previously discussed the first two dimensions of performance:

1. Length of data elements
2. Number of elements or complexity

This best practice adds an important third dimension, level of nesting, which translates to objects defined within other objects and collections of objects defined within other objects. Increasing the level of object nesting translates to an increase in the number of objects that are deserialized and created coming in from the input Web services XML. An object of a certain size that is composed only of primitive types, strings, etc., will yield better Web services performance than a similar size object composed of deeply nested Java objects. Web services deserialization is the differentiator. Figure 5 compares the performance effect of the first two dimensions, size and complexity, with the third dimension, nesting. The chart shows that an object that is 18K in length with 500 complex objects but that is deeply nested will perform worse than an object that – although 50K in length – is comprised solely of 500 strings of Java primitive data types.

USE WAS CUSTOM SERIALIZERS

The WAS 5.0.2 Web services engine provides business object–specific serialization/deserialization assistance that improves runtime performance. For each business object, these custom serializer/deserializer helpers assist the Web services engine by specifically describing the object's properties. This improves performance by reducing the Web services runtime use of introspection to obtain business object–specific information.

The business object–specific custom serialization/deserialization routines are a by-product of WSDL

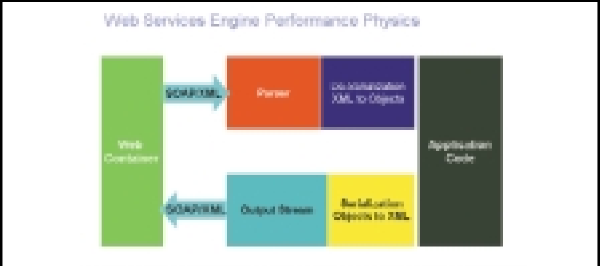

Web Services Engine Performance Physics
Web Container
SOAP/XML
Parser
De-serialization XML to Objects
Application Code
SOAP/XML
Output Stream
Serialization Objects to XML

