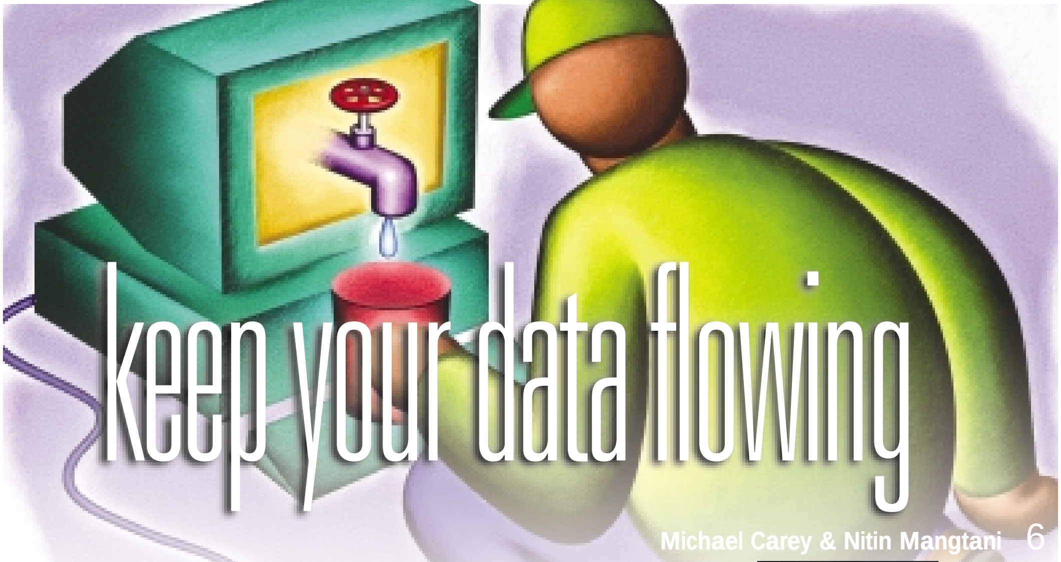
EXTENSIBILITY • DATA • INTEROPERABILITY

BEA WebLogic DEVELOPER'S JOURNAL

OCTOBER 2003 - Volume:2 Issue:10

weblogicdevelopersjournal.com



keep your data flowing

Michael Carey & Nitin Mangtani 6

RETAILERS PLEASE DISPLAY UNTIL NOVEMBER 30, 2003

$8.99US $9.99CAN

10

0 09281 03424 7

SYS-CON MEDIA

Hewlett-Packard

http://devresource.hp.com/d2d.htm

Trading Places

BY JOE MITCHKO

I have always been a firm believer in the value and importance of trade publications in the information technology industry. I remember back in my early days as a consultant being assigned to (more like thrown into) the maintenance end of an Informix-4GL project. Now, mind you, at that time I didn't have any prior experience working with relational database technology, let alone working with Informix.

But, giving it my best shot, I started reading the truckload of manuals that came with the product, hoping that I would be able to learn it fast enough to be productive on the project. I was able to pick up the programming part without much trouble, but relational database technology – that was a different story.

Sometimes the best way to learn something is just to get in there and do it, so it wasn't long before I was able to create a table or two and run some simple SQL statements. Over time, I found myself more and more fascinated with the technology. Somewhere along the line, I made a conscious decision to take steps toward becoming a database analyst, somehow, some way. Not only were data analysts and administrators held in high esteem at the time (at least in my mind), but they were also highly paid.

But how does one graduate from the programmer ranks to the higher echelon of projectland? In my opinion, keep learning as much as you can, any way that you can, and with a little luck, you can do it.

At the time, you couldn't get anything better than subscribing to *Database Programming & Design* magazine. Month after month, I would religiously read the magazine cover to cover, learning everything from relational theory to SQL tips (Joe Celko was my idol) to database tuning. Gradually, over the years of learning and project experience, I was able to work my way toward landing more database analysis and design assignments, so much so that at one time, I held a data architect title for a small Internet startup firm. I'd have to say there were a few nay-sayers along the way, saying that you cannot become an expert just by reading a magazine month after month. But, it surely can't hurt.

Now, years later, and not limited to just database technology but J2EE and BEA WebLogic as well, I find myself in a new position as editor-in-chief of ***WebLogic Developers Journal***, something I could not have imagined doing years ago when I was an avid reader of *DBPD*. But now, instead of eagerly waiting for the magazine to arrive in the mail, I'll be hard at work behind the scenes trying to guarantee that you, the reader, receive the same high quality of articles and monthly columns that I have come to expect.

Over the next several months, we plan to expand *WLDJ* to include articles on architecture and administration, as well as continue to provide new insights on the design and development end of things. Since technology is ever evolving, there will always be something new to learn on the BEA WebLogic platform – and that is why we're here.

This month, we focus on data integration (an old familiar topic for me) – it's not just databases anymore – and the various XML-based technologies that have come about in just the past few years.

We're on the verge of a new and exciting time in the industry, with Web services finally taking off and the birth of new technologies for business process management. With this growth, BEA will continue to be in the forefront, developing the standards and technology platform to make it all happen. And I hope that *WebLogic Developers Journal* becomes your monthly window into what's new and exciting at BEA. And, for those of you who want to become a WebLogic developer or architect, keep reading. You never know!

AUTHOR BIO...

Joe Mitchko is the editor-in-chief of *WebLogic Developer's Journal*. He is a senior technical specialist for CSC Consulting in New York and was product review editor of *Web Services Journal*.

CONTACT: joe@sys-con.com

BEA WebLogic DEVELOPER'S JOURNAL

EDITORIAL ADVISORY BOARD
TOM ANGELUCCI, LEWIS CIRNE, STUART HALLOWAY,
KEVIN JONES, TYLER JEWELL, WAYNE LESLEY LUND,
SEAN RHODY, CARL SJOGREEN, BLAKE STONE

FOUNDING EDITOR
PETER ZADROZNY

EDITOR-IN-CHIEF
JOE MITCHKO

PRODUCT REVIEW EDITOR
JASON SNYDER

EXECUTIVE EDITOR
GAIL SCHULTZ

EDITOR
NANCY VALENTINE

ASSOCIATE EDITORS
JAMIE MATUSOW, JEAN CASSIDY

ASSISTANT EDITOR
JENNIFER VAN WINCKEL

WRITERS IN THIS ISSUE
MIKE CAREY, AMIT DAYAL, DIDIER GUZZONI,
PETER HOLDITCH, WILL LYONS, NITIN MANGTANI,
JOE MITCHKO, KUNAL MITTAL, MARK NOTTINGHAM,
ROBERT PATRICK, VADIM ROSENBERG, MARK
ROTHERA, JASON SNYDER, HUB VANDERVOORT

SUBSCRIPTIONS
For subscriptions and requests for bulk orders,
please send your letters to Subscription Department.
SUBSCRIPTION HOTLINE:
888-303-5282
Cover Price: $8.99/Issue
Domestic: $149/YR (12 Issues)
Canada/Mexico: $169/YR
Overseas: $179/YR
(U.S. Banks or Money Orders)

PRESIDENT AND CEO
FUAT KIRCAALI

VP, BUSINESS DEVELOPMENT
GRISHA DAVIDA

SENIOR VP, SALES & MARKETING
CARMEN GONZALEZ

GROUP PUBLISHER
JEREMY GEELAN

TECHNICAL DIRECTOR
ALAN WILLIAMSON

PRODUCTION CONSULTANT
JIM MORGAN

ART DIRECTOR
ALEX BOTERO

ASSOCIATE ART DIRECTORS
LOUIS F. CUFFARI • RICHARD SILVERBERG

ASSISTANT ART DIRECTOR
TAMI BEATTY

VP, SALES & MARKETING
MILES SILVERMAN

ADVERTISING SALES DIRECTOR
ROBYN FORMA

DIRECTOR OF SALES AND MARKETING
MEGAN MUSSA

ADVERTISING SALES MANAGER
ALISA CATALANO

ASSOCIATE SALES MANAGERS
CARRIE GEBERT • KRISTIN KUHNLE

PRESIDENT, SYS-CON EVENTS
GRISHA DAVIDA

CONFERENCE MANAGER
MICHAEL LYNCH

REGIONAL SALES MANAGER
JAMES DONOVAN

FINANCIAL ANALYST
JOAN LAROSE

ACCOUNTS RECEIVABLE
KERRI VON ACHEN

ACCOUNTS PAYABLE
BETTY WHITE

VP, INFORMATION SYSTEMS
ROBERT DIAMOND

WEB DESIGNERS
STEPHEN KILMURRAY • CHRISTOPHER CROCE

ONLINE EDITOR
LIN GOETZ

CIRCULATION SERVICE COORDINATORS
NIKI PANAGOPOULOS • SHELIA DICKERSON
EDNA EARLE RUSSELL

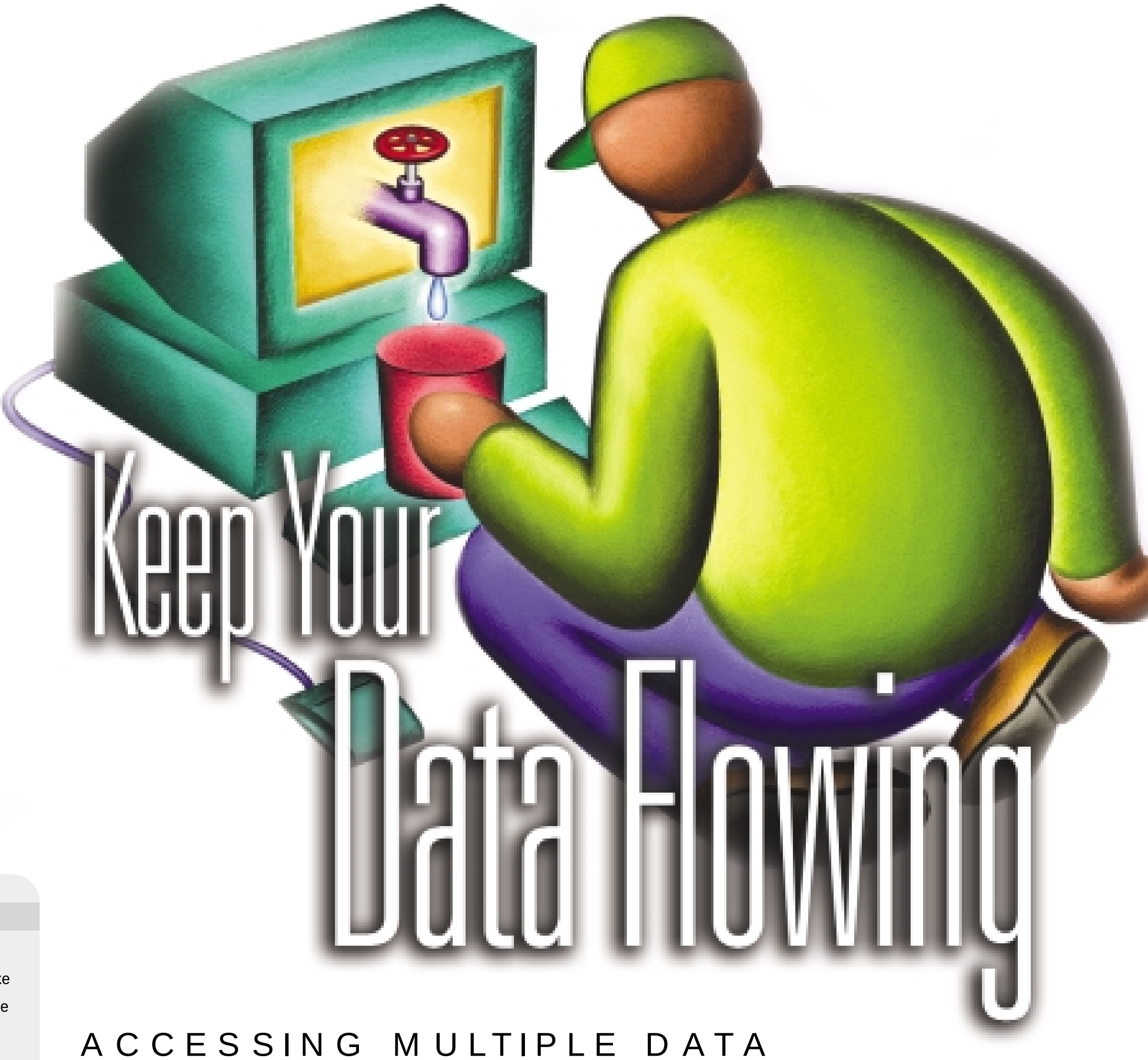
JDJ STORE MANAGER
RACHEL MCGOURAN

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
BEA WebLogic Developer's Journal (ISSN# 1535-9581)
is published monthly (12 times a year)
Postmaster: Send Address Changes to
BEA WEBLOGIC 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

SYS-CON MEDIA



Keep Your Data Flowing

ACCESSING MULTIPLE DATA SOURCES MADE EASY

BY MICHAEL CAREY & NITIN MANGTANI

AUTHOR BIO...

As a University of Wisconsin professor from 1983-1995, Mike Carey did research on database systems. He worked on data integration, DB2, and XML middleware at IBM from 1995-2000. He is now the architect for BEA Liquid Data for WebLogic. Mike is an ACM Fellow and a member of the National Academy of Engineering.

Nitin Mangtani is the technical program manager for BEA Liquid Data for WebLogic. As a founding member of the Liquid Data team, Nitin has been instrumental in defining the overall product vision as well as shipping Liquid Data. Previously, he worked at i2 Technologies in the distributed order management space

CONTACT...

mcarey@bea.com

nitinm@bea.com

REPRODUCED WITH PERMISSION FROM BEA SYSTEMS.

Providing a first-class online user experience can require access to multiple sources of data.

The required data often resides in multiple databases, packaged applications, and other information silos. Accessing and relating such data is a key challenge facing modern enterprises. The IT world knows this problem as the enterprise information integration (EII) problem: enterprise applications need to be able to easily access and combine information about a given business entity from a distributed and highly varied collection of information sources. To meet this need, BEA offers Liquid Data for WebLogic.

Liquid Data provides unified access to data in relational databases, Web services, packaged applications (through J2EE CA adapters and application views), XML files, XML messages, and, through a custom function mechanism, most any other data source as well. It provides default XML views of all of the enterprise data sources, and includes an XQuery-based graphical view-and-query editing tool for integrating and enhancing information drawn from one or more of the data sources. It includes a distributed XML query processor and provides advanced features such as query result caching and data source–level and stored query–level access control.

A general overview of BEA Liquid Data for WebLogic appeared in ***WLDJ*** (Vol. 2, issue 4) earlier this year. Related articles (*XML-Journal*, Vol. 2, issues 6 and 7) described Liquid Data's overall approach to enterprise information integration, namely the use of XQuery and related XML standards to define and query XML views of disparate data sources. This article highlights the key new features of Liquid Data for WebLogic 8.1.

Liquid Data 8.1

The main theme for the BEA Liquid Data for

WebLogic 8.1 release is seamless interoperability with the rest of BEA WebLogic Platform 8.1 for application developers. Within this theme, three areas of interoperability with other WebLogic Platform components were key focal points:

1. Making it easy for portal developers to consume results from Liquid Data queries
2. Making Liquid Data easy to use as an enterprise data access layer in Web services
3. Making Liquid Data easy to use for data enrichment and augmentation in integration projects

A major theme for the WebLogic Platform 8.1 release is the provision of a unified integrated development environment, the BEA WebLogic Workshop IDE, where developers can create entire applications that include portals, Web services, and integration components. The WebLogic Workshop IDE programming model is based on the notion of Java controls – simple components with a visual representation in the design view that have methods, events, and properties – and the generation and use of annotated Java code that enables developers to declaratively specify desired behaviors instead of writing complex J2EE infrastructure code. Java controls provide simplified access to J2EE resources such as JDBC data sources, JMS queues, EJBs, or Web services. Application developers can also create new controls to encapsulate business logic in a reusable package that other application developers can then use. Liquid Data for WebLogic 8.1 provides a Liquid Data control that makes it easy for WebLogic Workshop applications to use Liquid Data queries.

In addition to providing the new Liquid Data control, Liquid Data now includes support for accessing delimited files and "in-flight" XML fragments in queries. And, it provides a means to utilize SQL stored procedures and vendor-specific SQL extensions in developing queries and views. Improvement-wise, this release includes important enhancements in the area of query builder UI usability.

In the remainder of this article we'll build a distributed query using Data View Builder (Liquid Data's query builder UI) and then create a Liquid Data control to encapsulate the query in the WebLogic Workshop IDE. After creating this control, we'll demonstrate how easy it is to create portals, page flows, Web services, and workflows that use this control. At the end of this article we'll look at some of the other features and improvements mentioned above.

Liquid Data Java Control

The BEA Liquid Data for WebLogic Java control allows application developers to develop applications in the BEA WebLogic Workshop IDE that can easily consume Liquid Data queries. When creating an instance of the control using the Liquid Data control design environment, application developers can browse the Liquid Data server repository and select one or more stored queries. The selected queries then become available as methods on the control. In addition, the control also provides a way for applications to directly issue ad hoc queries against the Liquid Data server. In both cases, the query results are returned to the application using WebLogic Workshop's XMLBeans technology. For each method, the return type is an XMLBeans instance that the Liquid Data control design environment auto-generates based on the query's XML Schema result type.

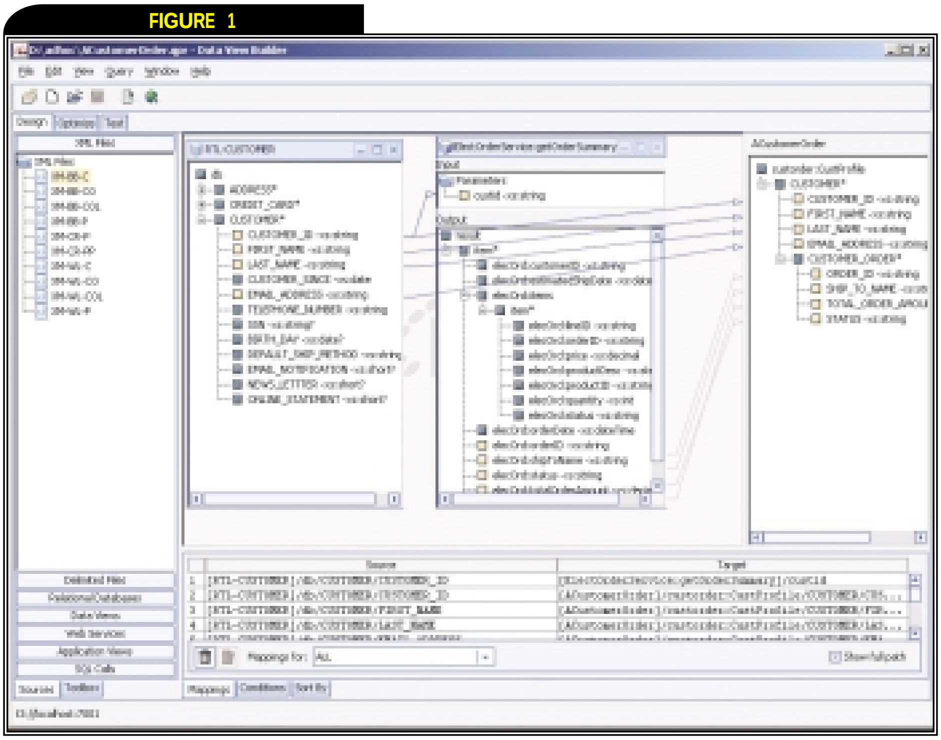
For those of you who haven't encountered XMLBeans before, it's a new XML-Java binding technology from BEA that uses XML Schema as a basis for generating Java classes that provide convenient, typed access to XML instance data in Java programs. XMLBeans was designed to support both type-safe Java access to XML instance data as well as full access to the underlying XML data itself, combining the best features of low-level APIs like SAX and DOM (which provide full access) with the convenience of Java binding. Its generated type-safe Java classes provide schema-constrained read/write access to the underlying XML data, while full access to the data is provided via an efficient cursor-based interface. The XML cursor defines a location in the underlying XML data where a program can perform actions (e.g., set or get values, insert or remove pieces of XML, copy chunks of XML to other parts of the document) on the selected XML data. By delivering its query results via XMLBeans, the Liquid Data control makes it easy for a WebLogic Workshop application developer to create page flows, portals, Web services, and workflows that use XML query results obtained through calls to a Liquid Data control.

An Example Application Scenario

Avitek is an online electronics retailer that wants to build and deploy a new customer self-service application. The goal is to reduce call center expenses by adding customer self-service capabilities to the Avitek Web site. Once a customer logs in, he or she should be able to view their profile information and check their order information. To build this customer self-service portal for Avitek, we'll start by developing a CustomerOrder query to retrieve the customer profile and order information for a given online customer.

To keep the example simple, suppose that there are two data sources containing

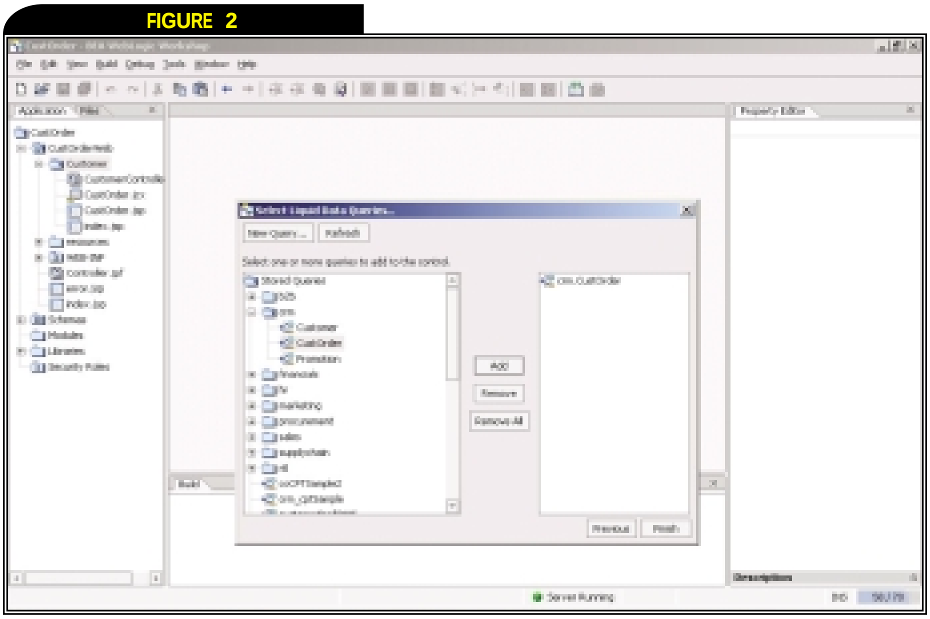
FIGURE 1



	Source	Target
1	[RTL-CUSTOMER]/db/CUSTOMER/CUSTOMER_ID	[illegible]
2	[RTL-CUSTOMER]/db/CUSTOMER/CUSTOMER_ID	[AOCustomerOrder]/customer:CustProfile/CUSTOMER/CUS...
3	[RTL-CUSTOMER]/db/CUSTOMER/FIRST_NAME	[AOCustomerOrder]/customer:CustProfile/CUSTOMER/FIR...
4	[RTL-CUSTOMER]/db/CUSTOMER/LAST_NAME	[AOCustomerOrder]/customer:CustProfile/CUSTOMER/LAS...

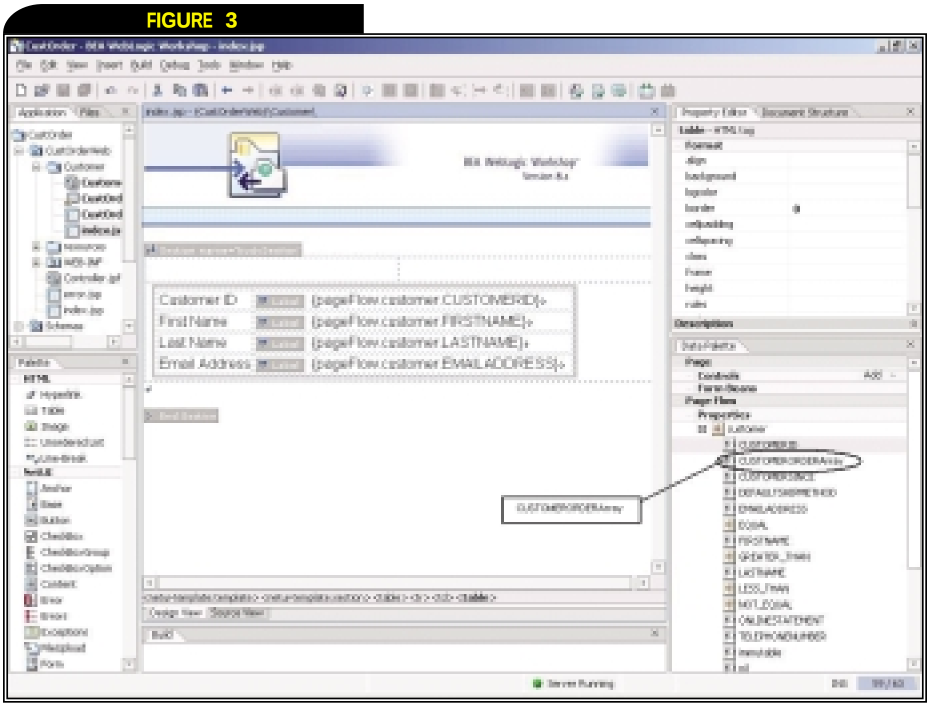
Query to obtain customer profile and order information

FIGURE 2



Query selection dialog for Liquid Data control creation

FIGURE 3


BEA WebLogic Workshop
Customer ID
{pageFlow.customer.CUSTOMERID}
First Name
{pageFlow.customer.FIRSTNAME}
Last Name
{pageFlow.customer.LASTNAME}
Email Address
{pageFlow.customer.EMAILADDRESS}
CUSTOMERORDERArray
Server Running

Page flow design view

data relevant to the customer self-service portal:

- ***RTL-Customer:*** A relational database that contains the customer profile information
- ***ElectOrderService:*** A Web service fronting a packaged order management application that contains the electronics order information for Avitek's customers

Graphical Customer Order Query

Figure 1 shows how the BEA Liquid Data for WebLogic 8.1 query builder tool, Data View Builder (DVB), can be used to graphically construct the query that assembles all the information needed for Avitek's customer self-service information page. The shape of the query result is determined by its target schema (shown on the right); the target schema contains a top-level element called CustProfile. CustProfile has a child element, Customer, that has basic profile information such as Customer_ID, First_Name, Last_Name, Email_Address. The Customer element itself has a child element Customer_Order that contains summary information for any/all orders for a given customer. The customer profile information for the query comes from the Customer table in the relational data source RTL-Customer. The order information for a given customer results from calling the Web service method ElectOrderService.getOrderSummary(). This Web service call takes custid as a single input parameter that comes from the Customer_ID column of the Customer table. The query is designed to be a parameterized query that takes a customer id as input and returns the profile and order information for the indicated customer.

Customer Order XQuery Source

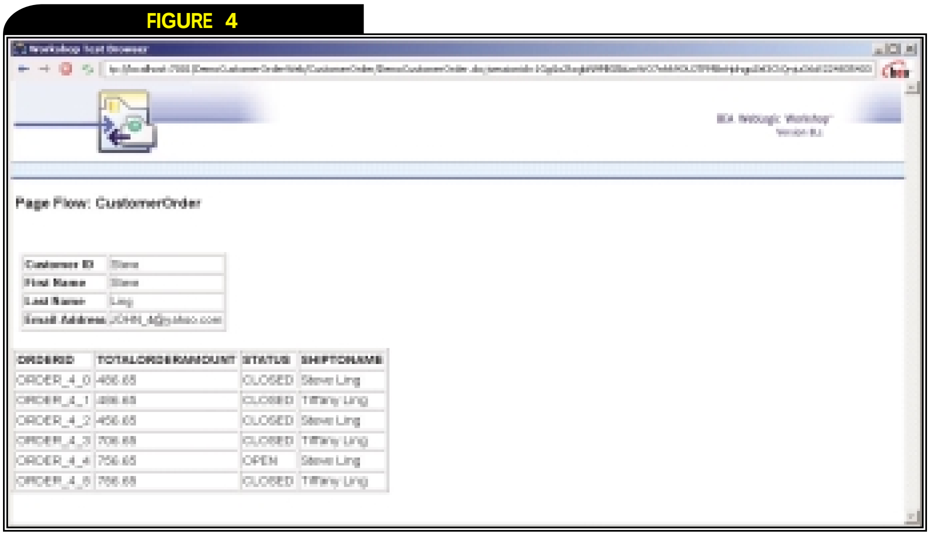
Listing 1 shows the detailed structure of the XQuery generated by Liquid Data for WebLogic 8.1 as a result of Figure 1's mappings. The topmost FOR clause in the XQuery binds a variable to each Avitek customer who satisfies the query's outer WHERE predicate, which only keeps those customers whose customer ID matches the given query parameter. The outermost RETURN clause specifies the query's output, which consists of customer data from the RTL-Customer database plus the result of a nested query. The nested query pulls order data from the Electronics Order Web Service. The nested query's WHERE clause ensures that only orders for the customer of interest are queried, and the inner RETURN clause retrieves the desired order information. Listing 2 shows a sample of this query's output.

Now that we have designed and tested the query successfully, we are ready to deploy the query and its associated target schema to the Liquid Data server as a stored query. Deploying a stored query directly from DVB is a new capability in Liquid Data 8.1. Once deployed, the query becomes immediately available for use by applications using the Liquid Data control in the WebLogic Workshop IDE, so we are now ready to create a Liquid Data Java control.

Creating a Liquid Data Control

Having deployed the Avitek customer order query using DVB, we are now ready to launch the BEA WebLogic Workshop IDE and start creating the desired application using the Liquid Data for WebLogic 8.1 control. To create an instance of a Liquid Data control, we will begin by creating a new default application in WebLogic Workshop and creating a folder called orders within it. We will then create a new control by selecting Liquid Data from the controls selection presented to us when we ask WebLogic

FIGURE 4



Page Flow: CustomerOrder

Customer ID	Steve
First Name	Steve
Last Name	Ling
Email Address	JOHN_A@yahoo.com

ORDERID	TOTALORDERAMOUNT	STATUS	SHIPTONAME
ORDER_4_0	456.65	CLOSED	Steve Ling
ORDER_4_1	[illegible]	CLOSED	Tiffany Ling
ORDER_4_2	456.65	CLOSED	Steve Ling
ORDER_4_3	706.65	CLOSED	Tiffany Ling
ORDER_4_4	756.65	OPEN	Steve Ling
ORDER_4_5	756.65	CLOSED	Tiffany Ling

Output from running the Web application

Workshop to create a new Java control. Having done this, clicking on the next button will take us to a second screen that asks us to choose between a local and a remote Liquid Data server. Liquid Data supports two deployment scenarios. It can either be colocated in the same server as the final application, or it can be on a remote. In this example we will select the local option and click Create. This will take us to a third screen in the Liquid Data control creation process, which allows us to browse the deployed stored queries in the repository of the Liquid Data server. At this point we can select one or more queries to become methods when we create the Liquid Data control. Figure 2 shows the dialog for choosing among a set of available Liquid Data queries.

We can now select the query crm.CustOrder and press the Finish button; the Java source code for defining the desired Liquid Data control instance will be created as the file CustOrder.jcx. As alluded to earlier, creating this control also auto-generates the XMLBeans classes related to the target schema associated with the query. These XMLBeans classes provide typed accessor methods for traversing the XML data returned by the query. Listing 3 (due to space limitations, Listing 3 is online at (www.sys-con.com/weblogic/source.cfm/) shows the contents of the Liquid Data control JCX file. Given that we selected just our one query when creating the control, this file defines just one method, one that corresponds to the query that we selected. The name of the method defaults to the name of the query, although developers have the option of changing the method name. Note that this method, CustomerOrder(), takes one parameter, Customer ID (the input parameter for the stored query CustomerOrder), and returns a result of type CustOrderDocument (the root document type for the collection of XMLBeans classes generated from the query's target XML schema). The method annotations in the JCX file include the name of the stored query and a copy of its XML schema.

Once this Liquid Data Java control instance has been created, it can be used in developing WebLogic Workshop applications such as page flows, portals, Web services, and workflows.

Creating Web Applications

BEA WebLogic Workshop 8.1 provides tools for developing Web applications using JavaServer Pages (JSPs) and page flows, separating presentation, business logic, and navigational control logic to help minimize application complexity. Page flows in WebLogic Workshop allow developers to separate their user interface code for a Web application from their navigational control logic and their other business logic (which may include data aggregation code). User interface code is placed where it belongs, in JSP files. Navigational control logic is implemented separately (but easily) in the page flow's controller file (which is a special Java file that uses a JPF file extension). Data aggregation logic can be implemented in a Liquid Data control. In this section we will walk through the process of creating a Web application that uses our Liquid Data control for data aggregation and page flows for data presentation.

Given a Liquid Data control, right-clicking on the control's JCX file causes WebLogic Workshop to prompt for one of two options: generate page flow or generate test JWS (Java Web service) file. If we select the option to generate a page flow, WebLogic Workshop will launch its page flow–generation wizard to guide us through the creation of a page flow based on our Liquid Data control. The default page flow that it generates includes JSPs that allow users to enter values for the input parameters for the control's methods and then to invoke the methods. The generated JSPs don't take care of result formatting, however; they simply display the returned results in XML string format. In this example, we want fancier formatting – we would like our result page to have a header to show the Customer profile data along with a table listing all the order information for the given customer.

To format the query's XML results as desired in the JSP result page, we will need to declare a public variable of type Customer in the page flow so that it will be accessible in the WebLogic Workshop page flow properties data palette. After declaring this variable and assigning the results of the query to it in the source code, we need to create a table for formatting header information. In the header block we'll simply create a table with two columns and four rows. The first column will have the labels for the Customer information and the second column will have the actual values. These values can be dragged and dropped into the table from the Customer object in the properties data palette. As shown in Figure 3, for the Order Information table we will need to drag-and-drop the CustomerOrderArray element from the Customer object in the properties data palette. As we drop the CustomerOrderArray onto the JSP, a RepeaterWizard will pop up and walk us through the creation of the orders table. The RepeaterWizard will provide us with choices, including the selection of what fields we want to display and the formatting style to be used. At this point we have completed the formatting of our JSP pages and are ready to test our application. Figure 4 shows the output of running the resulting Web application. It shows the results formatted with a header block containing the customer profile information and a detail block containing all of the orders related to the customer. It is interesting to note how little actual coding was required to create the desired customer self-service application.

Once we've created a Liquid Data control, it's just as easy to create a portal, a Web service, or a workflow that incorpo-

rates it. The paradigm for consuming Liquid Data queries in any other WebLogic Workshop project is the same as what we've shown for creating Web applications.

Other Notable Liquid Data Features

In addition to the Liquid Data control, which is the central new feature in Liquid Data for WebLogic 8.1, this release added several new ways for XML data to enter the Liquid Data world and improved on manageability and performance in several key areas.

Delimited Files

Liquid Data 1.0 supported only XML files when it came to file data sources. BEA Liquid Data for WebLogic 8.1 adds support for delimited files, which are text files that contain a predefined character as a data separator. Delimited files are typically exported from a database, a spreadsheet, or another application like a spreadsheet. It's now possible to create delimited file data sources in Liquid Data for use in queries. Such data sources have associated (partially implied) XML schemas and they make it possible to combine delimited file data with any other Liquid Data data source(s) by creating queries that use the delimited file data source.

Complex Parameter Types

Complex parameter types (CPTs) are another addition. Essentially, CPTs make it possible to use XML data as an input parameter to a Liquid Data query. CPTs make it possible to define an XML data stream of an arbitrary (XML Schema) type and then pass XML data of that type as input to a query. A typical CPT use case might be a workflow that receives an XML message and needs to enrich the message with information culled from a collection of back-end heterogeneous data sources.

SQL Calls as Data Sources

BEA Liquid Data for WebLogic 8.1 also includes support for using SQL stored procedures and SQL queries as parameterized data sources for use in Liquid Data queries and views. To use a stored procedure or a SQL query as a data source in Liquid Data, one creates a SQL Call Description File (SCDF). The SCDF is an XML file that defines the types and the functions for a set of SQL stored procedures and/or SQL queries. These SQL calls then become available for use in building queries in the Data View Builder just like any other data source.

Summary

In this article, we looked at how BEA Liquid Data for WebLogic 8.1 makes it easy to develop applications – including Web applications, portals, Web services, and workflows – that require access to data from multiple heterogeneous sources. You can try developing applications using BEA Liquid Data for WebLogic 8.1 yourself by downloading it from http://commerce.bea.com/index.jsp.

References

- Carey, M.; and Mangtani, N. "Liquid Data: XQuery-Based Enterprise Information Integration." *BEA WebLogic Developer's Journal*, Vol. 2, issue 4, April 2003.
- Carey, M.; Florescu, D; and Mangtani, N. "Integrating Enterprise Information on Demand with XQuery." *XML-Journal*, Vol. 2, issues 6 and 7.
- *BEA Liquid Data 8.1 Documentation:* http://edocs/liquiddata/docs81/index.html
- *XMLBeans: An Easy Way to use XML in Java:* http://dev2dev.bea.com/technologies/xmlbeans/overview.jsp.

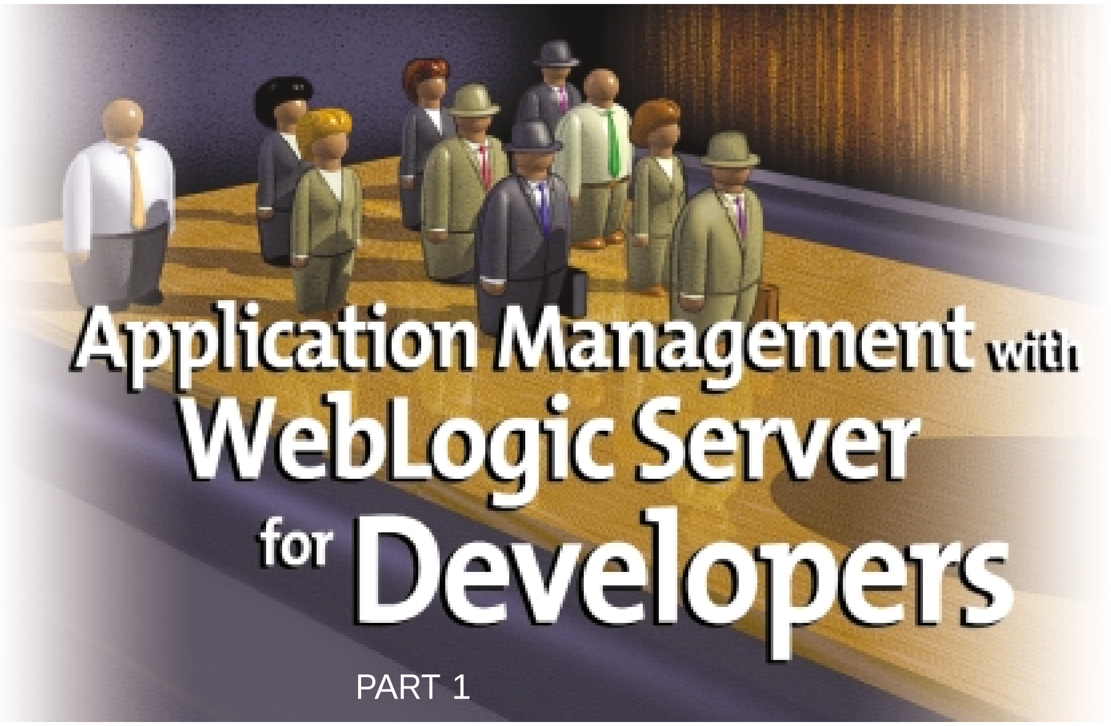
Listing 1: Generated by Data View Builder 8.1

```
namespace custorder = "http://www.avitek.com/custorder"
namespace elecOrd = "java:examples.ldi.webservice"
namespace Elect =
"http://www.bea.com//examples/ldi/webservice/customerOrder"

<custorder:CustProfile>
  {
  for $cust in document("RTL-CUSTOMER")/db/CUSTOMER
  where ($#custID of type xs:string  eq  $cust/CUSTOMER_ID)
  return
  <CUSTOMER>
    <CUSTOMER_ID>{ xf:data($cust/CUSTOMER_ID) }</CUSTOMER_ID>
    <FIRST_NAME>{ xf:data($cust/FIRST_NAME) }</FIRST_NAME>
    <LAST_NAME>{ xf:data($cust/LAST_NAME) }</LAST_NAME>
    <EMAIL_ADDRESS>{ xf:data($cust/EMAIL_ADDRESS) }</EMAIL_ADDRESS>
    {
    for $order in Elect:getOrderSummary($cust/CUSTOMER_ID)/result/*
    return
    <CUSTOMER_ORDER>
      <ORDER_ID>{ xf:data($order/elecOrd:orderID) }</ORDER_ID>
      <SHIP_TO_NAME>{ xf:data($order/elecOrd:shipToName)
}</SHIP_TO_NAME>
      <ORDER_AMOUNT>{ xf:data($order/elecOrd:totalOrderAmount)
}</ORDER_AMOUNT>
      <STATUS>{ xf:data($order/elecOrd:status) }</STATUS>
      </CUSTOMER_ORDER>
    }
  </CUSTOMER>
  }
</custorder:CustProfile>
```

Listing 2: Results Listing

```
<prefix1:CustProfile xmlns:prefix1="http://www.avitek.com/custorder">
  <CUSTOMER>
    <CUSTOMER_ID>Steve</CUSTOMER_ID>
    <FIRST_NAME>Steve</FIRST_NAME>
    <LAST_NAME>Ling</LAST_NAME>
    <EMAIL_ADDRESS>JOHN_4@yahoo.com</EMAIL_ADDRESS>
    <CUSTOMER_ORDER>
      <ORDER_ID>ORDER_4_0</ORDER_ID>
      <SHIP_TO_NAME>Steve Ling</SHIP_TO_NAME>
      <ORDER_AMOUNT>486.65</ORDER_AMOUNT>
      <STATUS>CLOSED</STATUS>
    </CUSTOMER_ORDER>
    <CUSTOMER_ORDER>
      <ORDER_ID>ORDER_4_1</ORDER_ID>
      <SHIP_TO_NAME>Tiffany Ling</SHIP_TO_NAME>
      <ORDER_AMOUNT>486.65</ORDER_AMOUNT>
      <STATUS>CLOSED</STATUS>
    </CUSTOMER_ORDER>
    <CUSTOMER_ORDER>
      <ORDER_ID>ORDER_4_2</ORDER_ID>
      <SHIP_TO_NAME>Steve Ling</SHIP_TO_NAME>
      <ORDER_AMOUNT>456.65</ORDER_AMOUNT>
      <STATUS>CLOSED</STATUS>
    </CUSTOMER_ORDER>
    <CUSTOMER_ORDER>
      <ORDER_ID>ORDER_4_3</ORDER_ID>
      <SHIP_TO_NAME>Tiffany Ling</SHIP_TO_NAME>
      <ORDER_AMOUNT>706.65</ORDER_AMOUNT>
      <STATUS>CLOSED</STATUS>
    </CUSTOMER_ORDER>
    <CUSTOMER_ORDER>
      <ORDER_ID>ORDER_4_4</ORDER_ID>
      <SHIP_TO_NAME>Steve Ling</SHIP_TO_NAME>
      <ORDER_AMOUNT>756.65</ORDER_AMOUNT>
      <STATUS>OPEN</STATUS>
    </CUSTOMER_ORDER>
    <CUSTOMER_ORDER>
      <ORDER_ID>ORDER_4_5</ORDER_ID>
      <SHIP_TO_NAME>Tiffany Ling</SHIP_TO_NAME>
      <ORDER_AMOUNT>756.65</ORDER_AMOUNT>
      <STATUS>CLOSED</STATUS>
    </CUSTOMER_ORDER>
  </CUSTOMER>
</prefix1:CustProfile>
```



Application Management with WebLogic Server for Developers

PART 1

BASIC CONCEPTS AND CONFIGURATION TOOLS

BY **VADIM ROSENBERG & ROBERT PATRICK**

AUTHOR BIOS...