FIG. 1: THE CONCEPTUAL FLOW AND PERFORMANCE PHYSICS OF WEB SERVICES

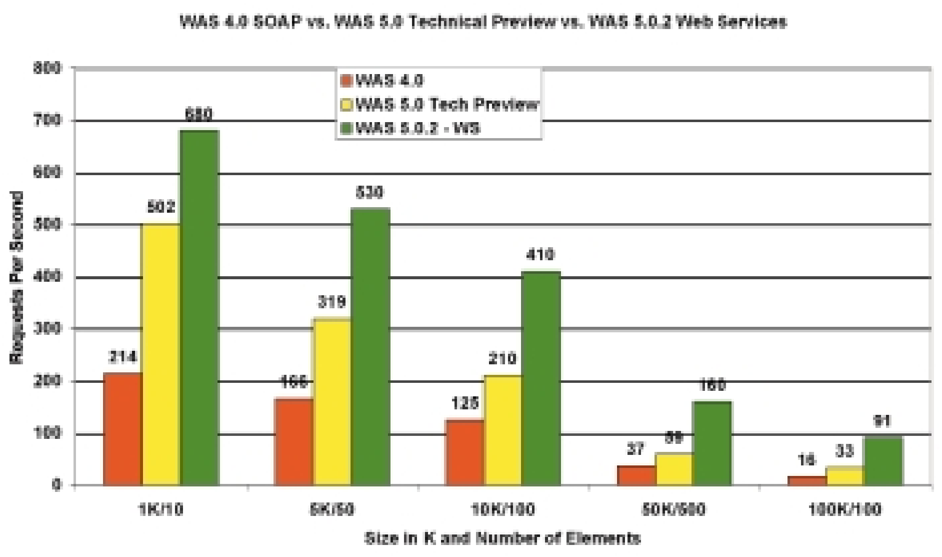


FIG. 2: WAS 5.0.2 WEB SERVICES COMPARED TO WAS 4.0 SOAP AND THE WAS 5.0 WEB SERVICES TECHNICAL PREVIEW

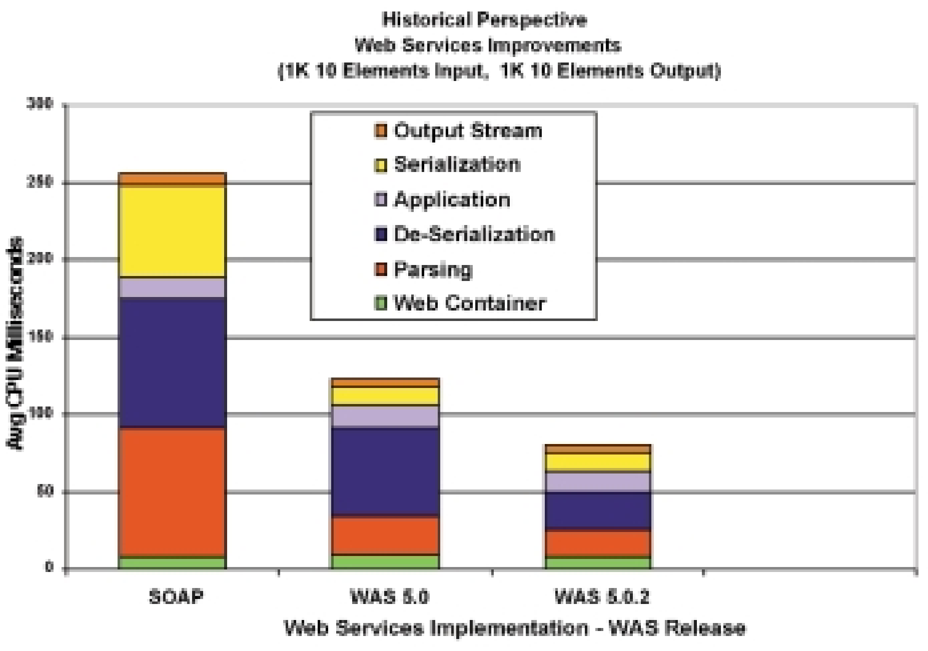


FIG. 3: HOW PERFORMANCE IMPROVED FROM WAS 4.0 SOAP TO WAS 5.0.2

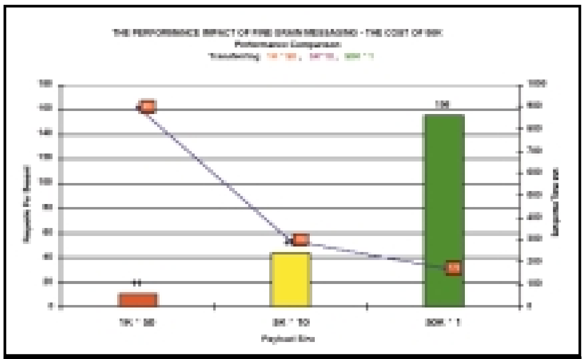


FIG. 4: THE PERFORMANCE IMPACT OF FINE -GRAINED MESSAGING

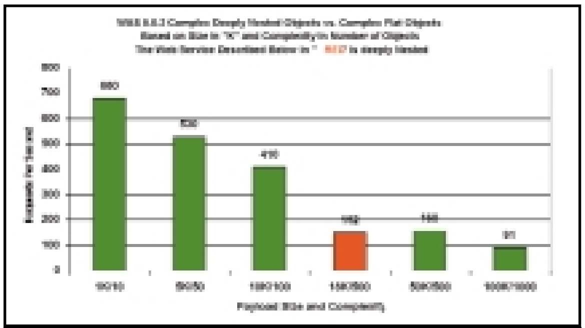


FIG. 5: EFFECT OF DEEP OBJECT-LEVEL NESTING ON PERFORMANCE

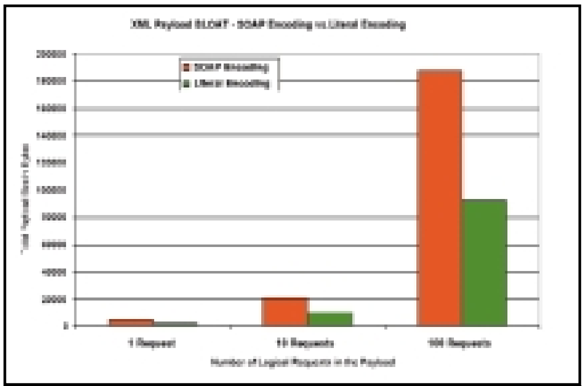


FIG. 6: MESSAGE LENGTH COMPARISON OF SOAP ENCODING VERSUS LITERAL ENCODING

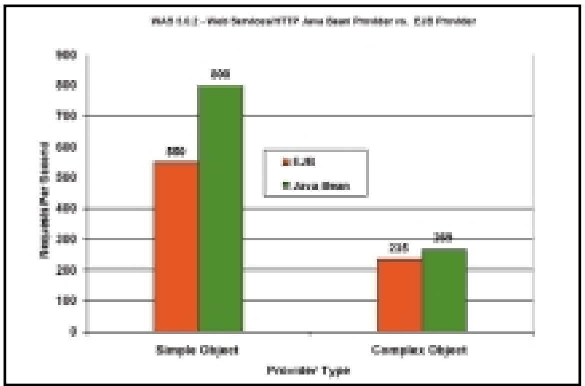


FIG. 7: PERFORMANCE DIFFERENCE BETWEEN THE JAVABEAN AND EJB PROVIDERS

definition during the Web services deployment process. Using these object-specific performance aids is a best practice. In various lab experiments I have seen anywhere from a 5% to 10% improvement in performance by including the custom serializers/deserializers in the packaged client and server implementations.

FAVOR LITERAL OVER SOAP ENCODING

Use literal encoding instead of SOAP encoding. Literal and SOAP encoding are alternate forms for encoding Web services requests and responses. SOAP encoding is the older and now less commonly used form of Web services encoding. Each element in the SOAP body includes the XML Schema definition type of the data element. SOAP encoding was needed to make the message self-defining. SOAP encoding with the embedded data types increases the amount of data transferred in the Web services request. We previously established the performance impact of XML message length. Figure 6 shows a comparison of data lengths for SOAP encoding versus document/literal for a typical workload.

USE DESCRIPTIVE BUT SHORT PROPERTY NAMES

There is a long-standing and fundamental best practice that recommends using long and descriptive property and variable names to make programs more readable and self-documenting. This presents a performance issue when applied to XML-based Web services. This best practice for long and descriptive property and variable names is based on a bytecode binary protocol that uses symbols and other identifiers for variables and properties. Web services is