Vadim Rosenberg is the product marketing manager for BEA WebLogic Server. Before joining BEA, Vadim spent 13 years in business software engineering, most recently at Compaq Computers (Tandem Division) developing a fault-tolerant and highly scalable J2EE framework.

Robert Patrick is a director of technology in BEA's Office of the CTO and coauthor of the book *Mastering BEA WebLogic Server: Best Practices for Building and Deploying J2EE Applications*. He has spent his career helping customers design, build, and deploy high performance, fault-tolerant, mission-critical distributed systems using BEA Tuxedo and BEA WebLogic Server.

CONTACT...

vadimr@bea.com
robert.patrick@.com

REPRODUCED WITH PERMISSION FROM BEA SYSTEMS.

You're a developer, not an administrator. You think you don't care about system and application administration. Well, you should... and this series of articles on WebLogic Server administration and management for developers will explain why, and how.

Before you can get too far developing a J2EE application with BEA WebLogic Server 8.1, you have to do some administration work to configure a server and deploy the application. Once you're ready to promote the application from your local development environment to another test or production environment, you need to understand what administrative tasks are necessary to prepare the other environment for your application. At some point during the application's life, you'll probably be asked to help troubleshoot some sort of problem with the application. This might require you to monitor the server's and/or application's behavior. You might even be asked to write some custom scripts or programs to automate some of these administrative or monitoring tasks. All of these tasks require some knowledge of system and application administration.

The BEA WebLogic Server administration and management subsystem is based on Java Management Extensions (JMX) version 1.0. So, what is JMX? JMX is the Sun Microsystems, Inc. specification that provides open and extensible Java management services. Even though JMX does not become an official part of J2EE until the release of the J2EE 1.4 specification, it has proven its value to enterprise application infrastructure, and most Java application server vendors today support JMX to some extent. BEA has supported JMX since the WebLogic Server 6.0 release.

WebLogic Server also supports the industry-standard Simple Network Management Protocol (SNMP) version 1 and version 2. SNMP support provides the ability to use the most popular enterprise system and application monitoring products, like HP OpenView, BMC Patrol, or CA Unicenter, to monitor WebLogic Server–based applications. WebLogic Server's SNMP interface is read-only and will not be covered here.

On top of the JMX infrastructure, WebLogic Server provides three major types of interfaces that you can use to achieve your management goals:

- ***GUI tools:*** A set of graphical tools that provide visual, easy-to-use access to most of the server's and application's configuration and runtime information. These tools include the WebLogic Server Administration Console, Configuration Wizard, and WebLogic Builder.
- ***Command-line tools:*** A set of simple, yet powerful, Java command-line programs that provides specialized commands to perform common tasks, as well as commands to manipulate the underlying JMX interfaces. The primary command-line tool is weblogic.Admin, which supports both single command and batch modes for executing administrative commands.
- ***Programmatic APIs:*** A set of APIs and Ant tasks that give you access to the JMX inter-

faces upon which the management infrastructure is based. These interfaces include the loosely typed, standard JMX interface, a strongly typed, easy-to-use interface to the JMX objects, and a set of Ant tasks that allow you to configure, deploy, and start and stop servers.

In this article, we'll talk about ways of managing BEA WebLogic Server that do not require an understanding of the JMX infrastructure, which will be covered in subsequent articles. We start with an overview of the graphical administration tools and then move on to discuss uses of the command-line tools that do not require an understanding of JMX. Before we dive into WebLogic Server administration, let's define some of the major concepts upon which WebLogic Server administration is based.

Administration Concepts and Terminology

BEA WebLogic Server uses the term *domain* to describe a logically related group of WebLogic Server resources that you manage as a unit. It's important to understand that a domain is a purely administrative entity, and logically can cover more then a single application, or even a portion of an application. A domain always includes at least one WebLogic Server instance called an Administration (or Admin) Server. The Admin Server serves as the central configuration repository and central point of contact for administration. A domain may also include additional WebLogic Server instances, called *managed servers*; and clusters, each of which includes one or more managed servers. A cluster can span one or more physical machines.

WebLogic Server also supports the use of a node manager. The *node manager* is a cross-domain entity that provides support for remotely starting servers, restarting failed servers, and monitoring server health. As such, you need only one node manager instance per physical machine. We'll take a closer look at the node manager later. Figure 1 shows one example of a WebLogic Server domain topology.

In addition to facilitating the domain's administration and management, the Admin Server also has at least one application deployed on it: a J2EE application that delivers the WebLogic Server Administration Console to your browser.

Graphical Tools: Configuration Wizard, Admin Console, and Builder

In this section, we'll take you on a tour of the WebLogic Server GUI tools from the perspective of getting your application packaged, and your BEA WebLogic Server 8.1 system and application environments configured and ready for application deployment.

Application Environment Configuration

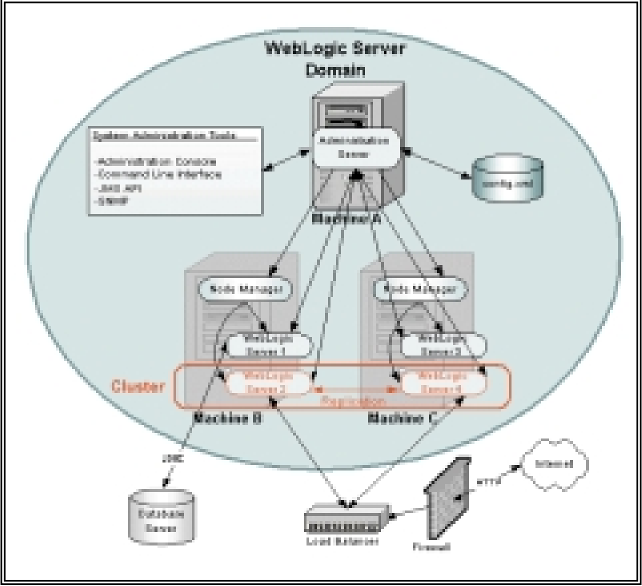
The first thing you need to do after installing the BEA WebLogic Server 8.1 software is to set up a domain and start your Admin Server. As we mentioned earlier, one of the responsibilities of the Admin Server is to act as the configuration repository for the domain. This configuration information primarily resides within an XML file called config.xml. In order to start the Admin Server and access the Admin Console to configure the rest of your domain, you need a base config.xml file (and some other files) that reflects your environment and has enough information to start the Admin Server. WebLogic Server provides the Configuration (Config, for short) Wizard that walks you through this process.

The Config Wizard is a stand-alone Java Swing application that comes with WebLogic Server. Since it is primarily a bootstrapping tool, it does not require the Admin Server to be running in order for you to use it to create the base configuration that your domain requires. You can choose to configure additional information about your domain such as topology information (e.g., clusters, managed servers, and machines), JDBC and JMS resources, and security (e.g., users, groups, and roles).

As in any other wizard-based tool, the Config Wizard walks you through a sequence of data entry screens that not only take your input but also verify accuracy throughout the configuration process. You can go forward and backward, change your decisions, and fix any errors encountered during the process. Figure 2 shows one of the Config Wizard screens.

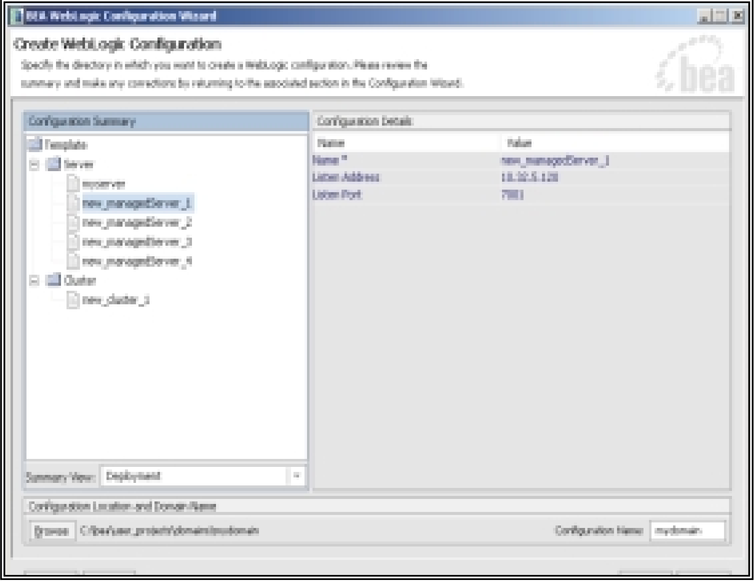
Once you've completed the configuration process, the Config Wizard creates all the necessary directories and files needed to start your Admin Server. At this point, you

FIGURE 1



WebLogic Server domain

FIGURE 2



Create WebLogic Configuration

Specify the directory in which you want to create a WebLogic configuration. Please review the summary and make any corrections by returning to the associated section in the Configuration Wizard.

Name	Value
Name *	new_managedServer_1
Listen Address	10.32.5.128
Listen Port	7001

Summary View: Deployment

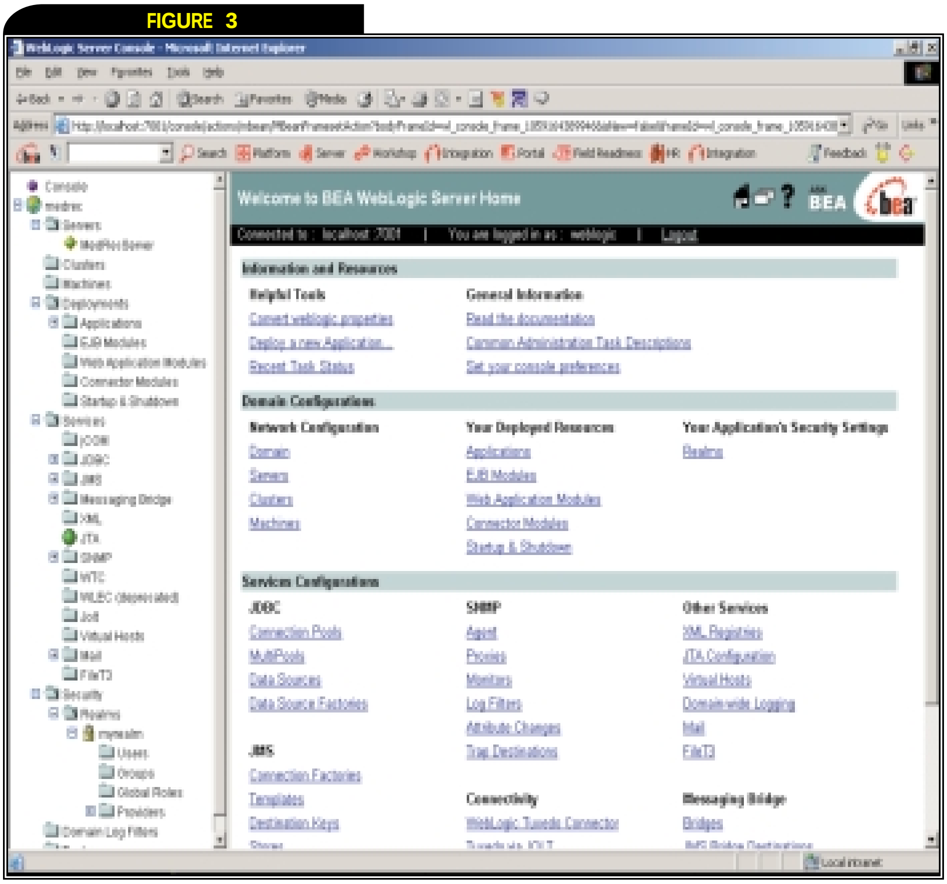
Configuration Location and Domain Name

Browse C:\bea\user_projects\domains\mydomain

Configuration Name: mydomain

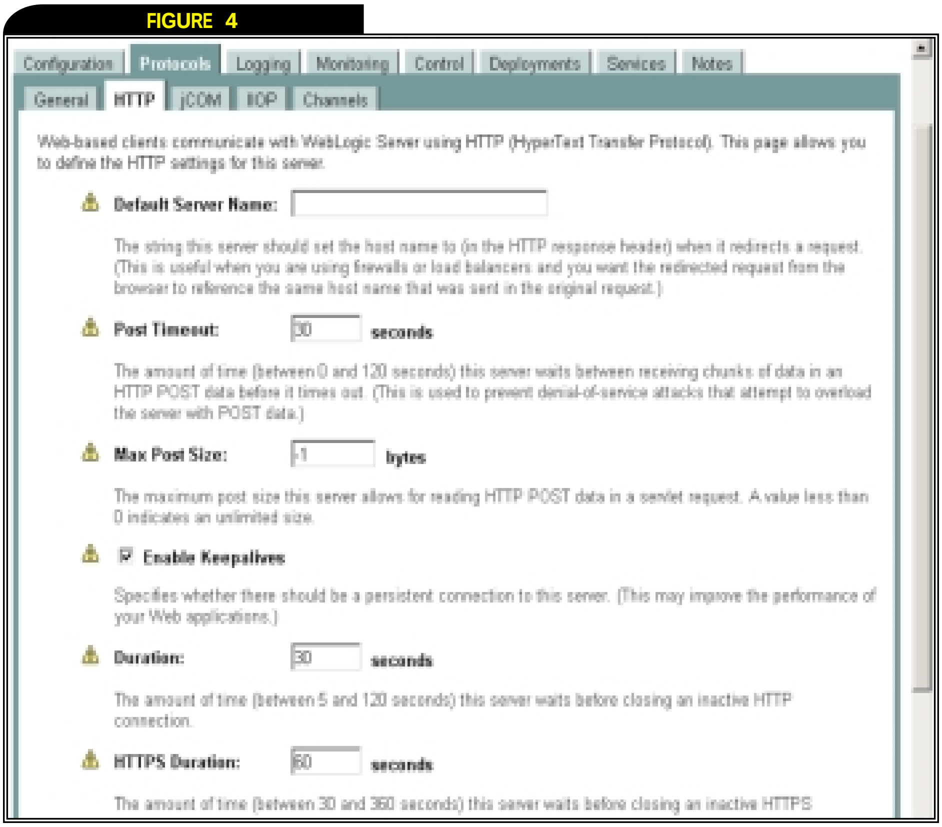
WebLogic Server Configuration Wizard

FIGURE 3



WebLogic Server Administration Console

FIGURE 4



Web-based clients communicate with WebLogic Server using HTTP (HyperText Transfer Protocol). This page allows you to define the HTTP settings for this server.

Default Server Name:

The string this server should set the host name to (in the HTTP response header) when it redirects a request. (This is useful when you are using firewalls or load balancers and you want the redirected request from the browser to reference the same host name that was sent in the original request.)

Post Timeout: 30 seconds

The amount of time (between 0 and 120 seconds) this server waits between receiving chunks of data in an HTTP POST data before it times out. (This is used to prevent denial-of-service attacks that attempt to overload the server with POST data.)

Max Post Size: -1 bytes

The maximum post size this server allows for reading HTTP POST data in a servlet request. A value less than 0 indicates an unlimited size.

☑ Enable Keepalives

Specifies whether there should be a persistent connection to this server. (This may improve the performance of your Web applications.)

Duration: 30 seconds

The amount of time (between 5 and 120 seconds) this server waits before closing an inactive HTTP connection.

HTTPS Duration: 60 seconds

The amount of time (between 30 and 360 seconds) this server waits before closing an inactive HTTPS

HTTP Protocol Configuration Screen

may need to do additional configuration through the Admin Console or, depending on your application requirements, you may be ready to deploy your J2EE application.

Once you've completed your domain's entire configuration, the WebLogic Server Configuration Template Builder allows you to turn an existing domain into a template that can be distributed and replayed to create equivalent configurations in other environments. This can be a big help when trying to promote applications from development to QA, user acceptance testing, and production.

System Configuration with the Administration Console

Once you start the Admin Server, you can access the Admin Console as you would any other Web application running on the server at the configured port. By default, the server uses port 7001, so the URL to access the console from the machine where the Admin Server is running is http://localhost:7001/console. Figure 3 shows the initial view of the Admin Console after logging in, with the Medical Records sample application that comes with BEA WebLogic Server 8.1 deployed. For example, some of the nodes in the object tree on the left have been expanded.

The Admin Console provides a graphical, intuitive, and comprehensive interface to WebLogic Server administration. As a developer, you may be more or less concerned about some of these capabilities; however, it is important that you are at least aware of what the Admin Console has to offer.

One of the functional areas that the Admin Console provides is general server configuration. There are parameters for things like server restart attributes, memory and thread settings, listen addresses and ports, settings for the different protocols that WebLogic Server supports (e.g., IIOP, HTTP, COM), logging, and many others. As a developer, you may not need to change many of these settings. However, they may become important as your application moves from development into QA, performance and load testing, and production. Figure 4 shows the server configuration screen for HTTP protocol configuration.

BEA WebLogic Server provides a set of services for your application, many of which probably contributed to the decision to use an application server in the first place. These services include J2EE and non-J2EE resources. Typically, you can't deploy your application unless you have all the

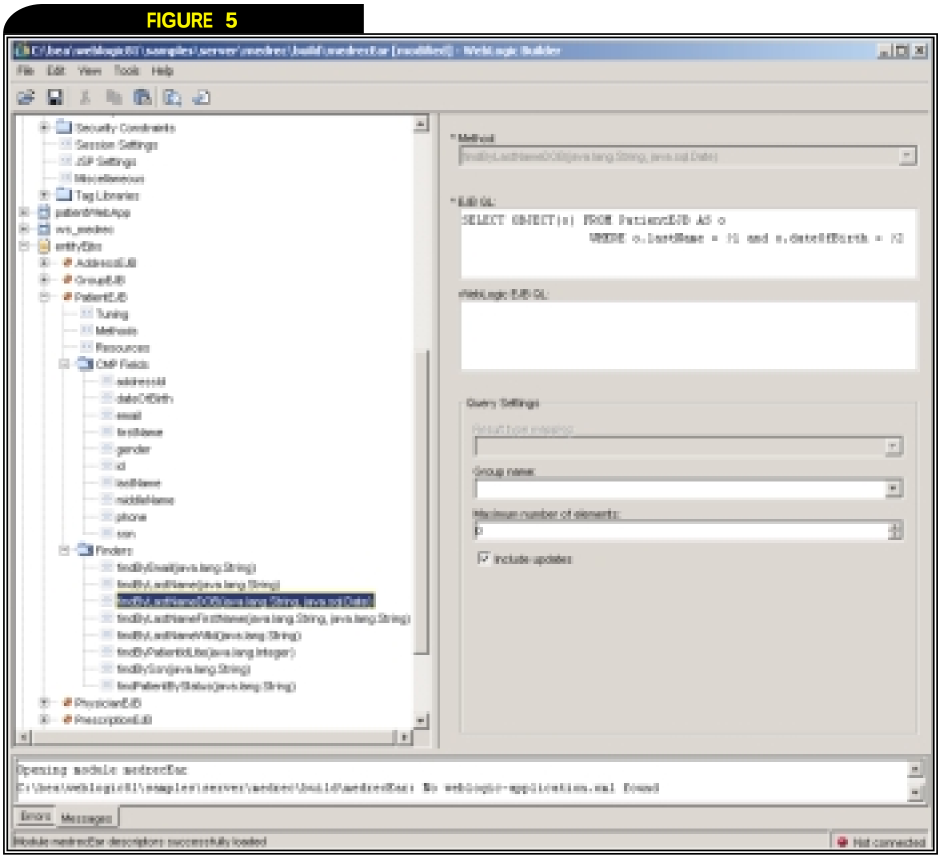
J2EE resources it requires already configured and available in the server. These resources include database connectivity (JDBC connection pools and data sources), asynchronous messaging support (JMS), e-mail connectivity (JavaMail), integration resource adaptors (JCA), and others. Other services need to be configured properly in order for your application to function correctly. These include things like transaction support (JTA), security, XML, and others. The Admin Console provides a convenient and intuitive view of all these services as a tree of objects that you can configure, manage, and monitor.

In addition to these application services, WebLogic Server provides a set of services targeted at providing high performance, scalability, fault tolerance, and reliability to your application. These services consist of things like replication groups for controlling the in-memory replication failover, different kinds of caching for performance and resource efficiency, object pools, thread pools, and so on. Through the Admin Console and the J2EE deployment descriptors, WebLogic Server gives you the ability to configure and fine-tune many of these services to optimize your applications' performance, availability, and reliability. For example, one of the most important configuration parameters for improving performance is the execute queue thread count. When a request is received by a server, the server places it in an execute queue. This request is then assigned to an execute thread that processes the request and returns the response to the requestor. In addition to the default queues for processing requests, WebLogic Server provides you with the ability to configure new execute queues, each with their own set of execute threads, and assign them to either entire applications or pieces of an application. As we will see later, the Admin Console provides powerful monitoring features that can help you understand and optimize your application's performance.

J2EE Application Configuration with WebLogic Builder

In addition to configuring different server-provided services, you need to configure and package your application before deployment so it can use all the server-provided services correctly. According to the J2EE specification, your application-specific configuration is stored in XML configuration files called deployment descriptors (DD). While you can create and edit DD by hand using any text editor, you will find this to be a very tedious and error-prone process. WebLogic Builder – a graphical J2EE application packaging and deployment tool – is a great help in this process.

FIGURE 5


C:\bea\weblogic81\samples\server\medrec\build\medrecEar [modified] - WebLogic Builder
File Edit View Tools Help
Security Constraints
Session Settings
JSP Settings
Miscellaneous
Tag Libraries
patientWebApp
ws_medrec
entityEjbs
AddressEJB
GroupEJB
PatientEJB
Tuning
Methods
Resources
CMP Fields
addressId
dateOfBirth
email
firstName
gender
id
lastName
middleName
phone
ssn
Finders
findByEmail(java.lang.String)
findByLastName(java.lang.String)
findByLastNameDOB(java.lang.String, java.sql.Date)
findByLastNameFirstName(java.lang.String, java.lang.String)
findByLastNameWild(java.lang.String)
findByPatientIdLike(java.lang.Integer)
findBySsn(java.lang.String)
findPatientsByStatus(java.lang.String)
PhysicianEJB
PrescriptionEJB
Query Settings
Result type mapping
Errors
Messages

* Method: findByLastNameDOB(java.lang.String, java.sql.Date)

* EJB QL:

```
SELECT OBJECT(o) FROM PatientEJB AS o
                 WHERE o.lastName = ?1 and o.dateOfBirth = ?2
```

WebLogic EJB QL:

Group name:

Maximum number of elements: 0

☑ Include updates

```
Opening module medrecEar
C:\bea\weblogic81\samples\server\medrec\build\medrecEar: No weblogic-application.xml found
```

Module medrecEar descriptors successfully loaded | Not connected

MedRec application archive configuration with WebLogic Builder

WebLogic Builder (see Figure 5) is a stand-alone Java Swing–based tool that comes with BEA WebLogic Server. It allows you to create and/or modify the application's configuration stored in both the J2EE-standard and WebLogic Server–specific deployment descriptors. While this tool does not require WebLogic Server to be running, it can validate many of the application configuration settings against a running instance of WebLogic Server if one is available. After completing your configuration, WebLogic Builder will package your J2EE application into the appropriate JAR, WAR, and/or EAR files; and even deploy it directly to the WebLogic Server instance of your choice.

You might notice that there is some overlap in configuration features between the Config Wizard, the Admin Console, and WebLogic Builder. Each of these tools focuses on a specific audience and task. WebLogic Builder is primarily intended to allow application developers or deployers to create and modify application deployment descriptors and package your J2EE applications for deployment to BEA WebLogic Server. The WebLogic Server Config Wizard is primarily intended to provide a domain configuration bootstrapping mechanism to create the files necessary to start your Admin Server. WebLogic Server's Admin Console is the primary configuration, deployment, and monitoring tool used for all other tasks. As a developer, it's beneficial to have at least some understanding of each of these tools.

Summary

This article introduced you to the major concepts and terminology for a BEA WebLogic Server domain. Then we showed you the most commonly used graphical tools for setting up WebLogic Server, and packaging and configuring applications to be deployed on it: the Administration Console, the Configuration Wizard, and WebLogic Builder.

In our next article, we'll look at application deployment, run-time management, and monitoring facilities available with WebLogic Server – both graphical and command line–based.

Data Views in Liquid Data

AN ENTERPRISE-LEVEL TOOL THAT OFFERS VIEWS FROM MANY SOURCES

BY **KUNAL MITTAL**

AUTHOR BIOS...

Kunal Mittal is a solutions engineer at Wakesoft, Inc., and a consultant for implementation and strategy for Web services and service-oriented architectures. He has coauthored and contributed to several books on J2EE, WebLogic, and Web services.

CONTACT...

kunal@kunalmittal.com

BEA Liquid Data for WebLogic provides a unified view of data aggregated from multiple resources such as databases, XML files, Web services, EJBs, or Java 2 Connector Architecture (J2CA) adapters. At a very high level, Liquid Data can be thought of as an enterprise-wide JOIN tool that provides an XML view of data from any source. This data can be accessed using evolving technologies such as XQuery.

Liquid Data provides several different mechanisms through which Data Views in Liquid Data can be leveraged by the rest of the BEA Platform. In this article I'll introduce several ways of accessing Data Views outside of the Liquid Data for WebLogic Data View Builder itself.

Quick Introduction to Data Views

Data Views are rich data sources built using XQuery technologies to aggregate data from discrete sources using a single query. BEA Liquid Data for WebLogic provides a rich graphical interface, called the Data View Builder, through which users can set up the data sources they need to access, set up security and authentication information, and then define various queries that they need to execute. The Data View Builder allows users to define XML Schemas for the individual data sources or the resulting data that they get from the execution of the query.

Data Views can aggregate data from various sources such as relational databases (using standard SQL or stored procedures), XML, or delimited files stored in a file system; Web services; and legacy EIS. Using the Data View Builder, one or more of these sources' schemas are aggregated to form the target schema. Users can define JOIN conditions, and other data aggregation or selection logic that you are used to doing while accessing relational data using SQL. Users also have the ability to design custom functions to access data in different way.

This article assumes that you have already created some Data Views. I'll focus on the various mechanisms available to access them.

Accessing Data Views

Data Views in Liquid Data are available through various access mechanisms to the rest of the BEA Platform. Any Java program with the appropriate security credentials, and that can access the Java Naming and Directory Interface (JNDI), can issue queries against a Liquid Data server. The different types of Java programs include:

- Stand-alone Java applications
- Java servlets
- JavaServer Pages (JSPs)
- JavaBeans
- Enterprise JavaBeans (EJBs)
- Web services
- Business process management (BPM) components or workflows in BEA WebLogic Integration
- Portlets in BEA WebLogic Portal

Access to Data Views Through EJB Clients

A stateless session EJB exposes the query API, which delegates queries through a query processor to the underlying data stores. Any Java program that can access EJBs can use this EJB-based API. Stand-alone Java applications, Javabeans,

and other EJBs all use this API, and collectively are called EJB clients. These programs can use the EJB remote or local interfaces directly.

EJB clients are any applications that invoke queries on the Liquid Data server using the Liquid Data EJB API. All Java clients can leverage the flexibility and data integration properties offered by XQuery to meet their data access needs. All of these clients access the EJB remote interfaces directly; therefore, they can be collectively characterized as EJB clients. The Data View builder that is part of the core Liquid Data product is a special kind of EJB client.

The four basic steps to access Data Views through this mechanism are:

- ***Connect and authenticate with a Liquid Data server***
  JNDI is used to look up the EJBs and standard EJB calls are used to connect to a Liquid Data server. Once connected, queries can be executed anonymously or through authentication itself by passing a security context to the corresponding JNDI environment properties SECURITY_PRINCIPAL and SECURITY_CREDENTIALS.

- ***Specify parameters for the search query***
  In most cases users would want to execute queries based on certain conditions. XQuery-based parameters are used to pass search conditions to a Liquid Data server. The following code snippet is an excerpt of how to pass search parameters to a data view.

```
import com.bea.ldi.server.common.QueryParameters;
...
QueryParameters qp = new QueryParameters();
qp.setString("title", "WebLogic Developers
Journal");
```

- ***Execute the query on the Liquid Data server***
  There are different mechanisms to execute the query based on the type of query being executed. If an ad hoc query is being executed you would use

```
execute (queryString, [qp]) command
```

  If the query is a stored, named query, you would use

```
executeStored (queryName, [qp]) command
```

  The "qp" parameter is optional based on whether you are passing query parameters.

- ***Receive and process the results to the query***
  When the query is executed, the Liquid Data server returns an unformatted XML string. You can use the toXML() or getDocument() methods to receive an XML or DOM format with the results of the query.

Access to Data Views Through JSP

Liquid Data for WebLogic provides a set of tag libraries that can be used by a JSP to access Data Views. The basic tags provided are a "query" tag and a "param" tag.

The "query" tag is used to connect to a Liquid Data server and execute a query. It takes the name of the query, the connection to the server, and the username and password needed to authenticate with the server

The "param" tag is used to pass parameters for the query using name/value pairs. Listing 1 shows some sample code on how to use the "query" and "param" tags.

You can use these tag libraries to execute both ad hoc and stored queries. The resulting XML can be displayed on the JSP using XLST translations. WebLogic server comes with an XSLT JSP Tag Library that can be used for an XSL transformation. Alternatively, you could use a third-party or open source library for this functionality.

Access to Data Views Through Web Services

A stored query in Liquid Data can be exposed as a Web service through the Liquid Data administration console. The URL of the WSDL of a generated Web service has the following pattern:

```
http://host:port/liquiddata/query_name/webservice?WSDL
```

Once you have exposed a Liquid Data query as a Web service, you would use standard Web services mechanisms to look up the WSDL and execute the Web service.

BEA Liquid Data for WebLogic 8.1 also provides a control in WebLogic Workshop that can be used to invoke this query. WebLogic Workshop can also be used to test the Liquid Data Web service–based query.

EJB clients are any applications that invoke queries on the Liquid Data server using the Liquid Data EJB API.

Access to Data Views Through WebLogic Integration BPM Components

BPM components in WebLogic Integration would typically use the stateless session EJB API discussed above to access Liquid Data queries. A business operation in WebLogic Integration would define the JNDI name of the Query EJB to be invoked (com.bea.ldi.server.QueryHome) and the method to invoke on the EJB.

Liquid Data helps BEA WebLogic Integration integrate data from various relational, non-relational, and Web sources. The Liquid Data views are read-only. WebLogic Integration components such as workflows are used to initiate two-way transactions from the rich set of data that is provided using the Data Views in BEA Liquid Data for WebLogic.

Access to Data Views Through Portlets

Portlets and other components in BEA WebLogic Portal can use all the different mechanisms discussed to access Liquid Data queries. Portlets could use Web services or the JSP tag libraries. Other Portal components could use the EJB APIs.

Liquid Data thus becomes a rich data feed and WebLogic Portal becomes the layer that handles the presentation of this rich data.

Summary

This article has briefly introduced the several ways to leverage Data Views in BEA Liquid Data for WebLogic outside of Liquid Data itself. Basically, any piece of Java code that can access an EJB or a Web service can execute queries on a Liquid Data server. Even JSP-based applications can leverage Liquid Data queries directly, without having to go through an EJB tier.

References

- *E-Docs for Liquid Data 1.1:* http://e-docs.bea.com/liquiddata/docs11

Listing 1: The "query" and "param" tags

```
<%@ taglib uri="LDS-taglib.jar"
prefix="ldquery" %>
...
< ldquery:query server="t3://localhost:7001"
username="wldj_user" password="wldj_password">

      < ldquery:param name=" title "
             value="<%=\" WebLogic Developers
Journal \"%>"/>

</ ldquery:query>
```

Simplifying Creation of Managed Service-Oriented Applications

IT'S NEVER TOO EARLY TO THINK OF MANAGEMENT

BY **AMIT DAYAL & DIDIER GUZZONI**

AUTHOR BIO

Amit Dayal is the VP of product management at Confluent Software, a leading provider of Web services management software. Amit has over 14 years of experience in enterprise software. Prior to Confluent, Amit cofounded interKeel, a company that pioneered the Web services management category.

Didier Guzzoni is a senior software engineer at Confluent Software. He previously worked as a research engineer at VerticalNet, where he was involved in the OSM project, an infrastructure for Web services facilitation and orchestration. He started his career as a research engineer at SRI International (Stanford Research Institute) working on the OAA project (distributed software architecture).

CONTACT...

amit@confluentsw.com
didier@confluentsw.com

Controls allow developers to focus on writing application logic and delegating infrastructural issues such as asynchronous messaging, conversations, and connectivity with remote resources. But much complexity still remains to make such applications manageable. In this article, we'll develop a sample Web services–based application using built-in WebLogic Workshop controls and use it to illustrate the challenges of managing service-oriented applications. Finally, we'll show you how Confluent Software offers developers and IT staff a possible Web services management solution that is integrated with BEA WebLogic 8.1 Platform.

Why Be Service-Oriented?

Few applications operate in a vacuum any more. Instead, most applications are constantly being integrated into intricate "networks" of application-to-application interconnections. These application networks are typically quite heterogeneous, linking applications developed by different organizations, running on different platforms, implemented in different languages, and secured differently.

Worse still, with today's tightly coupled integration approach, each application is acutely aware of the peculiarities of every other application. Thus, any change to one application sets off a cascade of changes across the entire network. No wonder development and maintenance costs are going through the roof. At the same time, management wonders if IT still matters, and cuts IT budgets every year – the price one pays for tight coupling!

Service-oriented architectures (SOA) provide a way out. Service-oriented applications bridge heterogeneity by leveraging standards-based Web services technologies. They tackle the change management nightmare by demanding loose coupling between applications, i.e., changes in one application's implementation don't break any other application.

WebLogic Workshop Controls

BEA WebLogic Workshop makes it easy to construct service-oriented applications by providing easy-to-use tools for distributed-component assembly, and by facilitating long-lived conversations of back-and-forth XML messages between components.

In WebLogic Workshop one of the core building blocks for any application is controls. Controls look like simple Java components (with methods and events) whose behavior can be customized via properties. Controls hide the J2EE plumbing issues involved in connecting with remote databases, Web services, EJB objects, and message queues. With controls, connectivity to these external resources can be configured by setting properties instead of API calls. Controls contribute to the service-oriented cause by facilitating easy assembly of loosely coupled application components.

To further enhance loose coupling, WebLogicWorkshop 8.1 also provides visual editors (and XQuery support) for building mappings between XML interfaces and underlying Java objects.

Let's walk through a simple Web services application and get a first-hand look at how WebLogic

Workshop takes the pain out of creating such applications.

Sample Order Management Application

Even implementing a simple business scenario, such as our sample order management Web service, creates a fair amount of technical complexity:

- The Web service must be deployed on a server with its WSDL definition file exposed for client use.
- The service relies upon and uses a variety of back-end components, including databases, EJBs, and other remote Web services.
- The service must enforce the sequence in which operations are called.
- The service must allow multiple clients to invoke it concurrently.

The sample Order Management Web service exposes the following operations:

- A user can check on the status of a pending order:

```
queryOrderStatus(String inAccountNumber,
    String inOrderNumber)
```

- A user can create a new order:

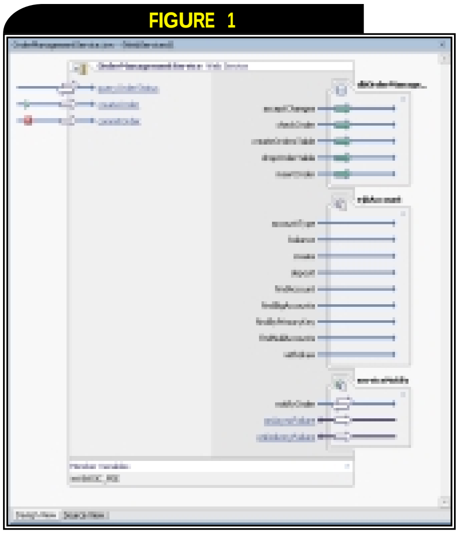
```
createOrder(String inAccountNumber, String
    inOrderNumber)
```

- A user can commit a newly created order:

```
commitOrder(String inAccountNumber, String
    inOrderNumber)
```

First the order is created, and if successful, committed. createOrder and commitOrder must be executed in a sequence. For each call to any of the methods, the service charges a fee to the customer's account. User accounts are modeled as EJBs. Orders are stored in a JDBC-compliant database. commitOrder notifies a remote server (through a Web service invocation) about each newly created order.

FIGURE 1



Sample design view

By using BEA WebLogic Workshop, and leveraging the powerful concept of controls, it is very easy to accommodate the inherent complexity of creating such a Web service. Figure 1 shows the design view of the sample service and the controls it uses.

First, the processes of creating, deploying, and exposing a Web service are straightforward in WebLogic Workshop. Simply create a Web service project, add all the methods, write the logic, and push the "play" button. Voilà! In addition, WebLogic Workshop automatically creates a set of Web pages to test and run the Web service (see Listing 1).

Second, accessing back-end components such as EJBs, databases, or even remote Web services, is a matter of drag-and-drop actions (see Listing 2). Once the component to use is dropped into the project, it appears as an object in the "source view" pane and can be easily integrated into custom code.

Third, making sure that methods are called in a given order is free in WebLogic Workshop. Each operation exposed by a Web service can be tagged as being part of a conversation. The tag also tells which part of the conversation the operation is involved in, thus enforcing the order in which those operations have to be invoked (see Listing 3).

Finally, WebLogic Workshop conversations help to keep contexts for multiple users. Conversation tags indicate that context information (state) is kept for the duration of a conversation. Using this mechanism, the order management service is able to handle multiple clients at the same time.

Service-Oriented Application Deployments Create New Challenges

Unfortunately, there are no free lunches. Service-oriented applications slash development and maintenance costs, but at the cost of increased management complexity:

Monitoring Health of an Application, Not Just Components

Current monitoring tools cannot handle loosely coupled, distributed applications – even relatively simple applications like our sample order management application. Current management tools only monitor individual components and that by monitoring lower-level infrastructure. Based on that we have no way to verify if a distributed Web services application is operating normally. The problem only intensifies if an application uses external Web services, such as a partner's quoting service. The conventional monitoring approach totally breaks down since there is no way to instrument servers operated by external organizations. So how do you detect that a problem is brewing before you're paged by an irate manager?

Troubleshooting Problems Without Knowing Where to Look

Once you starts getting paged every few minutes by an unhappy VP of Sales because the order management application is misbehaving (two days before quarter-end), where should you go to look for the culprit? Is it the WebLogic Workshop Control providing access to the inventory system managed by the Manufacturing IT group? Is it one of the many Web services that this application depends on? Worse still, how do you even tell which Web services, and external resources, the application depends upon? As the paging frequency increases, the only recourse you have is to start going through many logs on many systems.

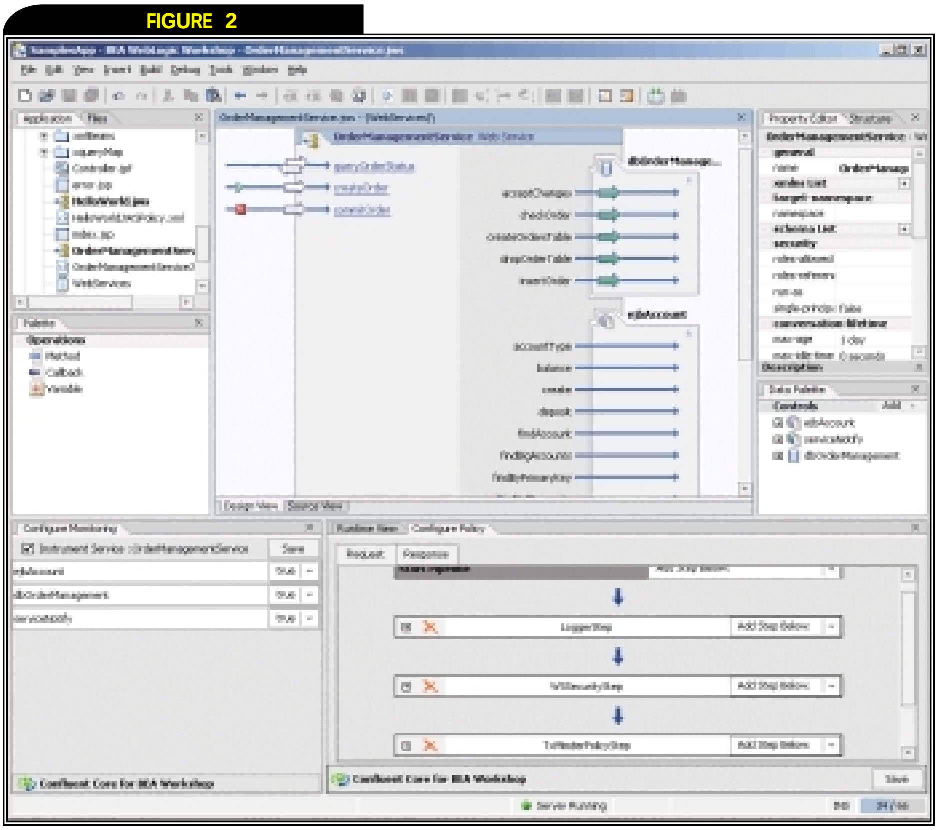
Logging Requests and Responses for Auditing and Billing

Internal IT departments are increasingly being asked to track usage of software applications so that the different lines of business pay their fair share of IT costs. Also, any application that automates a business process across organizational boundaries has to have a plan for settling repudiation disputes between senders and receivers. Consistent message logging, both in the calling application and the invoked application, is key to handling these auditing and billing requirements. How does an IT manager ensure that all developers implement the right level of logging, at all entry and exit points? For messages encrypted using XML encryption, how do you ensure that no logging is done when messages are in the clear (thus defeating the company's privacy policies)? For Web services published natively by packaged applications, how can message logging be accomplished?