an XML text–based exchange protocol; names of variables and properties are included in the XML for the SOAP body portion of the message. Keep in mind the best practice I discussed earlier – keeping XML payloads small and simple; be aware of the effect that message and SOAP body length have on Web services performance. Use descriptive names for variables and properties but be aware of the impact on message length, and in turn, the impact of message length on performance.

Performance Consideration: Favor JavaBean over EJB for Provider Type

J2EE Web services has two server or provider types: JavaBean and EJB session bean. There is a natural inclination to use EJB session beans as Web services. EJB session beans provide well-defined interfaces that characterize a distributed service endpoint. However, there is extra overhead associated with using an EJB session bean.

Consider using a JavaBean instead of an EJB as the Web services provider. This is not a best practice because there are often application factors that require EJB services for security and transactions. These factors outweigh the performance benefit of a JavaBean provider. If transactions and/or security are not needed for your Web service, then using a JavaBean provider might be the correct and better-performing option for your application.

Figure 7 shows the performance comparison of a JavaBean provider with an EJB provider. The overhead difference has a bigger impact on simpler messages and higher volumes. The performance difference tends to dilute as message size, complexity, and nesting increase.

USE 'PASS BY REFERENCE'

If you choose to use the EJB provider for your Web service, make sure that you use the WAS EJB/IIOP "Pass by Reference" optimization, "No Local Copies". The Web services EJB provider is conceptually nothing more than an XML message sent to a servlet that makes an EJB call. Because all of the artifacts are in the same JVM, there can be a performance boost for using "Pass by Reference" call semantics. Figure 8 shows the performance improvement for using Web services with an EJB provider with the WAS

A new tool for MX professional developers and designers...

ADVERTISE
Contact: Robyn Forma
robyn@sys-con.com
(201) 802-3022
for details on rates
and programs