Ensuring Security Policies are Consistently Enforced

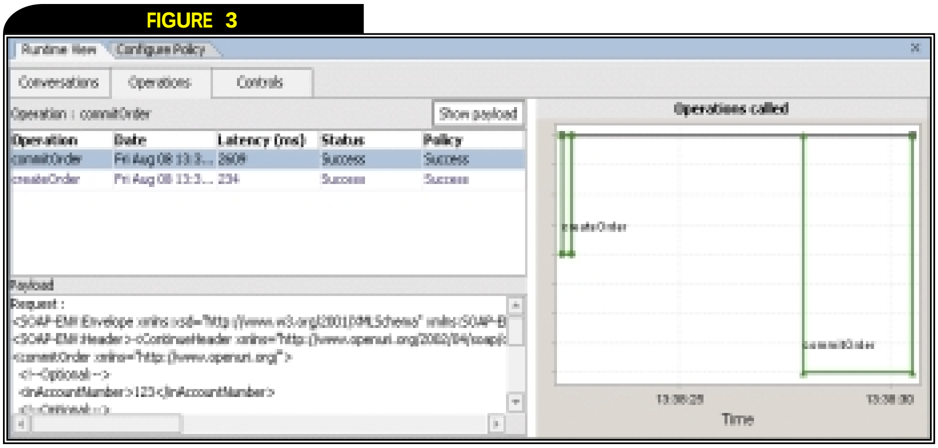
Even in the simplified sample application, a document goes through multiple hops across application components. A client application sends a P.O. when invoking the createOrder operation; this PO is ultimately sent to a different Web service once the order is committed. To ensure end-to-end confidentiality and integrity,

FIGURE 2



Runtime View tab

FIGURE 3



Operation : commitOrder

Operation	Date	Latency (ms)	Status	Policy
commitOrder	Fri Aug 08 13:3...	2609	Success	Success
createOrder	Fri Aug 08 13:3...	234	Success	Success

Payload

```
Request :
<SOAP-ENV:Envelope xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:SOAP-E
<SOAP-ENV:Header><ContinueHeader xmlns="http://www.openuri.org/2002/04/soap/c
<commitOrder xmlns="http://www.openuri.org/">
  <!--Optional-->
  <inAccountNumber>123</inAccountNumber>
  <!--Optional-->
```

Operations tab

developers can use WS-Security, XML encryption, and XML signature (BEA WebLogic 8.1 supports WS-Security). Having this capability is great, but without consistent usage, it doesn't buy much. How does a security architect make certain that all Web services across the enterprise verify the digital signature of a message before processing?

These are just some of the new challenges IT staff are grappling with as they start moving loosely coupled, distributed applications into industrial-scale deployments. For the sake of brevity, we have chosen to highlight only a few key issues here. There are many other issues we will mention in passing – including implementing failover and retry schemes to handle transport-level failures; implementing single sign-on; and service versioning.

Evaluating New Management Tools for Managing SOA Deployments

By now, hopefully we have convinced you that successful deployment of Web services initiatives requires tackling a bunch of daunting management issues. Even if you are still only in the pilot stage with your Web services today, you should not delay thinking about management. The good news is that there are hundreds of management tools to look at. The bad news is that there are hundreds of management tools to look at! There are new Web services management tools, application server management tools, systems management tools, XML routing tools, manager of manager tools, and so on – and the word "Web services" seems to be showing up in all their datasheets.

The following are a few things to look for when evaluating tools for managing the new service-oriented applications you're building.

Does It Nonintrusively Monitor All Invocations to WebLogic Controls and Web Services?

In BEA WebLogic 8.1, all external resources are accessed using controls. Thus a management tool that monitors latencies and inputs/outputs of all control invocations is able to provide visibility into all components the application depends upon. This tool should "instrument" any Web services irrespective of the platform used to publish the service (J2EE application server, .NET, integration brokers, or packaged applications). The monitoring of controls and distributed Web services must be done in a nonintrusive manner. At a minimum, it means that the management layer should be insertable without requiring any application changes. But the management layer should go beyond that and also be nonintrusive to the application's run-time behavior. For example, the management layer should take advantage of in-process SOAP interception mechanisms wherever possible to minimize latency overheads and to avoid adding new points of failure.

Does It Track End-to-End Transaction Flow and Conversations?

To enable you to troubleshoot distributed applications, the management tool must be able to track an application's executions in real time. This tracking must include all the controls utilized by the application.

BEA Weblogic 8.1 simplifies the creation of asynchronous, conversational, application-to-application interactions – to track such conversations the management tool must be able to identify and correlate messages that share a conversational context, and the ability to visually represent the execution timeline. While troubleshooting, you should be able to go from a high-level view of all Web services invocations involved in a conversation, down to execution details

about any control used within that service. Once the problem has been narrowed down to a specific control, conventional systems management tools can be used for debugging. You can then move execution vertically down through the multiple tiers of supporting infrastructure from the Java object to the application server, and down to the operating system and finally to the hardware server.

Does It Actively Manage Interactions Between Web Services?

To enable consistent enforcement of IT policies such as logging and security, these actions must be extracted out of the application code, and instead be performed automatically by a separate policy management layer. For example, developers should no longer be writing code explicitly to log Web service request and response messages. Nor should they be writing code to verify digital signatures or do compression, or check for viruses or for any such operational issues. Instead, the management layer should be responsible for enforcing policies by intercepting messages and actively manipulating message contents. Later, if a company changes a policy – for instance, it decides to log only message headers and not all message contents – no code will need to change. A simple configuration change will do the trick.

Does It Integrate with Your WebLogic 8.1 Development Environment?

Historically, management has always been "bolted-on" after applications have been developed and tested. Even with passive monitoring tools this was not ideal. With active management tools, this can be disastrous. With a management layer that needs to intercept in-flight messages and act on them, the dichotomy between development and management has to disappear. Management needs to be "baked-in" early in the development cycle, so that all testing – functional and performance – is performed on applications in the same way they would be in production. As discussed above, active management tools help achieve consistent policy enforcement by replacing code buried inside each service with centrally configurable policy handlers. To "bake-in" management early in development, and to replace coding with configuration when dealing with security and logging, your management tool must be well integrated with your WebLogic development tools and your end-to-end application development processes.

Does It Provide a Single Point for Configuring Security/QOS Policies Across Disparate Remote Services?

As Web services usage within your applications proliferates, Web services will be published and consumed by multiple applications. The management tool should allow you to centrally view and provision monitoring, security, and QOS policies for any service in your environment, and to "broadcast" management policy updates to multiple services without having to locally configure server-by-server and service-by-service.

Confluent's Web Services Management Solution

Confluent provides a management tool that plugs into the BEA WebLogic Workshop IDE, and adds easy-to-use management functionality. It allows you to automatically monitor Web services and execution of all constituent controls without requiring any additional coding.

Let's quickly walk through this tool in action, as it manages our sample application.

Once you've started CORE for BEA, you'll notice some additional panes in the lower half of the BEA WebLogic Workshop environment (see Figure 2). On your lower right, you have a Configure Policy pane, which you'll use to define the operational policies to be enforced for the Web service, e.g., logging, signature checking, or invocation to a third-party SAML provider. The lower left pane displays the controls used by the Order Management Web service – dbOrderManagement, serviceNotify, and ejbAccount. Monitoring can be switched-on for a Control with a single click!

Clicking the Runtime View tab displays details about all conversations associated with the service. For example, a conversation (representing an order transaction initiated at 13.38.39) completed successfully with a latency of 5422 ms. The user can also review the number of conversations, simple operations, successes, and failures.

The Operations tab enables a drill-down into the selected conversation (see Figure 3). It shows all the Web service operations that were invoked within the selected conversation. It also shows a graphic display representing the operations, their sequence in time, latency, and status. Note that the payload panel shows both the SOAP message that triggered the execution of the operation, and the SOAP result sent back to the requester.

Click the Controls tab to view more detail about the controls invoked within the selected operation. For each invoked control, the name, type, operation, status, and latency are displayed as both a table and a graph.

To summarize, the Runtime View pane allows a programmer to analyze the runtime behavior of a Web service at multiple levels: from conversations to operations and finally to the underlying Controls.

For more details on Confluent's Web services management software for BEA WebLogic, look up the BEA Premier Component Gallery at: http://dev2dev.bea.com/products/wlworkshop81/partners/index.jsp or check out www.confluentsoftware.com.

Conclusion

There is more to service-oriented applications than just WSDL, UDDI, and SOAP. BEA WebLogic 8.1 provides a comprehensive platform for developing enterprise-class service-oriented applications. But remember, it's never too early to start thinking of management. Ignoring this homily in the past might not have been fatal. Not anymore.

Sample application source code can be found at www.sys-con.com/weblogic/sourcec.cfm.

Listing 1

```
   /**
     * @common:operation
     */
    public String queryOrderStatus(String inAccountNumber, String inOrderNumber)
```

Listing 2

```
/**
     * Remote Web services
     * @common:control
     */
    private service.NotifyOrderServiceControl serviceNotify;
```

Listing 3

```
 /**
     * Create a new order
     * @common:operation
     * @jws:conversation phase="start"
     */
    public void createOrder(String inAccountNumber, String inOrderNumber)
```

Application Environments, Migration, and Transactions

ALL WITH ADDED INVESTMENT PROTECTION

No, don't worry, it's not a a floor-wax/dessert-topping/toothpaste article this month; it's simply a look at how multilanguage application environments might be used together in highly distributed systems. Interested? Well, don't worry, somebody has to be… Swallow your pride and read on, MacDuff!

BY PETER HOLDITCH

AUTHOR BIO

Peter Holditch joined BEA as a consultant in the Northern European Professional Services organization in September 1996. He now works as a presales architect in the UK. Peter has a degree in electronic and computer engineering from the University of Birmingham.

CONTACT...

peter.holditch@bea.com

REPRODUCED WITH PERMISSION FROM BEA SYSTEMS.

Looking at the IT systems of most large organizations is often closely akin to going on a fossil hunt. You dig through the J2EE surface to find a strata of C++ code in the supporting layers. Dig a little deeper and you start finding relics of a bygone age when C and Unix ruled the world. A little more scraping and an outcrop of COBOL is likely to be revealed, or maybe a little PL/1 and then, brushing aside the minicomputers that everyone forgot about, you might be lucky enough to unearth some real assembler. The archaeological metaphor starts to decay here, however. In archaeology, all the creatures in the various strata are dead. In many (most?) IT shops, the creatures from each period in history are actually cohabiting. They are working together to run the business! It's pretty scary stuff.

In short, heterogeneity is the norm. No surprises here; that's the main issue that Web services technology and service-oriented architecture are trying to address in their joint attempt to provide some kind of "Esperanto" and rules of engagement that all these systems can use to communicate (a less costly and more flexible proposition than the custom bits of string that have spent the last few years multiplying out of control). Of course, along with the strata of technology tend to exist strata of technologists, indispensable due to their business knowledge, not to mention their business application knowledge, but hardly next year's news. "That bearded guy in the corduroy trousers and Arran jumper… what does he do?!" (And we haven't even talked about client-side or small-scale systems and their associated entourage).

"So, that's a fascinating statement of the obvious", I hear you yawn. "What's it got to do with WebLogic and transactions?" I'm getting to that…

Well, Hold Up and Stop Getting Shirty

It will not have escaped the notice of the more attentive readers that BEA has application-server technology that addresses two of these strata; J2EE and COBOL/C/C++ on Unix. WebLogic Server (as you all know) is arguably the industry's preeminent Java application server. Tuxedo, meanwhile (as many of you may know), is indisputably the industry's leading C/COBOL/C++ on nonmainframe application server. So why is that interesting? For a large organization it's highly probable that there are pockets of expertise that fall into both Java and non-Java open-systems camps (to take but two strata). Clearly, the best way to get value for money from your developers is to set them to work doing what they do best – a C/Unix guy can't be expected to be a J2EE guru overnight, however good the training course. For the maximum short-term bang for the buck, get him writing the C that he knows and loves; over time, let him migrate to the modern Java world. Wherever he is at any point in time, BEA can provide him with an application server to underpin his endeavors, bringing to bear the required scalability and reliability whilst he concentrates on the business rules – where his focus is adding value to the business. Here, we have a technology migration scenario.

Just Opening a Port and Listening for HTTP Won't Cut the Mustard

Alternatively, maybe the organization is looking to expose a service-based interface to some of its Unix legacy code in order to include it more cost effectively into the new interconnected world. Web services are the key to the language it should speak, but what about the runtime characteristics? If the new service-oriented interface will generate a significant volume of requests to the old logic, just opening up a port and listening for HTTP SOAP requests isn't going to cut the

ATTN: Developers
STEP UP to the mike and be...
Go to http://developer.sys-con.com
HEARD!
Calling Sleek Geeks Everywhere!
Make sure you have your finger on the pulse of i-Technology...bookmark http://developer.sys-con.com today.
i-Technology News
i-Technology Views
i-Technology Comment
i-Technology Debate
DEVELOPER.SYS-CON.COM
© COPYRIGHT 2002, SYS-CON MEDIA WWW.SYS-CON.COM
SYS-CON MEDIA

mustard. A runtime infrastructure will be needed to throttle and balance load, not to mention making the resultant online system manageable in production. A good solution would be to move the old C/COBOL/C++ Unix code into an application server environment and we've already agreed (well, I already asserted, and I have the references) that Tuxedo is by far the preeminent UnIX non-Java application server environment. So here we have introduced Tuxedo to provide new levels of scalability and visibility to existing business logic – a technology renewal scenario.

The common thread between the two scenarios is that both envisage a Java application server (and a person of your exquisite taste is bound to have chosen WebLogic!) and Tuxedo coexisting in an operational environment.

The good news is that BEA understands this. WebLogic Server contains a piece of technology called the WebLogic Tuxedo Connector (WTC). WTC's sole purpose is to allow you (from a runtime perspective) to view Tuxedo and WebLogic as a single logical platform – messages can flow back and forth as needed, with the connectivity all defined by administrative configuration. In the technology migration scenario, this allows for the maximum return on developed assets – C or C++ developed in Tuxedo today does not need to be replaced tomorrow with Java, since it can be used from the Java code. Mixed skill development teams can put together applications composed of new (and existing) code in Java and non-Java languages with no integration pain. For the technology renewal scenario, the connectivity was an explicit requirement; it was the driver for the whole effort.

Now I hear a murmur going up: "Hang on, I thought SOAP was the way to do technology strata integration. And how can you be pushing WTC with a clear conscience? It's WLS specific, and I know there are competing Java 2 Connector Architecture–based products available from third parties!" So before you go into a frenzy, let me come back on those points.

WTC is architected after J2EECA, but it would be impossible (at this point) for it to be a J2EECA implementation; for a start, I already mentioned that it's bidirectional (unlike the standard) and as for SOAP and Web services, well... the standards as they are today lack some features that are needed for enterprise-grade solutions. One of the lacking areas (and finally, I get to the point that ties all this to the theme of my column!) is transactions!

Finally, I Get to the Point

The WebLogic Tuxedo connector can propagate two-phase commit transactions between the BEA WebLogic and Tuxedo environments. In my mind, that is implicit in making them a logically united platform (but that's just the kind of mind I have, I guess). None of the alternative products on the market that I know of (however standards based they claim to be) can do that. They all claim to support transactions, but what they mean is that they can start a Tuxedo transaction from the Java side – not that they can flow a transaction from it (or to it, come to that). In the migration scenario, it could be common for atomic updates to be needed where the data to be persisted has been computed on both sides of the technology fence; remember the scenario involved a mixed-skill development team. Full two-phase commit is needed and it is needed in such a way that it spans the two technologies. WTC provides that. Even for the renewal scenario, it would be lovely for all the systems to be interconnected in such a way that they were loosely coupled enough that they never needed to coexist in the same transaction. Life isn't always like that, and until implementations of WS-Transaction hit the market, WTC is the only game in town.

Oh, and don't forget those client-side, small-scale guys. Wouldn't it be great if they could apply their PowerBuilder/Delphi/-Visual Basic– style skills to this unified applications environment! Oh, sorry, I must have forgotten to mention... they can! With BEA WebLogic Workshop, not only can they access the Tuxedo-based resources via an out-of-the-box control, but since the fruits of their labors run atop WebLogic, their transactions can flow as freely as anyone else's between the old and the new technology worlds.

So, in conclusion, the WebLogic Tuxedo Connector is a kick-ass piece of technology (if you'll excuse the hyperbole). By its ability to stitch together the WebLogic and Tuxedo worlds with transactions (and security, come to mention it), it gives you maximum ability to leverage assets in the Java and non-Java worlds with the minimum of developer effort. And did I mention it's free?! Now that's what I call return on investment!

SYS-CON MEDIA

304,187 of the World's Foremost IT Professionals

DIRECT MAIL, EMAIL OR MULTI-CHANNEL

Target CTOs, CIOs and CXO-level IT professionals and developers who subscribe to SYS-CON Media's industry leading publications

Java Developer's Journal... The leading publication aimed specifically at corporate and independent java development professionals

LinuxWorld Magazine... The premier weekly resource of linux news for executives with key buying influences

Web Services Journal... The only Web Services magazine for CIOs, tech, marketing & product managers, VARs/ISVs, enterprise/app architects & developers

XML Journal... The world's #1 leading XML resource for CEOs, CTOs, technology solution architects, product managers, programmers and developers

PowerBuilder Developer's Journal... The only PowerBuilder resource for corporate and independent enterprise client/server and web developers

WebSphere Developer's Journal... The premier publication for those who design, build, customize, deploy, or administer IBM's WebSphere suite of Web Services software

ColdFusion Developer's Journal... The only publication dedicated to coldfusion web development

WebLogic Developer's Journal... The official magazine for BEA WebLogic application server software developers, IT management & users

.NET Developer's Journal... The must read iTechnology publication for Windows developers & CXO management professionals

Wireless Business & Technology... The wireless magazine for key corporate & engineering managers, and other executives who purchase communications products/services

Recommended for a variety of offers including Java, Internet, enterprise computing, e-business applications, training, software, hardware, data back up and storage, business applications, subscriptions, financial services, high ticket gifts and much more.

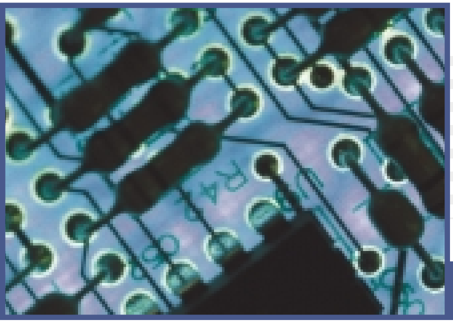
NOW AVAILABLE!
The SYS-CON Media Database
304,187 postal addresses

e-POSTDIRECT®
Your iMarketing Source

For email information:
contact Frank at 845-731-3832
frank.cipolla@epostdirect.com
epostdirect.com 800-409-4443 fax845-620-9035

For postal information:
contact Kevin at 845-731-2684
kevin.collopy@edithroman.com
edithroman.com 800-223-2194 fax845-620-9035

List Brokerage & Management
EDITH ROMAN
Data Processing • Email Marketing



BEA WebLogic Platform 8.1, first announced in March 2003, is now generally available. This release provides substantial productivity benefits for developers wishing to build new applications, integrate existing applications, and extend these applications to different groups of end users.

Getting Started with WebLogic Platform 8.1

WHAT YOU NEED FOR MAXIMUM PRODUCTIVITY

BY **WILL LYONS**

AUTHOR BIO

Will Lyons is the product manager for the BEA WebLogic Platform product. Will has been in product management with BEA since 1999, and has more than 15 years of experience in the information technology industry.

CONTACT...

wlyons@bea.com

REPRODUCED WITH PERMISSION FROM BEA SYSTEMS.

This article will encourage you to get started with WebLogic Platform and perform your own evaluation. We begin with a discussion of WebLogic Platform's capabilities and benefits, and follow with an overview of the Platform Tour application that ships with the product. The Platform Tour illustrates how BEA WebLogic Platform enables development of applications combining enterprise portals, Web applications, Web services, business processes, and custom business logic, all within the unified development environment provided by BEA WebLogic Workshop.

Capabilities and Benefits Summary

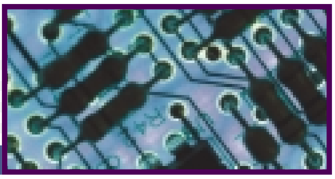
BEA WebLogic Platform 8.1 includes these component products.

- ***WebLogic JRockit:*** The industry's first JVM designed and optimized for server-side applications. WebLogic JRockit 8.1 incorporates new J2SE 1.4 features and other development and performance enhancements.
- ***WebLogic Server:*** The leading enterprise application server, and the foundation of WebLogic Platform. New features in this release include performance enhancements, Web services security and reliability enhancements, and configuration and monitoring enhancements.
- ***WebLogic Workshop:*** The unified development environment and runtime framework for WebLogic Platform applications. WebLogic Workshop 8.1 supports development of Web applications, custom controls, EJBs, business processes, and portals, as well as Web services applications, without requiring J2EE expertise.
- ***WebLogic Integration:*** The only solution offering rapid business integration today, supporting business process management, enterprise resource access, and integration services. WebLogic Integration 8.1 leverages the WebLogic Workshop development environment, and includes significant runtime and administration enhancements.
- ***WebLogic Portal:*** A comprehensive solution for delivery of custom enterprise portals. WebLogic Portal 8.1 provides a more flexible architectural framework for portal deployments, enhancements to business services such as content management and search, and new tools for portal life cycle management.

BEA WebLogic Platform combines the individual component products listed above into an integrated package with unified installation, configuration, licensing, documentation, operating system support, and maintenance support. WebLogic Platform 8.1 provides customers with faster time to value in their development projects through:

- ***Converging development and integration:*** Development of custom applications and application integration is accomplished in the same environment. Developers are able to focus on developing business logic rather than technology integration.
- ***Mass market J2EE:*** WebLogic Workshop now supports development of a wide variety of enterprise applications without requiring J2EE expertise, bringing the power of J2EE to a broader range of developers and development projects.
- ***Foundation for service-oriented architectures:*** Built-in support of software reuse through controls, pervasive Web services and XML support, and other capabilities enables development through composition of existing services and components.
- ***Unified development environment:*** WebLogic Workshop now provides a single tool in which the full range of WebLogic Platform applications can be developed.

Finally, BEA WebLogic Platform 8.1 makes it easy to get started with development of applications by leveraging all of the above capabilities. In addition to component product tutorials and examples, we've shipped a "Platform Tour" with the product that introduces the major product features, and illustrates how applications combining portals, Web



applications, controls, Web services, and business processes can be developed using the unified WebLogic Workshop development environment. The following sections provide an overview of the Platform Tour, and how Platform applications can be constructed.