SUBSCRIBE
www.sys-con.com/
mx/subscription.cfm
1(888) 303-5282

SYS-CON MEDIA

MX
developer's journal

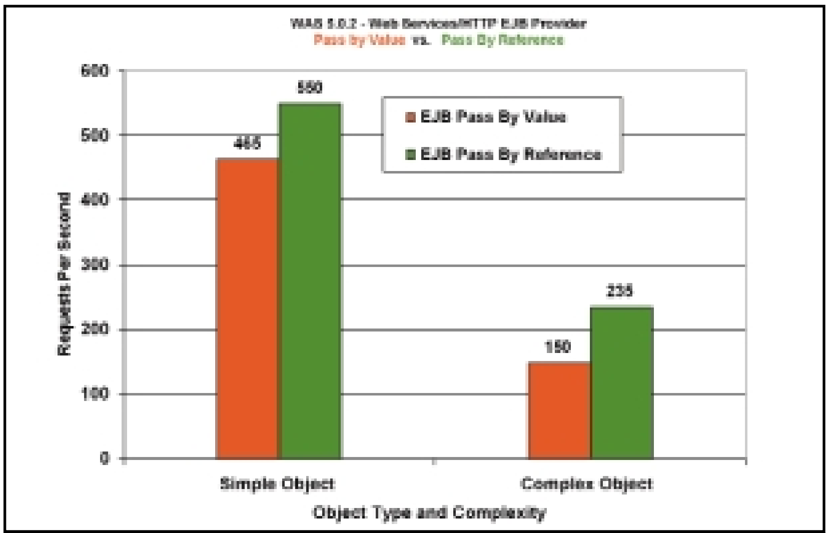


FIG. 8: WEB SERVICES EJB PROVIDER "PASS BY REFERENCE" VERSUS "PASS BY VALUE"

Server Properties for the ORB Properties set to "Pass by Reference".

Conclusions

WAS 5.0.2 provides a significant Web services performance enhancement. Using the performance best practices discussed in this article, and based on WAS 5.0.2, you can write new applications or leverage existing application artifacts without fear of a Web services performance penalty.

Acknowledgments

The author would like to acknowledge:

- Gennaro Cuomo, IBM Distinguished Engineer and CTO of the WebSphere Technology Institute
- Mike Conner, IBM Distinguished Engineer and CTO of Applied Web Services
- Noah Mendelsohn, IBM Distinguished Engineer, Lotus Research
- Ruth Willenborg, manager of the WebSphere Performance team, for her assistance in reviewing this article and securing its publication
- Keys Botzum from IBM WebSphere Services for his unrelenting betterment of my articles
- Glenn Marcy, from Parser Development
- Greg Truty, from WebSphere Development
- Joe Latone and Eric Perkins from Lotus Research

WebSphere Developer's Journal Advertiser Index...

COMPANY	URL	PHONE	PG
BOWSTREET	WWW.BOWSTREET.COM/EASY	603-559-1900	51
CANDLE CORPORATION	WWW.CANDLE.COM/WWW1/AXA1_WSDJ	800-898-4426	2,3
DIRIG SOFTWARE	WWW.EBIZPERFORM.COM	603-889-2777	17
H & W COMPUTER	WWW.HWCS.COM	208-377-0336	47
JAVA DEVELOPER'S JOURNAL	WWW.SYS-CON.COM/JAVA	201-802-3021	35
KENETIKS	WWW.KENETIKS.COM	888-KENETIKS	13
MQSOFTWARE	WWW.MQSOFTWARE.COM/QNAMI	800-998-2858	27,29,31
MX DEVELOPER'S JOURNAL	WWW.SYS-CON.COM/MX/SUBSCRIPTION.CFM	888-303-5282	45
PROLIFICS	WWW.PROLIFICS.COM/WEBSERVICES	800-675-5419	9
QUEST SOFTWARE	HTTP://JAVA.QUEST.COM/JPROBE/WDJ	800-663-4723	19
SYS-CON MEDIA	WWW.SYS-CON.COM/2001/SUB.CFM	888-303-5282	36,37
TRILOG GROUP	WWW.FLOWBUILDER.COM	800-818-2199	52
VERSATA	WWW.VERSATA.COM/BUSINESSLOGICDESIGNER	800-984-7638	23
WEBSERVICES EDGE EAST 2004	WWW.SYS-CON.COM	201-802-3069	33
WILY TECHNOLOGY	WWW.WILYTECH.COM	888-GET-WILY	5

WebSphere Developer's Journal Coming Next Month...

INTEROPERABILITY

Implementing J2EE-.NET Interoperability Using WebSphere MQ, Part 2

BY BORIS LUBLINSKY AND DIDIER LE TIEN

DATA MINING

If You Build It Well, They Will Come: WebSphere Personalization

BY E. KENNETH NWABUEZE

TOOLS

Managing the WebSphere Platform

BY MARINA GIL-SANTAMARIA

WEB SERVICES

Breathing New Life Into Legacy Systems

BY JEFFREY PAYNE

H & W COMPUTER

WWW.HWCS.COM

IBM, RIM Team to Extend Reach of BlackBerry

(Somers, NY, and Waterloo, ON) – IBM and Research In Motion have announced an agreement aimed at providing mobile workers with better access to enterprise information and applications. This relationship, with development and integration efforts focused on IBM's WebSphere Everyplace Access (WEA) mobile middleware and BlackBerry Enterprise Server, allows BlackBerry users to access and work with enterprise data and wirelessly sync their handhelds with server applications, according to the companies. Developers will also be able to easily customize enterprise applications for the BlackBerry platform.