Getting Started

Get started by downloading BEA WebLogic Platform 8.1 from the download link on www.bea.com and perform a complete installation. Installing on Windows or Linux systems is recommended in order to use the Workshop IDE as well as to run the Platform Tour. At the end of the installation you will have the option to launch QuickStart, which provides useful links for new users and product evaluators. When the QuickStart application launches, select "Take the end-to-end WebLogic Platform tour", which will launch the Platform Tour application.

Platform Tour Business Scenario

The Platform Tour is based on a business scenario in which Avitek, a vendor of electronic office equipment, has built a corporate intranet for managing employee information and equipment purchases. Launching the Platform Tour launches a browser and prompts the user to log in. Logging in as an employee using Tour instructions brings you to the employee portal.

The employee portal contains five portlets. The "Navigation" portlet on the upper left illustrates how Platform components are being used to render the page you are currently viewing. The "Doc Tour" portlet in the lower left provides access to detailed documentation on the Platform Tour. The "Log out" portlet on the upper right enables users to log out of the application. The "Employee" portlet displays employee information from Avitek's employee database. This information is being presented by a Web application that accesses database information via a Workshop control and renders it in the portlet. The "Order" portlet enables users to place orders for office equipment and view order status. Submitting an order via this portlet causes a Web application to call an order management business process via a Web service. This business process will automatically route the order to the employee's manager for approval.

Logging out and then logging back in as a manager brings you to a manager view of the intranet portal. This page allows the manager to approve employee equipment orders. Approving the employee order from this page will call the same order management business process, resulting in the order being approved and submitted, and the employee being notified of the approval. Logging out and then logging in again as an employee allows you to view the approved status of the order. This simple example application illustrates functionality commonly used in enterprise applications: portals to provide application access to different target users, Web applications to access databases and implement business logic, Web services for application integration, and business process management for orchestrating execution of business processes across applications.

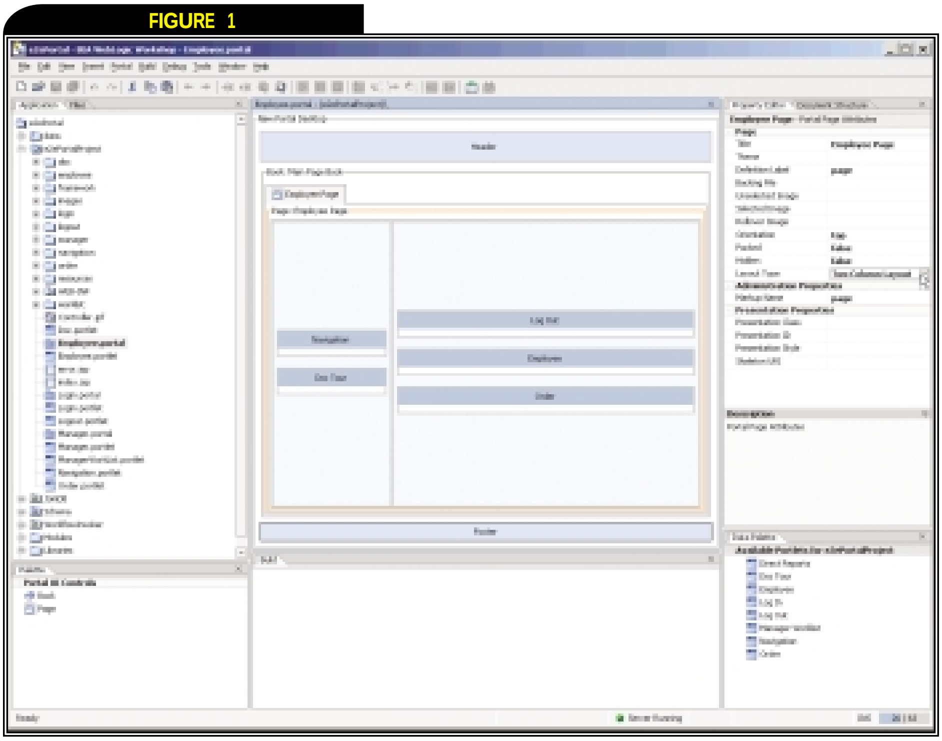
Now let's see how WebLogic Platform lets you build these types of applications within WebLogic Workshop. To do this on Windows and Linux systems, you launch the WebLogic Workshop IDE, through either QuickStart or the Windows Start Menu.

Viewing the Employee Portal Within the Workshop IDE

The Platform Tour is implemented as two WebLogic Workshop applications, one (e2ePortal) implementing the user portals and one (e2eWorkflow) implementing the business process, which can be deployed, managed, and updated independently from the portal application. Opening the e2ePortal application and selecting the employee.portal file enables you to view in the IDE the employee portal that we were accessing earlier when we executed the Platform Tour application (see Figure 1).

The WebLogic Workshop IDE uses consistent paradigms that enable visual development of different application and file types. In Figure 1, the Application Tab in the upper left-hand side shows a structured view of the files included in the e2ePortal application. The Edit Pane in the center provides a visual representation of the file that is currently open. This view of the employee.portal file shows the five portlets incorporated into the employee portal we accessed earlier. The Palette on the lower left shows UI controls that enable books and pages to be dragged, dropped, and added to portals under development, and the Data Palette on the lower right shows portlets that are available to add to portal pages being edited. The Property Editor on the upper right shows properties, such as layout properties, that have been assigned to the Employee Page currently selected in the design view. Let's take a closer look at how this application has been structured to cause the display of employee information within this portal.

FIGURE 1



Employee portal in WebLogic Workshop IDE

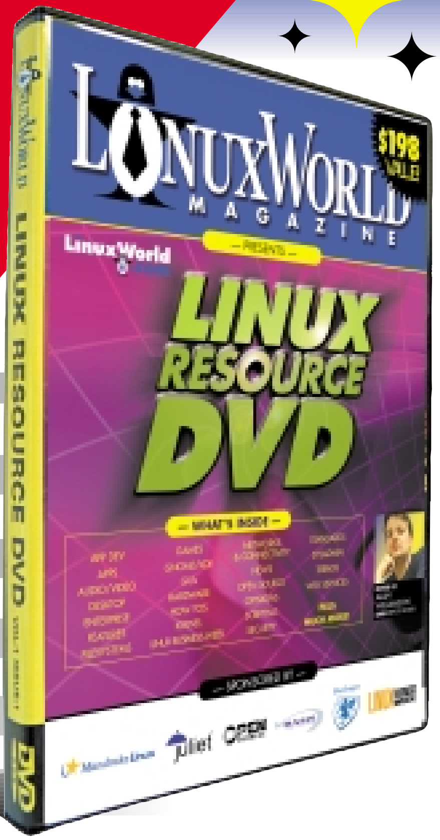
FREE

LINUX RESOURCE DVD

$198.00 Value

SIGN UP NOW!

Charter Subscription Offer



LinuxWorld Magazine™

Two Exclusive Offers
One Linux Magazine
You Can't Do Without!

— SPONSORED BY —

Mandrake Linux

juliet®

OPEN SYSTEMS INC.

BASIS International

BlueDragon

LINUX BUSINESS WEEK

By now, i-technology professionals are aware that Linux is emerging from the back room at a blistering speed. It is headed straight into the boardroom.

As software giants like IBM, HP, Sun, Oracle, and CA strive to make Linux easier for companies to adopt and deploy, *LinuxWorld Magazine* reflects how the market for Linux solutions is maturing, with insightful articles, features, interviews, and news. *LWM* editors and contributing writers are all thought leaders in their own right, writing Linux books, giving Linux lectures, and circulating widely within the Linux/open source world while holding down their real-world day jobs. Finally, a magazine dedicated to the IT managers and business leaders involved with assessing/making the business case for Linux has arrived!

Taking the Cost Out of Firewalls: Knowing Your Linux Pays

WWW.LINUXWORLD.COM

LinuxWorld Magazine

THE LEADING MAGAZINE FOR ENTERPRISE AND IT MANAGEMENT

SPECIAL PREMIER ISSUE! of LWM

Facing Microsoft's Fury...: Taking the Geek Out of Linux

Why Migrate to Linux?

A comprehensive solution for enterprise IT

Exclusive Interview with Eric Steven Raymond

LWM talks to the cofounder and president of the Open Source Initiative (OSI)

The Rise of Linux in the Enterprise

Big Gains in Small Business

FREE $198 DVD! SIGN UP NOW

www.sys-con.com/linux/subscribe

SUBSCRIBE NOW...

OFFER **1**

Subscribe NOW at our low introductory price and receive a FREE DVD with the completeLinuxWorld.com archives, leading Linux Software from Mandrake Linux, and more.

FREE DVD...

OFFER **2**

Receive a FREE copy of our first issue, and you can still take advantage of our special offer.

Two Year / 24 Issues - U.S. - $89.99 (Save $53.77 off newsstand price)
One Year / 12 Issues - U.S. - $49.99 (Save $21.89 off newsstand price)

Two Year - Canada & Mexico (airmail) - $119.99 (Save $48)
One Year - Canada & Mexico (airmail) - $79.99

Two Year - All Other Countries (airmail) - $176.00 (Save $40)
One Year - All Other Countries (airmail) - $99.99

echarterofferLW.cfm

LWM *also offers an online version of every print issue through our Digital Edition e-Newsletter.*

LinuxWorld Magazine :DigitalEdition

Tuesday, July 1, 2003

LinuxWorld Magazine Premiering Aug. 5, 2003, at LinuxWorld in San Francisco !

ORACLE

Leading Linux Luminaries

LinuxWorld Magazine
- Come to Our Launch at LinuxWorld Conference & Expo

One Year - Digital Edition
(International Subscribers Only) - $39.99
Includes the electronic edition (.pdf) and a FREE Archives CD of your choice!

Other Linux properties offered by SYS-CON Media include

Linux Business Week e-Newsletter,

www.linuxbusinessweek.com, and

LinuxWorld.com e-Newsletters.

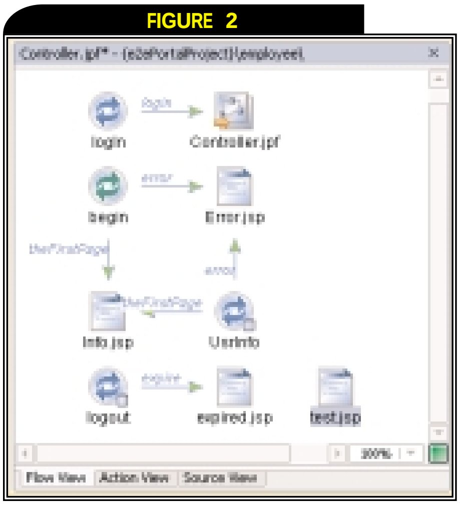
www.sys-con.com/linux/subscribecharterofferLW.cfm

LINUX BUSINESS WEEK   LinuxWorld.com

FOR MORE INFORMATION
contact customer service at
1 (888) 303-5282
or e-mail
subscribe@sys-con.com.

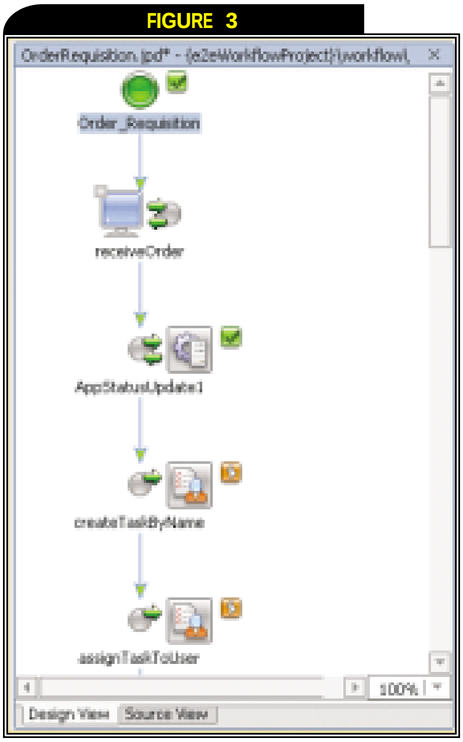
SYS-CON MEDIA

FIGURE 2



Java page flow in WebLogic Workshop IDE

FIGURE 3



Business process in WebLogic Workshop IDE

Displaying Employee Information

If you select the employee portlet in the design view, you can see the Portal instance properties that have been assigned to this portlet. The Portlet URI property shows that this Portal page points to a separate employee.portlet file, which controls which application information will be displayed within the employee portlet. Opening the employee.portlet file and inspecting the Portal Windows properties reveals that the portlet will call the begin action on a Java Page Flow (JPF) file implemented in BEA WebLogic Workshop.

Java Page Flows, based on the Struts architecture, enable the development of Web applications within WebLogic Workshop. A page flow links together multiple Web pages in a Web application and provides a central control mechanism (controller.jpf file) that coordinates the user's path through the pages and associated flow of data. Figure 2 shows the employee page flow called from the employee portlet using the Flow View in the Workshop IDE.

While the visual representation of a JPF is different from the visual representation of a portal file, Palettes and Properties continue to provide a consistent visual, drag-and-drop development environment. WebLogic Workshop provides two-way visual/source editing for JPFs that enables programmers to view and edit the source code corresponding to the Flow View of the application, and to write and include their own Java business logic. The flow diagram in Figure 2 indicates which pages are loaded by action methods, and which actions are raised on JSPs. In this case we've called the begin action that loads the info.jsp page. Double clicking on info.jsp displays this JSP in the Workshop JSP editor.

The WebLogic Workshop JSP Editor supports two-way visual/source construction of JSPs, as supported with other file types. In the Design View you can see that this JSP has been constructed to display the employee information (employee name, SSN, etc.) that was displayed earlier when we executed the Platform Tour. The application has been implemented such that employee information is accessed from an employee database using a Java control with the results displayed in the form specified for this JSP. Java controls make it easy to encapsulate business logic and to access enterprise resources such as databases, legacy applications, and Web services. WebLogic Workshop applications can use built-in controls provided with the product, or custom controls created by developers with the Workshop IDE.

Integrating Business Processes

Performing a visual inspection similar to the above indicates how the e2ePortal application is integrated with the business process in e2eWorkflow. The properties of the order portlet within the employee portal point to an order.portlet file. This file references the order page flow as defined in the order controller.jpf. Opening the order controller file reveals that the order page flow contains an action called placeOrder, which uses the WorkflowInvoker Java control to invoke the order requisition business process via a Web service. The resulting application behavior is that when an employee submits an equipment request, the order requisition business process is invoked via a Web service.

The order requisition business process, OrderRequisition.jpd, is depicted within the Edit Pane of the WebLogic Workshop IDE (see Figure 3), which provides a Design View and a Source View of the business process being created. Each icon represents a node or step in the business process. When the employee submits the order via the Web service, the receive order node receives the order and starts the business process, sending a message to the manager for order approval, and so on, consistent with the process flow illustrated when we originally executed the Platform Tour application. The same drag-and-drop visual paradigm is supported for adding nodes and controls into business processes, and to set properties on business process nodes.

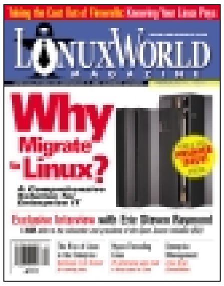
Summary

By inspecting the Platform Tour application shipped with BEA WebLogic Platform 8.1, we've seen how WebLogic Platform supports the ability to develop, in a single tool, applications combining enterprise portals, Web applications, Web services, business processes, and custom business logic. Common, intuitive, visual paradigms are used throughout these tools, simplifying the development experience, enabling developers to focus on business logic implementation rather than product or technology integration. Users interested in getting started can walk through the steps described in the article and familiarize themselves with product functionality at a high level in a matter of minutes. However, this discussion touches on the Platform Tour implementation only at a high level. The Doc Tour included with the application provides more detailed explanations for developers interested in further exploration. The WebLogic Workshop tutorials, component examples that ship with each of the component products, and Platform documentation provide developers with what they need to become productive developers of WebLogic Platform applications. Download the product now and get started – good luck!

A LIMITED TIME SAVINGS OFFER FROM SYS-CON Media

SUBSCRIBE TODAY TO MULTIPLE MAGAZINES

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



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

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

RECEIVE YOUR DIGITAL EDITION ACCESS CODE INSTANTLY
WITH YOUR PAID SUBSCRIPTIONS

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

Subscribe Online Today www.sys-con.com/2001/sub.cfm

SYS-CON MEDIA

What's WS-I Up To?

USING PROFILES TO IMPROVE INTEROPERABILITY

BY MARK NOTTINGHAM

Eighteen months ago, BEA, IBM, Microsoft, and a number of other companies who have invested in the future of Web services got together and formed WS-I, the Web Services Interoperability (WS-I) organization.

In the past few months, WS-I has delivered its first batch of final material, including the Basic Profile 1.0, and supporting test tools and sample applications. This article takes a look at what the WS-I aims to do, what it's done to date, and where the organization is going.

Why WS-I?

Web services aren't being developed and standardized all at once; there's simply too much work involved (e.g., messaging, transport, description, discovery, security, reliability, and so on). Not only would this be impractical but it would also face the risk of failure in one area overshadowing the entire solution. These are some of the reasons that modularity and evolvability – the ability to revise different portions of the stack over time – are core concepts in Web services architecture.

As a result, Web services standards are designed to be as open ended and extensible as possible, so new pieces can be plugged in easily. Unfortunately, the very traits that assure the long-term success of Web services bring about a much more immediate problem: interoperability.

Because Web services are being developed by different people in different places, and because they're changing over time, there are glitches in how they fit together. Additionally, because all of the pieces are built to allow a great deal of extensibility, it's difficult to know which pieces to use.

WS-I was founded to improve interoperability by establishing profiles of Web services specifications that show people how to use them together, and to narrow down the optional extensions so that WS-I – compliant Web services can interoperate effectively and easily. It does not create standards itself, but rather guides people in the use of existing standards.

Growing Up

WS-I's first major deliverable was the Basic Profile 1.0, which was finalized in early August. BP1.0 is designed to provide a foundation for further profiling work by assuring interoperability between SOAP 1.1, WSDL 1.1, and UDDI 2. It also discourages the use of some optional features, like SOAP Encoding, that were judged to be the source of interoperability problems.

The Basic Profile is a step towards interoperability; it provides a basis for further refinement with other profiles that are focused on one particular function (e.g., security or reliability) or on a particular class of uses (e.g., coarse-grained business messaging or fine-grained systems-level RPC).

To accompany the Basic Profile, the Test Tools Working Group has produced a "sniffer" that is used to intercept Web services messages; an analyzer that examines them, along with their WSDL files; and a set of Test Assertions that can help verify a Web service's conformance to the Basic Profile (because of the nature of some requirements, it's difficult, if not impossible, to automatically verify all aspects of conformance).

The Sample Applications Working Group has created a variety of materials to demonstrate how Basic Profile–conformant Web services might be constructed, as well as to showcase the interoperability enabled by following WS-I profiles. Usage Scenarios document the basic patterns that profile-conformant Web services might use, while Use Cases are specific example applications that many WS-I Members have implemented to demonstrate interoperability.

For more information on all of these deliverables, see the WS-I Web site at www.ws-i.org.

Behind the scenes, the WS-I has been maturing as an organization as well. The Board has opened up several committees, including Marketing and Communications and Liaison, so that any WS-I member can participate in their work. The Liaison committee has been working to establish formal ties with a variety of organizations, including the W3C, the Open Mobile Alliance, and others.

What's Next?

WS-I is currently working to improve interoperability in a number of areas.

The Basic Profile Working Group is examining

AUTHOR BIO

Mark Nottingham is a principal technologist in the Office of the CTO at BEA Systems, focusing on Web and Web services standards. He is the lead editor for the WS-I Basic Profile 1.0. He also participates in the W3C XML Protocol (SOAP) Working Group, and in the past has participated in the W3C, Internet Engineering Task Force, Java Community Process and other organizations.

CONTACT...

mark.nottingham@bea.com

REPRODUCED WITH PERMISSION FROM BEA SYSTEMS.

WebServices JOURNAL

.NET J2EE XML

LEARN WEB SERVICES. GET A NEW JOB !

The Best .NET Coverage Guaranteed!

SUBSCRIBE TODAY TO THE WORLD'S LEADING WEB SERVICES RESOURCE

Get Up to Speed with the Fourth Wave in Software Development

- **Real-World Web Services**: XML's Killer App*!*
- How to Use **SOAP** in the Enterprise
- Demystifying **ebXML** for success
- **Authentication**, **Authorization**, and **Auditing**
- **BPM** - Business Process Management
- Latest Information on **Evolving Standards**
- Vital technology **insights** from the nation's leading Technologists
- Industry **Case Studies** and **Success Stories**
- Making the Most of **.NET**
- **Web Services Security**
- How to Develop and Market Your Web Services
- **EAI** and Application Integration Tips
- **The Marketplace**: Tools, Engines, and Servers
- Integrating **XML** in a Web Services Environment
- **Wireless**: Enable Your **WAP** Projects and Build **Wireless Applications** with Web Services*!*
- Real-World **UDDI**
- **Swing**-Compliant Web Services
- *and much, much more!*

Only $69.99 for 1 year (12 issues)*

* Newsstand price $83.88 for 1 year

Subscribe online at www.wsj2.com or call 888 303-5252

*Offer subject to change without notice

EVALUATING and ADOPTING WEB SERVICES
A pragmatist's view
UNDERSTANDING THE WEB SERVICES VISION
Delivering on a new service-oriented model of computing
Scalable Web Services Using JMS & JCACHE
GETTING YOUR FUTURE: .NET OR J2EE?
WebServices JOURNAL
.NET or J2EE
STREET FIGHTING COMES TO WEB SERVICES

SYS-CON MEDIA

SYS-CON Media, the world's leading *i*-technology publisher of developer magazines and journals
brings you the most comprehensive coverage of Web services

the issues involved in extending the Basic Profile with SOAP with Attachments.