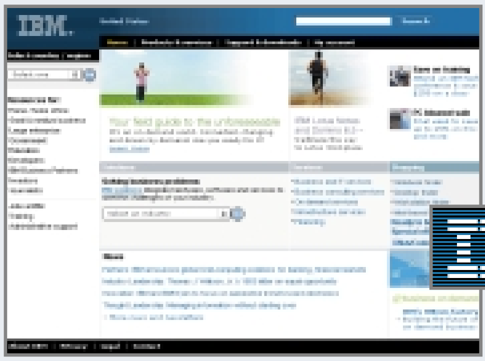
The relationship combines the advantages of IBM's enterprise application access capability with RIM's BlackBerry Mobile Data Service and wireless platform. A broad portfolio of portlets linked with a variety of applications enable end users to easily customize their start page – providing better control over their interaction with their BlackBerry handhelds. IBM's WebSphere Studio Developer toolkit can be used to easily modify or build new portlets to access additional enterprise applications, allowing developers to bring enterprise solutions to BlackBerry users faster and more efficiently.

www.blackberry.com
www.ibm.com

IBM Expands Pervasive Computing Portfolio

(Somers, NY) – IBM has announced a range of product enhancements and partnerships designed to transform the way enterprise applications are delivered to wireless devices. Central to these announcements is new mobile middleware technology, Extension Services for WebSphere Everyplace, aimed at enabling new and existing applications and services to be extended to pervasive devices more easily.

New offerings also include device management applications from partners and new speech and multimodal enhancements that can provide more natural ways to access technology. The announcements build on growing industry support for IBM's open standards–based mobile middleware portfolio, according to the company.



Starting in September, developers will be able to make use of extended Java functions on Palm devices via IBM's WebSphere Micro Environment. Optimized for Palm's Tungsten line of handhelds, this is the result of an ongoing relationship with Palm. IBM now supports more handheld operating systems worldwide than any other vendor: RIM, Palm, Linux, Symbian, and Pocket PC. Together, these handheld operating systems make up at least 90% of the worldwide market, according to Gartner.

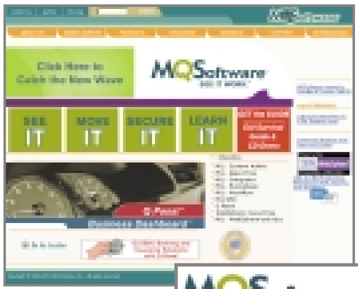
IBM's new Extension Services for WebSphere Everyplace technology is embedded middleware that enables partners, device manufacturers, and enterprise customers to extend the WebSphere platform and Java-based applications to devices. Rather than rewriting applications for mobile devices or requiring a connected micro-browser to access information, this technology provides a services-oriented runtime environment that enables connection-independent deployment and life-cycle management of applications and network services, according to IBM. With Extension Services technology, end users can remotely download only portions of applications and data they need to complete secure transactions – whether or not they are connected to the network.

www.ibm.com

MQSOFTWARE ANNOUNCES Q PASA! VERSION 3.1

(Minneapolis) – MQSoftware, Inc., a leading provider of middleware technology solutions for the IBM WebSphere family of products, has announced a new release of its flagship Q Pasa! middleware management and monitoring solution. Version 3.1 adds a Web-enabled management console that will facilitate anytime, anywhere authorized access to real-time data through Q Pasa!'s unique business dashboard, the company says. The Q Pasa! Business Dashboard gives a meaningful, consolidated view of what is happening in the middleware environment in real-time, according to the company, helping enterprises proactively manage technology and associated resources.