The Basic Security Working Group is concentrating on improving various aspects of security, a critical requirement for functional Web services in the enterprise and beyond. According to its charter, this includes a variety of mechanisms, including SSL/TLS for transport security, S/MIME for attachments, and the output of the OASIS Web Services Security Technical Committee for message security. It is important to have multiple mechanisms because of the variety of requirements; while simple SSL is often adequate to keep prying eyes away from your SOAP messages in flight, more advanced applications require encryption of only part of the message. This allows intermediaries, for example, to understand the parts of the message that they need, without showing them other potentially sensitive data in it. XML-based security also allows security properties like encryption and digital signatures to stay with a message no matter which transport is used, and even to stay with it if it is stored (for example, in a database).

Another newer activity is the formation of a Requirements Gathering Working Group. Although it had not officially started as of this writing, this effort has caused a lot of excitement because it enables WS-I to gather business scenarios directly from end users and other organizations as input to the profiling process. We anticipate that this group will drive the future direction of WS-I, and help assure that the WS-I has participation from the broader community, rather than just the Web services vendors.

There are many rumors about which problems WS-I will attack next. At some point, a Basic Profile based on SOAP 1.2 and perhaps WSDL 1.2 will be required, but most people believe that more implementation experience is needed with these specifications before they can be profiled. Likewise, some have advocated aggressive profiling of reliability and transaction specifications, but WS-I is reluctant to do so without a well-recognized standard in these areas.

More interestingly, there has been some discussion of profiling intermediaries, one of the more potentially useful yet vaguely specified mechanisms in Web services.

Challenges

For WS-I to meet these goals, it must master a number of challenges. Foremost in most people's minds are the political dynamics of the organization. Web services is somewhat unique in that virtually every IT vendor has signed onto it as an important new technology. However, this doesn't mean that there isn't disagreement about the details of their implementation, and sometimes traditional corporate rivalries come into play as well. WS-I, as a new organization in the center of this storm, must prove that it represents the interests of the community as well as its members in a fair and transparent fashion. Several steps have been taken in this direction, but only time will show it to have fully matured.

A more concrete issue is that of certification of Web services. WS-I has chosen not to require third-party certification to use its logo, because of the high costs this would bring to users. Instead, WS-I's recommendations are best effort and advisory, relying on the community to police itself over time. Whether this approach will work relies on consumers' ability to properly judge an implementation's claims.

> Unfortunately, the very traits that assure the long-term success of Web services bring about a much more immediate problem: interoperability

Finally, WS-I is focused on conformance of Web services instances (that is, a particular endpoint), rather than tools or platforms made to generate and consume them. This is because it is very difficult to describe conformance requirements for tools and platforms, and because there is still a wide variety of opinion about what is desirable in a platform. Despite this, many members strongly believe that this is an important step in proving the organization, and it seems inevitable that WS-I will ultimately move in this direction.

Getting Involved

Unlike traditional standards organizations, it's possible to get involved in WS-I and make a difference with comparatively little effort because it's an organization focused on users and requirements, rather than research and development. The input of developers, both in terms of experience gained and requirements for the future, is actively sought in the Requirements, Sample Applications, and Profile Working Groups.

THE WORLD'S LEADING INDEPENDENT WEBLOGIC DEVELOPER RESOURCE

Helping you enable intercompany collaboration on a global scale

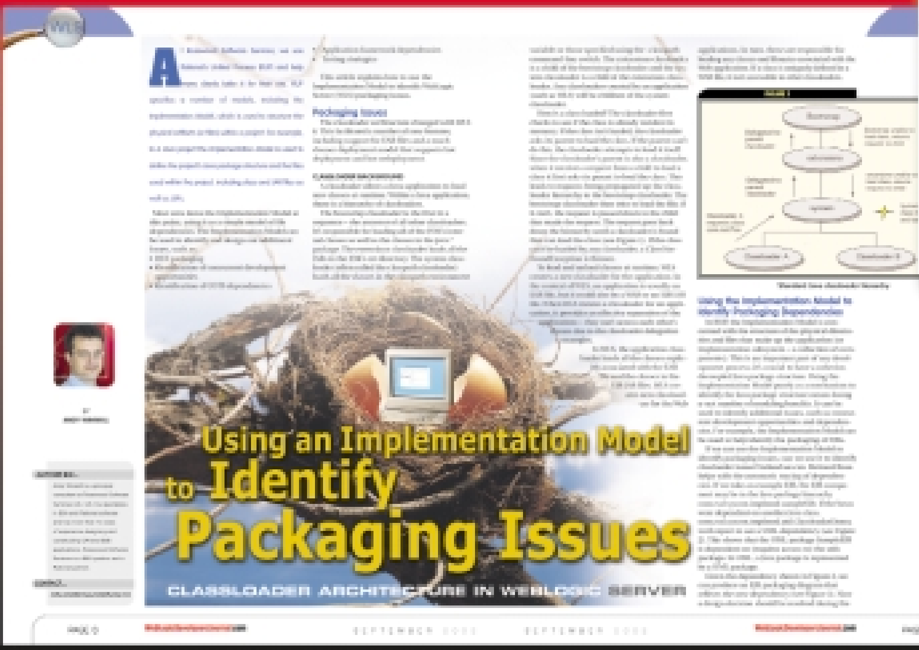
- Product Reviews
- Case Studies
- Tips, Tricks and More!

SPECIAL OFFER!
SAVE $31*
OFFER EXPIRES NOV 30, 2002

Now in More Than 5,000 Bookstores Worldwide – Subscribe *NOW!*

Go Online & Subscribe Today!

***Only $149 for 1 year (12 issues) – regular price $180.**


Using an Implementation Model to Identify Packaging Issues
CLASSLOADER ARCHITECTURE IN WEBLOGIC SERVER

CONTENT MANAGEMENT TAKES CENTER STAGE
BEA WebLogic DEVELOPER'S JOURNAL
A DYNAMIC SYSTEM FOR WEB APPLICATIONS
CONFERENCE AND EXPO OCTOBER 1-3
BEA WebLogic DEVELOPER'S JOURNAL
A CONVERSATION WITH ADAM BOSWORTH
SECURITY • PERFORMANCE • INTEGRATION
BEA WebLogic DEVELOPER'S JOURNAL
BUILDING BETTER BRIDGES

WebLogicDevelopersJournal.com

SYS-CON MEDIA

SYS-CON Media, the world's leading publisher of *i*-technology magazines for developers, software architects, and e-commerce professionals, brings you the most comprehensive coverage of WebLogic.

Panorama from Altaworks

AN ENTERPRISE-WIDE LOOK INTO YOUR SYSTEM

Reviewed by Jason Snyder

The clues were all right there. An application that had been through intense performance testing was getting regular complaints about its online response times. Because it couldn't be reproduced in the lab and the logging only stored exceptions, the lackluster performance was initially blamed on older browsers or other user technology problems. The Help Desk was inundated with frustrated user calls, and one of the Help Desk reports included the comment that the "10 and 4 o'clock rush hours" were seemingly getting worse.

This clue helped one of the more spirited of the application team track down the problem. In a conversation with the application DBA, it was mentioned that another application was using some of the data, "but that couldn't be a problem because they only needed one table." Further research indicated that late requirements changes and odd programming choices had resulted in the other application siphoning much more data than that during two scheduled processes that ran at, you guessed it, 10 a.m. and 4 p.m.

Applications don't exist in a vacuum. Tight integration across the enterprise exists almost everywhere, and the impact of some of this integration on performance is difficult to measure. Performance testing frequently focuses on one application, and its impact on others is not easy to measure. In reality, statistical analysis on enterprise production performance is needed, and Altaworks Panorama provides that capability. Panorama provides the necessary leads for tracking down cross-application, shared resource impacts that otherwise are very difficult to resolve.

Among other performance monitoring capabilities, Panorama also provides detailed performance analysis, alert notification, and key monitoring views through a Web-based user interface.

Panorama provides a series of low system–overhead Dynamic Sampling Agents that are installed on each system. Specific data adapters then collect information about the application or environment (see Table 1).

The Dynamic Sampling Agent controls this collection, consolidates the information, and sends it to the Service Management Platform (SMP).

The SMP then cycles through all of that data, performing complex statistical summarization, pattern analysis, and cross-component correlation. Alerts can be set that notify the Panorama Web-based user interface of key events for tracking.

I had a chance to test these features within Panorama using their Guided Tour, which runs on your PC. I recommend trying to get this, as it does a great job of demonstrating the key capabilities of the product.

Following install, I started it up with the Guided Tour, which set up a series of Data Adapters and began collecting information about my laptop and a sample application. From there, I was able to start and stop a series of simulations that allowed me monitor realistic changes in my system.

The user interface provides a series of tabs: "Performance", for live viewing of key performance statistics; "TopN", for ranking of performance data; "Deviation", which shows components' performance against expected; "Correlation", which allows multiple components to be analyzed together; "Events", where threshold limits for notification can be viewed; "Reports", where Crystal Reports against the data can be viewed; and "Configuration" for setup. A series of "Jump To" buttons exist that allow like data to be viewed among the tabs without having to set up the query again.

While testing a series of servlet calls, I was able to view the overall performance and then drill down on the perfor-

Panorama also provides "HedzUp" Event notification, which automatically calls out potential problems for the user

Altaworks

Corporate Information
Altaworks
55 Northeastern Blvd.
Nashua, NH 03062
USA
603-598-2582
1-888-877-7111

Sales:
sales@altaworks.com

Support:
support@altaworks.com

Minimum System Requirements
SMP System: Windows
OS: Windows 2000 Server v5.0 SP2
Database: Microsoft SQL Server 2000, Standard Edition

SMP System: Sun Solaris
OS: Sun Solaris 8
Database: Oracle enterprise Edition v8.1.7

User Interface Clients
Browser: Internet Explorer v5.5 (SP2) or Netscape Navigator v.7.0

mance to view specific servlet behavior. I used the "Top N" view (see Figure 1) to see the servlet performance ranked by peak servlet calls completed/second. Similarly, Panorama can enable method-level collection and ranking for EJBs. You can also do this for generic Java classes via an additional configuration file.

All of this information can be analyzed together to determine any performance correlation between the various components of an application. For example, I showed the correlation between a series of system calls and the percentage of processor time being used. The metric Correlator can be set up to eliminate metrics that have little impact on current performance. Panorama's Analysis Engine also provides a catalog of known performance-impact relationships that ease problem identification.

The performance data is collected every second by the DSA from the individual Data Adapters. When live data is being viewed, it is retrieved every 10 seconds from the appropriate DSA. The SMP harvests consolidated data from the DSAs every 15 minutes. This data is stored in the database, either SQL Server or Oracle. Historical information can then be accessed easily for additional analysis and review. The Time Control button on the left controls the time frame to retrieve.

Another feature that provided some real benefit was the ability to view the data through deviation scores. Deviation scores are roughly defined as "% outside of dynamic limit range for that time interval." This scoring technique lets you compare metrics from any system or application tier, even if they have different units.

I tried creating a few new views of the data being collected, but the process was confusing. A recommendation from Panorama is to use the TopN, Correlation, or Performance view to find the activity that interests you, pause screen refresh, press the Correlation button at the bottom of the page, and save the appropriate view you get with just those metrics. This method worked well with me.

It is also very easy to select and filter out information as needed. A series of dropdown boxes at the top of every view provides this functionality. Drilling down on the metrics is intuitively provided for by double-clicking on the metric that interests you. Setting up event notification is also fairly straight-forward. Panorama also provides "HedzUp" Event notification, which automatically calls out potential problems for the user.

Conclusion

Panorama provides the ability to monitor a wide variety of performance data across many tiers and applications. It also provides the ability to view this information in an easy-to-use browser-based UI. Most important, Panorama comes with a variety of statistical analysis tools that identify the impacts of this data across tiers and applications. It employs this intelligent analysis to identify and report significant events, data patterns, and abnormal behavior. This provides an enterprise-wide view into your entire system or suite of applications, instead of an application-by-application view that doesn't mirror how the enterprise really works.

AUTHOR BIO

Jason Snyder is the product review editor for *WebLogic Developer's Journal*. He is an architectural expert for CSC Consulting in Boston, and has served as the lead architect for several J2EE development projects. He has over 10 years of experience in software development, OO design, and application architecture.

CONTACT...

jasonsnyder@townisp.com

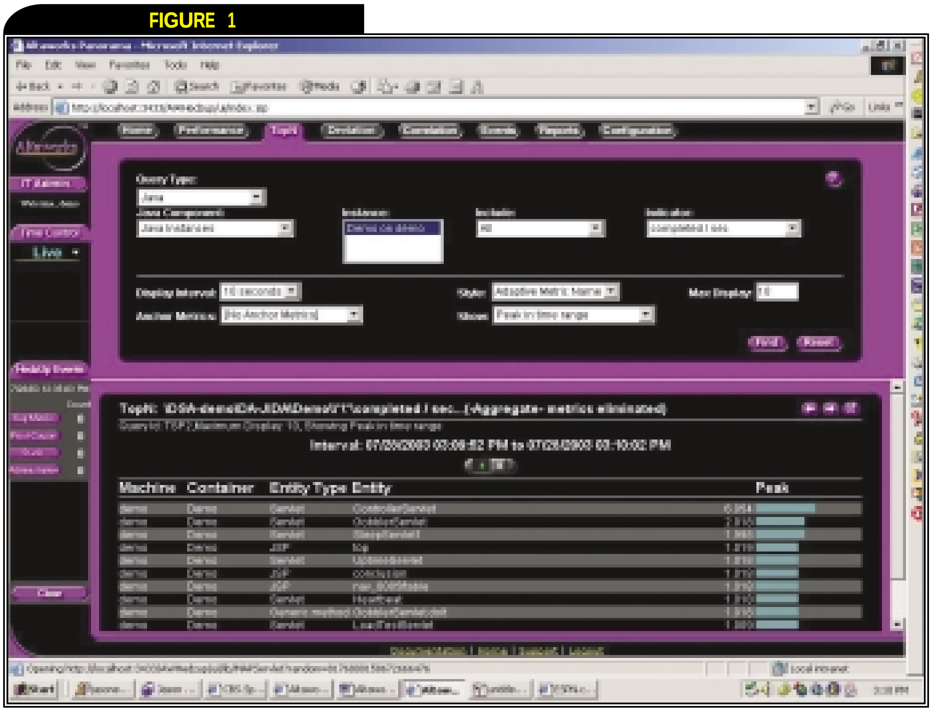
TABLE 1

Panorama includes agents for monitoring the following components. In addition to the Data Adapters listed here, custom Data Adapters can be written using a separately licensed Software Developer's Kit.

Web Servers	Application Servers	Databases	Operating Systems
Apache Microsoft IIS Microsoft Site Server Edition	BEA WebLogic (including JMX) Apache Tomcat with JBoss	IBM DB2 Oracle Microsoft SQL Server	Sun Solaris IBM AIX Windows NT/ 2000/XP Commerce **SNMP** Any network device or software product with an SNMP agent

Data Adapters supplied with Panorama

FIGURE 1



"Top N" view into servlet performance

Enterprise Portal Integration and the Enterprise Service Bus

PART 2

A BACKBONE FOR DYNAMIC CONTENT

BY MATT ROTHERA & HUB VANDERVOORT

AUTHOR BIO

Matt Rothera is a practice manager at Sonic Software (www.sonicsoftware.com), and works with customers to help plan, design, and deploy real-time, service-oriented architectures. With over 15 years of technical experience, Matt has worked with Fortune 100 customers to integrate their services with internal applications, business partners, and enterprise portals.

Hub Vandervoort is vice president of Professional Services for Sonic Software. He has over 20 years of experience as a consultant and senior technology executive in the networking, communications software, and Internet industries. Hub previously cofounded three startup ventures, including early message-oriented middleware (MOM) leader Horizon Strategies, Inc., which he merged with Momentum Software Corporation. He also cofounded, and served as board member of the Message-Oriented-Middleware Association (MOMA).

CONTACT...

hvanderv@sonicsoftware.com, mrothera@sonicsoftware.com

REPRODUCED WITH PERMISSION FROM BEA SYSTEMS.

Enterprise portals have become the most popular method of offering a common user interface to a suite of services across the enterprise. Offering business visibility, flexibility, and knowledge management, portals promise users the ability to monitor, search, and manage business activity across the enterprise.

However, as portals mature from basic offerings to more complex sets of services, they will require a robust, agile, and pervasive integration network to provide critical business services to portal users in real time. The Enterprise Service Bus (ESB) not only provides the necessary infrastructure to reliably link services with the portal across heterogeneous environments, disparate organizations, and geographic boundaries – it also has the capability to provide or enable more advanced services, such as integrated business process management and business activity monitoring.

In part 1 of this series (***WLDJ***, Vol. 2, issue 9), we discussed how the ESB can provide a reliable "services network" for a portal, providing reliable transport, intelligent routing, transformations, and the ability to operate in a highly distributed and federated environment. Strictly speaking, Gartner has defined the ESB has a "lightweight integration platform composed of message-oriented middleware (MOM), transformations, intelligent routing, and Web services." However, once coarse-grained services are accessible from the bus, services can be recombined in multiple ways to form unified business processes that span the enterprise. As business processes execute over days, months, or even years, a new wealth of information can be extracted to provide deeper business awareness and insight. These types of capabilities are the next evolution of the ESB and are critical to providing the "missing link" in a service-oriented architecture (SOA).

In this article, we'll examine two use cases that demonstrate the extended power of the ESB, beyond what a single application server or portal platform can provide on its own. The first use case discusses the notion of a "Worklist Waypoint," a step in a long-running business process that needs manual intervention. A portal offers the user a "human interface" accessible from virtually anywhere, making it a good choice for users to interact with a business process. The second use case shows how portals can provide a window into the operational aspects of the business processes, typically referred to as "Business Activity Monitoring." To illustrate these points, we will once again enhance the "Avitek Medical Records" tutorial shipped with BEA WebLogic 8.1. The tutorial is modeled around a portal (a J2EE Struts application) that allows patients, physicians, and administrators to view an aggregated set of medical records.

THE ESB: DISTRIBUTED PROCESS EXECUTION

When considering how a business process will interact with systems and humans on the ESB, it's important to consider if the business process will span geographic and organizational boundaries. This simple requirement introduces a new level of complexity, which some ESB providers handle "out of the box." For example, when considering distributed-process execution, is there a reliance on a centralized business process engine, or can the ESB provider distribute the process execution over multiple systems and across the ESB? Distributed-process execution eliminates the reliance on a centralized business process engine, allowing business processes to continue to execute in failure conditions. It also creates a more scalable processing environment, eliminating the bottleneck of a centralized business process engine.

Worklist Waypoint

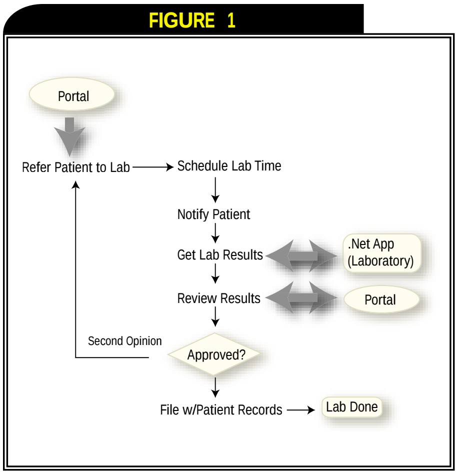
One of the great opportunities of SOA is the ability to create a set of common business processes that bind together services in different applications, technology domains, and even organizational domains. But with this opportunity come significant challenges. What type of infrastructure is required to allow business process execution to operate over long durations (days, weeks, months, or even years)? How will manual steps be intermixed with automated steps, and how will the business process execution interact with my various presentation tiers, both now and in the future?

Let's consider that Avitek would like to automate the process of scheduling lab work for a patient. Through the Avitek portal, a doctor will be able to refer a patient to one of several labs for a test. The business process might look something like Figure 1.

There are a few characteristics that should be observed from this diagram. The first, and primary item of note is that the business process must be long running. The business process combines manual steps (shaded in Figure 1) with automated steps (services on the ESB) that may take an undetermined amount of time. In addition to the latency caused by manual steps, the availability of services themselves may also extend the duration of a business process. Given the long-running nature of this type of business process, the ESB provides a perfect foundation to manage long-running business processes. Leveraging the guaranteed delivery characteristics of the underlying Java Message Service (JMS)–based transport, business process orchestration is a natural extension to the bus.

However, further analysis of the business process reveals a more complex requirement. The manual steps require some type of human interaction to satisfy the conditions of the business process. An open, standards-based, flexible interaction model is required to allow any presentation tier interface to interact with the business process from anywhere on the ESB. For example, Figure 1 shows two different interface types providing input to the process: a portal (in our case, the Avitek portal), and a custom .NET application (designed specifically for the lab technicians to provide results back to the process). The interfaces should be flexible enough to interact with other types as well, such as Interactive Voice Response (IVR), Web services, or even some type of handheld device used by a lab technician. A loosely coupled approach, such as Web services/XML or JMS/XML, is an appropriate choice to satisfy these requirements. The ESB supports both of these natively.