Q Pasa! version 3.1 has raised the bar on the product's ability to monitor very large enterprise implementations with demonstrable real-world scalability, MQSoftware says, adding that they recently deployed Q Pasa! at one of the world's largest WebSphere MQ sites in a simulated pre-production rollout. The Q Pasa! simulation included over 5,000 WebSphere MQ queue managers reporting to a single Q Pasa! server running on AIX. The customer was able to monitor, manage, and provide alerts on over 250,000 properties in their MQ



infrastructure, enabling them to monitor and manage over 5,000 distinct sites from a single server. The roll-out also included real-time traffic from their z/OS mainframe, Linux, AIX, and Windows agents where all these updates were captured historically in a DB2 7.2 instance running on AIX hardware and sustaining 5,000 database inserts a second.
www.mqsoftware.com

LIBRADOS ANNOUNCES EXPANDED SUPPORT FOR WEBSPHERE PLATFORM

(San Ramon, CA) – Librados Inc., a leading provider of JCA adapter technology, has announced that it will expand its support of IBM's WebSphere Internet infrastructure platform with Librados' royalty-free Enterprise Integration Component Server (EICS) and family of pre-built JCA Adapters.

Librados adapters will enable IBM WebSphere customers to extend and expose back-end EIS functionality from systems such as SAP R/3, Siebel, and PeopleSoft into the WebSphere Application Server via XML, Web services, or any other data format. Librados' solution, built on IBM WebSphere, will reduce the cost of application integration by offering either royalty-free source code to independent software vendors (ISVs) or extremely low-cost software to customers with its pure J2EE product offering on WebSphere, an open platform that enables customers to connect disparate systems with the scalability desired for future growth.

Librados offers pure Java/J2EE products so there is no gateway between the WebSphere Application Server and Librados' EICS and JCA Plus Adapters. This facilitates the ability of Librados' products to fully leverage the scalable and reliable foundation of WebSphere Application Server that helps customers to develop, deploy and integrate next-generation e-business applications, from basic Web publishing to enterprise-scale transaction processing. Since the solution is built on an open platform no changes are required to target applications, enabling trouble-free integration of packaged ERP and CRM applications, as well as legacy mainframe applications.
www.librados.com

PORTLET FACTORY NOW SUPPORTS WEBSPHERE PORTAL 5.0

(Tewksbury, MA) – Bowstreet, a provider of development tools for adaptive J2EE applications, has announced the availability of Bowstreet Portlet Factory version 5.6.2. This new release, which is now available, features support for IBM WebSphere Portal 5.0.

"As an IBM strategic partner, we are pleased to support WebSphere Portal 5.0 right from its initial release date," said Rose O'Donnell, vice president, engineering, Bowstreet. "The integration of WebSphere Portal 5.0 and Bowstreet Portlet Factory provides simpler installation, better integration and search, new productivity tools, and the ability to create, deploy, and customize portlets quickly and easily."

Bowstreet Portlet Factory supercharges the WebSphere Portal Server with tools and technology for rapidly creating, customizing, maintaining, and deploying portlets. The Portlet Factory's ease of use and advanced development features dramatically streamline the entire portlet development process, enabling developers to deliver adaptive, robust portlets in a fraction of the time and at a fraction of the cost it would take otherwise, according to Bowstreet.
www.bowstreet.com

DataPower Enhances Web Services Security and Performance for WebSphere MQ

(Cambridge, MA) – DataPower Technology, Inc., a provider of intelligent XML-Aware Network (XAN) infrastructure, has announced that its latest firmware release of DataPower XS40 XML Security Gateway and XA35 XML Accelerator network devices includes support for IBM's WebSphere MQ protocol. This integration, achieved in part by establishing a technology licensing relationship with IBM Corporation, enables direct access to the XML acceleration and Web services security benefits of DataPower's XAN products from MQSeries applications.

Support for MQ also makes it possible to use the XS40 XML Security Gateway as a wirespeed trusted gateway between internal MQ-based Web services and external HTTP Web services. The integration is available immediately as an optional part of firmware Release 2.3, according to DataPower.

"WebSphere MQ is the de facto standard for messaging and queuing and provides our customers the reliability for guaranteed transport of Web services messages over the Internet," says Rachel Helm, director of Product Management for WebSphere Business Integration at IBM. "Integrating MQ with a reliable, high-performance security gateway is a tremendous advantage for our customers needing to securely integrate with partners over the Internet."

The WebSphere MQ integration is part of DataPower's Firmware Release 2.3, which includes other XML intelligence features such as a WS-I Basic Profile 1.0–compliant SOAP interface, signed message auditing, message velocity control, and bad message capture.
www.datapower.com.

RSS Feed Now Available on WSDJ Home Page

The *WSDJ* home page (www.sys-con.com/websphere) now offers an RSS feed (www.sys-con.com/WebSphere/feed.rss) to help you keep current with the latest goings-on in the WebSphere community. Sign up today to start receiving timely updates of product announcements, news items, and other important happenings in the world of WebSphere.

Migrating from Notes to WebSphere

The exodus of a million Notes professionals

BY ALEX **EL HOMSI**

Lotus Notes certainly was one of the most successful rapid application development platforms of the '90s. The speed at which you could develop workflow applications to streamline your business processes was simply unparalleled. What made Notes ideal for developing this particular class of applications is a unique framework that was architected from the ground up around three fundamental concepts:

- A rich document-centric development model
- A sophisticated security model to control information sharing
- A tight integration with messaging and directory services

But Notes still has many idiosyncrasies and design limitations that can be directly attributed to its 15-year-old proprietary architecture. Seasoned Notes developers have built valuable expertise in working around these limitations and made a career out of their Notes and Domino skills. But today it is no longer Lotus Notes, but rather J2EE, WebSphere, and WebSphere Portal that are the new IBM technology darlings.

If you are a Notes developer and your company hires a team of fresh Java talent to work on WebSphere projects while excluding you, I would be quite worried if I were you. You can always try to make the case that developing the same application in Notes would be a much better choice because it will be five times faster to implement. But it seems that developer productivity and the ability to work closely with the business are no longer driving IT decisions these days.

I have been following the heated debate about the future of Domino on the different forums and blogs. What I find most interesting about this debate is that developers tend to think that their survival in the job market depends on the survival of the technology that they have invested so many years to master. The sad truth is that Notes jobs are on the decline and WebSphere jobs are on the rise. So the survival of Notes as an application development platform is clearly not what Notes professionals should bank their careers on. The move of a million developers to standards-based application development is inevitable, if quite painful.

While IBM has done a very good job at marketing WebSphere, it has done very little so far to help the Notes/Domino community make the transition to WebSphere. From a developer perspective, there is almost nothing in common between the Notes and J2EE development models. It's like comparing apples and oranges. This means a steep learning curve and – especially for collaborative/workflow applications – a significant loss of productivity. Notes is a much higher-level application development model. Using the Domino Designer, in no time you can create a database, forms, and views to access documents created with these forms. In contrast, with J2EE you have to understand Java, the J2EE APIs, and the Model-View-Controller architecture. Even frameworks like Struts are functionally very poor compared to the Notes framework.

Scripting inside HTML tags can be construed as bad design, as it cannot preserve the WYSIWYG development experience of Lotus Notes or Microsoft Visual Basic. The latest attempt by IBM and the Java community to improve the JSP developer experience is called JavaServer Faces (JSF), which was proposed in JSR127. Simply put, JSF will allow JSP developers to manipulate visual components instead of taglibs. IBM has announced that it is currently developing a Web RAD component for a future release of WebSphere Studio based on JSF and is advertising that this will make J2EE more accessible to Notes developers. Quite frankly, JSF is no better than Microsoft's ASP.NET, and this is probably why more and more developers are weighing the benefits of .NET rather than WebSphere right now.

Perhaps the solution is to port the Notes development model itself on top of the J2EE stack. This seems like quite a daunting task, but it suffices to re-create the entire Notes framework in native XML. Consider an XML document–centric development model, an LDAP-based security framework, and a virtual XML document store to replace the NSF format for information sharing.

Also, a "VB-like" component model can enhance the J2EE development experience to make it completely WYSIWYG. Draw an HTML form, drop XML documents as data sources, and bind them to the form fields using W3C's XPath standard. There is absolutely no reason why J2EE should be more complex than Notes. EJBs are dead. You want scalability? Try Grid computing. I just came back from OracleWorld, and the new Oracle 10g infrastructure gives you infinite scalability without having to specifically design for it. Just add as many cheap computers to the Grid as you need when you need it. It's that simple.

As always, expect innovation to come from the Lotus Business Partner community itself. When you have a million captive developers, there is a high likelihood that someone has already started digging, and chances are that the tunnel has already been dug. Of course, there will always be those who will continue waiting for the door to magically open some day. But the good news is that there is life after Notes.

ABOUT THE AUTHOR... Alex El Homsi is the founder and CEO of Trilog Group, a leading software company that offers advanced tools and solutions for Notes developers to migrate to J2EE, WebSphere, and WebSphere Portal.

E-MAIL
alex@triloggroup.com

BOWSTREET

WWW.BOWSTREET.COM/EASY

TRILOG GROUP

WWW.FLOWBUILDER.COM