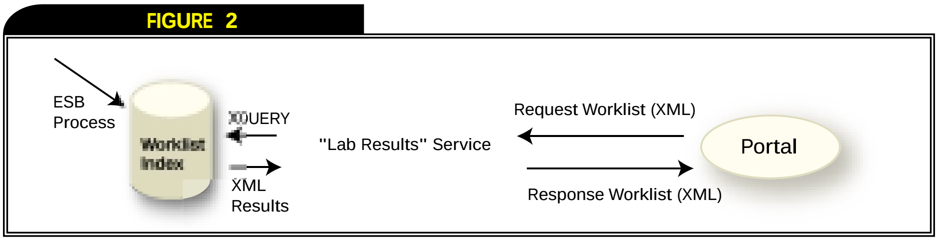
FIGURE 1



Lab Referral Process

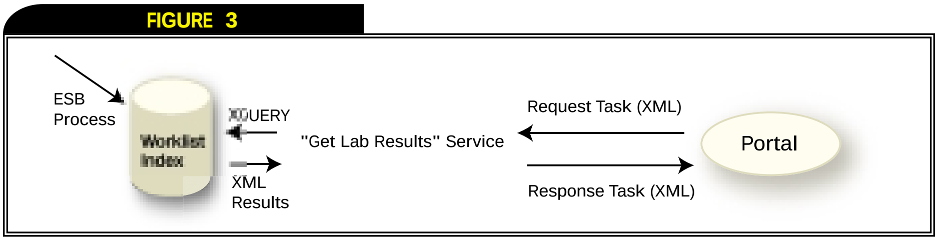
In business process terminology, manual steps are handled through a technique known as a "Worklist." A Worklist is a specialized "inbox" for tasks that need to be accomplished by an "Actor" in a business process. A task correlates to a specific instance of a business process definition. For example, a lab technician may have seven open lab requests that need fulfillment. In this context, a Worklist would contain seven process instances of the "Lab Referral Process" (from Figure 1). The first required service is a "Worklist Summary" that returns the list of tasks for a specific user (see Figure 2).

FIGURE 2



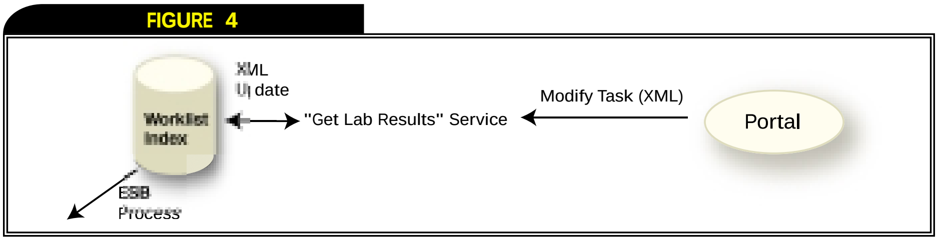
Worklist summary example: Lab Results Service"

FIGURE 3



Worklist task example: "Get Lab Results" service

FIGURE 4



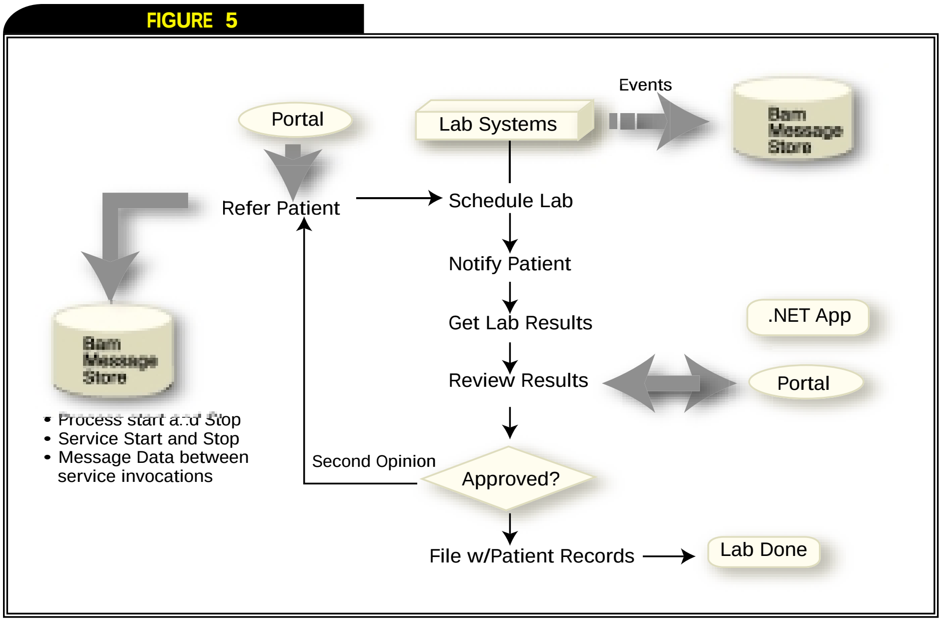
Task submission service

Once the task lists have been retrieved from the Worklist Summary service, the user may select a task, perform an operation on the task, and submit the task back to the process instance to continue executing. To illustrate the process, the presentation tier first extracts the task (using a process id embedded in the Worklist). The task is represented in XML and returned to the presentation tier (see Figure 3).

The presentation tier would then allow the user to operate on the task (XML). Once the user has completed the necessary editing, the presentation tier submits the modified task (XML) back to the process instance to continue. It's important to note here that the presentation tier submits only a modified XML document back to the process, and does not specifically invoke operations within the process itself. This simplifies process maintenance by keeping the presentation tier unaware of the definition of the business process (see Figure 4).

One common theme throughout these design patterns is the use of XML to represent the list of tasks, as well as the tasks themselves. XML provides the most widely accepted means of data interchange and will have the most adaptability for different

FIGURE 5



Lab referral process with BAM

presentation tiers. However, a significant point of concern is the number of different XML schemas that may be used to support the different business processes within the ESB. To understand the full nature of the problem, let's consider the Worklist interaction points from the Lab Referral Process shown in Figure 1:

1. When a lab technician makes a "Worklist Summary Request," they might need to know the patient ID, the type of lab work, and the due date.
2. When a lab technician makes a "Worklist Task" request for more information, they might need to know the name of the patient, the doctor, relevant history for the lab, and some detailed information related to the type of lab.
3. Finally, when the lab technician submits the task back to the process, they may need to fill out data specific to the type of lab work performed.

In this simple discreet use case, it is clear that there is a need to support a very contextual set of information to the presentation tier to provide meaningful interaction. When one starts considering the numbers of potential business processes and Worklist requirements within a real-world ESB, it begins to highlight some critical technical requirements on the infrastructure to provide a flexible, adaptive environment:

1. The Worklist Inbox Must Efficiently Manage XML

Imagine the prospect of manually constructing relational database tables and implementing marshaling code to move XML between the tables for every business process on the ESB. The manual effort required to implement the solution would outweigh the benefits of a service-oriented approach.

For this reason, a natural extension to the ESB is a native XML database, allowing XML of any schema to be persisted, indexed, retrieved, queried, and aggregated, without requiring support from a database administrator. The XML database features a schema-less design, allowing multiple XML schemas to co-exist in the database without prior knowledge or administration.

2. The Worklist Interaction Model Must Support a Standards-Based, Open, Flexible XML Query Mechanism

To accommodate the presentation needs of various business processes, allowing any data to be retrieved and manipulated at the presentation tier, a flexible XML query mechanism should be used to automate integration with the presentation tier. In addition to the sheer numbers of business processes that may need to be accommodated, imagine also the different needs of the presentation tiers. An IVR system may request a subset of the information, while a "thick" .NET client may have the ability to request all of the information.

Luckily, a W3C standard known as XQUERY is being developed for this purpose. An XQUERY facility that natively operates with the ESB's native XML database provides a standards-based approach for different presentation tiers on the ESB to interact with any business process, regardless of the XML schemas in use.

The "Lab Results" service could be quickly integrated with the Avitek Portal by constructing a JSP that invokes the service using XML parameters (the principle would be a useful parameter) and extracts the response in XML. The results are returned in XML and rendered in HTML back to the browser. The Java Standard Tag Library could be used to automate the process of parsing the XML and rendering the results to HTML, as well as any other type of XML-to-HTML mapping utility.

The service invocation mechanism could be any loosely coupled type of mechanism, such as JMS or HTTP/Web services. BEA WebLogic 8.1 provides automatic pooling of JMS connections, making the run-time invocation of JMS-based services extremely efficient and fast.

While the XQUERY statement could certainly be coded directly in the JSP and passed as a parameter to the "Lab Results" service, it may be better to embed the XQUERY into the service itself, providing better abstraction of presentation components from the underlying implementation details (see Listing 1).

Business Activity Monitoring

As critical business processes across the enterprise are moved into a loosely coupled, service-oriented framework on the ESB, new information can be unlocked that can have significant business value. Traditional application implementations typically capture the "end result" of a business process with business intelligence reporting tools. The ESB can capture the intermediate steps of a business process, allowing critical usage patterns to be identified and integrated into decision support systems in real time. Usage patterns can be identified to streamline business processes, or even head off problems before they cause irrevocable damage.

To illustrate the idea of business activity monitoring, let's once again consider the "Lab Referral" process outlined above. In this simple business process, what types of information could be derived?

1. As doctors refer patients to the lab, hospital administrators may want to be notified if the number of referrals in a given week exceeds the capacity of the lab. This might allow administrators to change work schedules for the lab technicians to accommodate the peak in activity.

JDJEDGE CONFERENCE & EXPO
JAVA DEVELOPER'S JOURNAL
Whole House Audio
EJB Techniques
Feature
J2EE Projects
Industry Commentary
Java Techniques
Feature
Java & Bluetooth
Product Review
First Look

JAVA Industry >Newsletter
WebServices Industry >Newsletter

2. As doctors refer patients to the lab, the CDC may want to know if the number of referrals of a certain type exceeds a certain threshold. Certain lab tests may indicate some type of new outbreak that needs to be dealt with quickly.
3. Administrators may want to monitor overall lab processing times and perform some analysis to optimize processes in the future. For example, they may want to determine the processing time or failure rates for individual lab technicians.

These use cases highlight some core capabilities that can be embedded directly into an ESB: XML data generation, capture, aggregation, and triggering. Once these basic facilities are inplace, the portal can harness these core services to provide interactive dashboards using standard XML facilities such as XQUERY and XSLT. It's important to recognize that there is no "one-size fits all BAM solution." Just as business models differ between organizations, it would stand to reason that the BAM solutions must also differ. The ESB takes the approach of providing a core set of building blocks that provide users with the ability to build their own BAM solutions quickly and in the context of their own business models at a lower cost than dedicated stand-alone platforms. To appreciate the infrastructure behind the portal to support these features, let's take a deeper look at the five layers of BAM.

Generate

In order to support a highly flexible BAM environment, the ESB must provide quick and efficient mechanisms to generate the core data that will feed the BAM message store, poviding business users the ability to perform real-time analytics on the data. There are primarily two sources of BAM data:

1 ***Business processes:*** As business processes execute, the ESB must be able to extract intermediate messages between services in the process, service start and stop times, and process start and stop times. The ESB must also provide the ability to determine information on processes in the aggregate, such as the total, average, and min/max values of critical process data.

   This is where ESB providers differ in implementation. The ideal ESB infrastructure will allow process tracking and intermediate data capture without refactoring the existing business processes. Tracking of BAM data should be a transparent deployment-time operation to streamline the process of tracking business data.

2. ***Systems on the Bus:*** As services are invoked within systems on the bus, there may be data internal to the systems that should be communicated out in an event-driven fashion. The ESB provides this feature based on its inherent underpinnings of JMS. As a standards-based asynchronous transport, JMS provides the best (and widely accepted) method of generating events from any system on the bus (see Figure 5).

Capture

Once data is generated from anywhere on the ESB, it must be intelligently routed to BAM Collection Points for real-time analysis. While the ESB provides an efficient means of routing through a distributed infrastructure, it must also provide efficient means of storing XML natively. Again, the concept of a native XML storage mechanism is crucial to a flexible BAM infrastructure. The schema-less design of the native XML storage provides the ability to quickly introduce new types of BAM data without the overhead of marshaling the XML into custom-defined relational tables.

Equally important is the ability to distribute the XML storage anywhere along the ESB. This allows deployment engineers to provide BAM collection points close to the point of origination, streamlining the flow of data traffic in the ESB. Imagine a case where the Lab System (from Figure 5) was generating a lot of event data. If a slow network or a WAN separated the portal and the Lab System, it would be better to capture and aggregate the data locally while also allowing aggregation of BAM data across all of the collection points.

Aggregate/Threshold/Trigger

Once the data is efficiently captured, there must be ways of aggregating the XML data to create useful reports of trends over time. XQUERY and XSLT can be used to aggregate information into summary XML documents, which can subsequently be dashboarded on a portal interface.

Once the data has been aggregated, business rules need to be executed on the aggregated data to determine if critical thresholds have been reached. Again, an XQUERY statement can be run to compare the data against critical business thresholds. If the thresholds are exceeded, the XQUERY can construct a "Business Alert" message to the ESB. Once the alert is on the ESB, it can be routed and handled in any method appropriate. For example, the alert can be routed to an e-mail service to notify one or more users of the event.

Event Processing

As Business Alerts are generated from the BAM Message Stores, they can be routed back to the ESB and handled in arbitrarily complex ways. The alerts may even kick off new, long-running business processes that run over the ESB. One interesting application of business alerts with respect to a portal is the notion of "feedback loops." Feedback loops provide the ability for the portal to modify its own behavior (within a set of constraints) based on external events.

Using the example "Lab Referral" process, imagine the case where the overall lab processing times exceeded a specific time threshold. In this case, the portal could provide additional content informing the doctor of the delay in lab processing times, and possibly even reroute the request to a secondary lab for processing.

Dashboard

Once the BAM data is in an agile XML database, the ability to view the data in a variety of ways on any presentation device (including a portal) becomes important. As executives begin to see the business value the BAM data, the requests for more visualization will increase. There are couple of challenges to consider:

1 An open, generic service architecture is important to integrate with different presentation devices. For this purpose, again the XQUERY and XML is a good choice.
2. For rapid integration, flexible methods of moving the XML to the presentation layer are required. In the case of a portal, XSLT is a great way to quickly move XML data into HTML visualization.

Conclusion

As the notion of SOA gains traction, Enterprise Portals will be expected to rapidly integrate new services reliably with the portal, providing users with instant access to these services. The ESB will be a critical component of the portal infrastructure, providing scalablity, reliability, and the ability to harness the complexity of very large services networks. However, there are still critical infrastructure services necessary to provide integrated business process management and business activity monitoring. As ESB's gain in popularity, core features such as native XML storage and integrated business process management will be considered standard and necessary features for any portal.

Listing 1

```
{-- Simple XQUERY to list all LabResults needing approval for a spe-
cific doctor
 -- Note: The developer can use run-time substitution to replace the
specific
 -- doctor's principle id in the where clause of the query
 --}

<LabResults>
  {
    for $i in document("labreviewworklist.xml")/LabApprovals/*
    where
      contains($i/doctor, "rothera")
    return
        <LabResult>
            { $i/laborder }
            { $i/description }
        </LabResult>
  }
</LabResults>
```

WLDJ ADVERTISER INDEX

ADVERTISER	URL	PHONE	PAGE
Attachmate	www.attachmate.com/SC13	800-426-6283	13
BEA Systems	http://dev2dev.bea.com/useworkshop	800-817-4BEA	3
Confluent Software	http://www.confluentsoftware.com	408-523-9000	25
Dirig Software	www.ebizperform.com	603-889-2777	27
Hewlett-Packard	http://devresource.hp.com/d2d.htm	800-752-0900	2
Intel	www.intel.com/IDS/register	800-628-8686	9
Intel	www.intel.com/ids/bea	800-628-8686	51
Java Developer's Journal	www.javadevelopersjournal.com	888-303-5282	47
LinuxWorld Magazine	www.linuxworld.com	888-303-5282	34-35
PANACYA	www.panacya.com	877-726-2292	29
Quest Software	http://java.quest.com/performasure/wldj	800-663-4723	11
Quest Software	http://java.quest.com/jcsv/wldj	800-663-4723	52
SYS-CON Media	http://developer.sys-con.com	201-802-3000	30
SYS-CON E-newsletters	www.sys-con.com	888-303-5282	47
SYS-CON Media Database	www.edithroman.com,	800-223-2194,	31
SYS-CON Media Database	www.epostdirect.com	800-409-4443	31
SYS-CON Publications	www.sys-con.com/2001/sub.cfm	888-303-5282	37
Web Services Journal	www.wsj2.com	888-303-5282	39
WebLogic Developer's Journal	www.weblogicdevelopersjournal.com	888-303-5282	41
Wily Technology	www.wilytech.com	888-GET-WILY	5, 19
Yahoo	www.enterprise.yahoo.com/bea-testdrive	866-267-7946	17

Advertiser is fully responsible for all financial liability and terms of the contract executed by their agents or agencies who are acting on behalf of the advertiser.

LOOK WHAT'S COMING NEXT MONTH

A Practical Solution to Parallel Business Logic Processing

A reusable, extensible, and robust framework that utilizes the asynchronous features offered by JMS and the robust runtime environment provided by the EJB container can solve the problem of parallel and concurrent business logic processing in enterprise and mission-critical applications.

Automated BEA WebLogic Server Domain Setup

BEA WebLogic Server domains in large-scale enterprises satisfy a broad range of requirements. To handle complexity, you can consider creating a clustered WebLogic Server domain in a multi-machine environment that starts the domain and deploys applications, in an automated manner.

Application Management with BEA WebLogic Server for Developers, part 2

The BEA WebLogic Server Administration Console can be used to manage WebLogic Server and to deploy and monitor the applications deployed on it. The command line tools weblogic.Admin and weblogic.Deployer that provide the full scope of management functionality from the command-line interface are discussed.

Management and Monitoring using the JRockit JVM

Although monitoring always has some impact on performance (and don't believe anyone who says otherwise), nevertheless the ability to monitor your applications through your development life cycle and in production environments is key.

BEA WebLogic DEVELOPER'S JOURNAL

News & Developments

Plumtree Supports WSRP and Proposed JSR 168 Standards

(San Francisco) – Plumtree Software has released new software to support WSRP from OASIS and the proposed final draft of the JSR 168 portlet standard. Plumtree is one of the first software vendors to release products supporting WSRP and JSR 168, and one of the only vendors to build and test those products for use on application servers from several different vendors. A customer investing in Plumtree's Enterprise Web software can now develop industry-standard portlets, and ensure that the investment can be put to work within virtually any portal. The Plumtree Container was designed to run on many application servers, including BEA WebLogic. www.plumtree.com

Borland Launches Optimizeit ServerTrace 2.0 DataCenter

(Scotts Valley, CA) – Borland Software Corporation has launched Optimizeit ServerTrace 2.0 DataCenter, a performance assurance solution designed to streamline the development and testing of J2EE applications in distributed and clustered environments.

Optimizeit ServerTrace 2.0 DataCenter is designed to help testers automatically diagnose the cause of application problems so that they can be fixed much more quickly. By optimizing their application performance, enterprises can reduce hardware and software platform costs in deployment. It also strengthens the test phase of the Borland Application Lifecycle Management (ALM) solution, designed to align business needs with software capability. www.borland.com

Quest Software Develops Spotlight for BEA WebLogic Server

(Irvine, CA) – Quest Software, a BEA star partner, has announced the immediate availability of Spotlight for BEA WebLogic Server, a real-time diagnostics tool designed to provide a visual representation of the components within WebLogic Server. This can allow administrators to drill down into actual or impending performance issues and enact rapid resolution to maintain optimal system response times for end users.

Spotlight for WebLogic Server is a critical component of Quest Software's integrated J2EE performance management solutions. The product suite includes Foglight for WebLogic, Spotlight for WebLogic, and PerformaSure, providing complete, end-to-end management to rapidly detect, diagnose, and resolve issues impacting end-user performance. www.quest.com/spotlight-portal/index.asp.

Fitech Laboratories Builds Financial System on WebLogic Platform 8.1

(San Jose, CA) – BEA Systems, Inc., the world's leading application infrastructure software company, has announced an alliance with Fitech Laboratories in which Fitech Laboratories will build its financial transaction system on BEA WebLogic Platform 8.1. Their system, xTrade, is designed to offer financial services and securities organizations in Japan with a low-cost alternative for online financial transactions.

Fitech Laboratories' xTrade provides general-purpose software components for online financial transactions, such as spot, margin, futures option transactions, and cash transfers or mail remittance. Built on the BEA WebLogic Platform, it enables customers to quickly build flexible financial trading systems at a low cost by combining, reusing, or adding software components. www.fitecklabs.com, www.bea.com

Cyanea Announces Support for BEA WebLogic Platform 8.1

(Oakland, CA) – Cyanea Systems Corp. has announced support for the BEA WebLogic Enterprise Platform in its flagship product Cyanea/One, a complete solution for managing distributed and mainframe-based applications. Cyanea helps enable data center managers, technical support personnel, test personnel, and application developers to effectively pinpoint and resolve complex J2EE application performance issues, minimizing downtime, enhancing availability, and improving capacity planning. www.cyanea.com

ReportingEngines Adds Embedded Application Data Reporting Tool

(Overland Park, KS) – ReportingEngines, a division of Actuate Corporation and provider of embedded reporting solutions for the J2EE platform, has announced that the Formula One e.Report Engine now includes the ability to access and generate reports from in-memory Java objects. This functionality, known as application data reporting, improves data-access efficiencies and increases report system scalability. It also supports a distributed application architecture and provides developers with another data source for reporting in addition to previous support for relational databases, EJBs, flat files, and XML data streams.

ReportingEngines is offering Java developers the opportunity to try Application Data Reporting free for 30 days. A trial version of the Formula One e.Report Engine and a 10-step tutorial can be downloaded from the ReportingEngines Web site at www.reportingengines.com/info/adr9.jsp.

BEA Adds to WebLogic 8.1 Support

(San Jose, CA) – BEA Systems, the application infrastructure software company, is now offering consulting, education, and support services designed to accelerate return on investment and speed time-to-value for customers using BEA WebLogic Platform 8.1. BEA saysthese services will complement their standards-based application platform suite and are designed to help lower customers' costs while increasing their ROI. www.bea.com