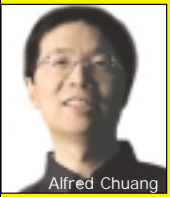


# BEA WebLogic

DEVELOPER'S JOURNAL

Volume:1 Issue:3

weblogicdevelopersjournal.com



**A LETTER FROM THE PRESIDENT**

Founder, CEO, and President, BEA Systems, Inc. page 5



**A STRATEGY FOR THE FUTURE**

An Interview with Scott Dietzen page 56

FROM THE EDITOR

"The Platform of the 21st Century"

by Jason Westra

page 5

NEWS & DEVELOPMENTS

page 62

web services **EDGE** world tour 2002

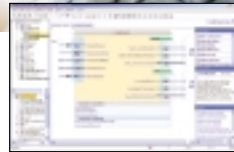
NEW YORK, NY ..... MARCH 19  
SAN FRANCISCO, CA ..... APRIL 22

Page 28 Register for Web Services Edge 2002 East Gold Passport, Attend the World Tour FREE!



Carl Sjogreen 10

TRANSACTION MANAGEMENT: **Transactions: What are they, anyway?** The beauty of building on an app server



8

Peter Holditch

Sam's Soapbox: **Caching** Increased performance with investment in design and architecture



18

Sam Pullara

eWORLD FEATURE: **Introducing BEA WebLogic Server 7.0** The tools necessary to meeting today's IT challenges



22

Jim Rivera

eWORLD FEATURE: **The BEA WebLogic Platform** Rich, easy-to-use application infrastructure functionality



30

Will Lyons

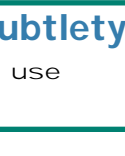
eWORLD FEATURE: **BEA WebLogic Portal Overview** A true implementation of the J2EE component model



36

Dmitry Dimov

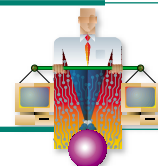
SOLUTIONS: **Holistic Infrastructure Monitoring and Management** Becoming a true business enabler



46

Greg Peters & Lance Peterson

ARCHITECTING AND ARCHITECTURE: **The Subtlety and Power of EJBs** Why you should use EJBs on your next implementation



50

Tyler Jewell

# BEA

[www.developer.bea.com](http://www.developer.bea.com)

# BEA

[www.developer.bea.com](http://www.developer.bea.com)

# Wily Technology

[www.wilytech.com](http://www.wilytech.com)

**EDITORIAL ADVISORY BOARD**  
TYLER JEWELL, FLOYD MARINESCU,  
SEAN RHODY

**FOUNDING EDITOR**  
PETER ZADROZNY

**EDITOR-IN-CHIEF**  
JASON WESTRA

**EDITORIAL DIRECTOR**  
JEREMY GEELAN

**EXECUTIVE EDITOR**  
GAIL SCHULTZ

**MANAGING EDITOR**  
CHERYL VAN SISE

**SENIOR EDITOR**  
M'LOU PINKHAM

**EDITOR**  
NANCY VALENTINE

**ASSOCIATE EDITOR**  
JAMIE MATUSOW

**ASSOCIATE EDITOR**  
JEAN CASSIDY

**WRITERS IN THIS ISSUE**

ALFRED CHUANG, DIMITRY DIMOV,  
PETER HOLDITCH, TYLER JEWELL, WILL LYONS,  
GREG PETERS, LANCE PETERSON,  
SAM PULLARA, JIM RIVERA,  
CARL SJOGREEN, JASON WESTRA

**SUBSCRIPTIONS**

For subscriptions and requests for Bulk Orders, please send your letters to Subscription Department. SUBSCRIPTION HOTLINE: SUBSCRIBE@SYS-CON.COM Cover Price: \$15/Issue Domestic: \$149/YR (12 Issues) Canada/Mexico: \$169/YR Overseas: \$179/YR (U.S. Banks or Money Orders)

**PUBLISHER, PRESIDENT AND CEO**  
FUAT A. KIRCAALI

**VP, PRODUCTION & DESIGN**  
JIM MORGAN

**SENIOR VP, SALES & MARKETING**  
CARMEN GONZALEZ

**VP, SALES & MARKETING**  
MILES SILVERMAN

**VP, EVENTS**  
CATHY WALTERS

**VP, BUSINESS DEVELOPMENT**  
GRISHA DAVIDA

**CHIEF FINANCIAL OFFICER**  
BRUCE KANNER

**ASSISTANT CONTROLLER**  
JUDITH CALNAN

**ACCOUNTS PAYABLE**  
JOAN LAROSE

**ACCOUNTS RECEIVABLE**  
JAN BRAIDECHE

**ACCOUNTING CLERK**  
BETTY WHITE

**ADVERTISING ACCOUNT MANAGERS**  
MEGAN RING • ROBYN FORMA

**ASSOCIATE SALES MANAGERS**  
CARRIE GEBERT • ALISA CATALANO  
KRISTIN KUHNLE

**CONFERENCE MANAGER**  
MICHAEL LYNCH

**EXECUTIVES, EXHIBITS**  
MICHAEL PESICK • RICHARD ANDERSON

**SHOW ASSISTANTS**  
NIKI PANAGOPOULOS • JACLYN REDMOND

**ART DIRECTOR**  
ALEX BOTERO

**ASSOCIATE ART DIRECTORS**  
AARATHI VENKATARAMAN • LOUIS CUFFARI  
CATHRYN BURAK • RICHARD SILVERBERG

**WEBMASTER**  
ROBERT DIAMOND

**WEB DESIGNERS**  
STEPHEN KILMURRAY • CHRISTOPHER CROCE

**CONTENT EDITOR**  
LIN GOETZ

**JDSTORE.COM**  
ANTHONY D. SPITZER

**CUSTOMER SERVICE**  
ANTHONY D. SPITZER

**CUSTOMER SERVICE LIASION**  
PATTI DEL VECCHIO

**EDITORIAL OFFICES**

SYS-CON Publications, Inc.  
135 Chestnut Ridge Road, Montvale, NJ 07645  
Telephone: 201 532-3000 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 DEVELOPERS' JOURNAL,  
SYS-CON Publications, Inc.  
135 Chestnut Ridge Road, Montvale, NJ 07645

# Introducing... "the Platform of the 21st Century"



BY JASON WESTRA  
EDITOR-IN-CHIEF

**W**elcome to the eWorld issue of **BEA WebLogic Developers**

**Journal!** Each year WebLogic developers and managers make a pilgrimage to eWorld to meet with vendors hawking wares in the exhibit hall, to listen to BEA visionaries in jam-packed sessions, and perhaps most of all, to see what new, cool stuff BEA will announce at the show! Well, comrades, many new products are coming our way, and if you miss the chance to hear about them at the show, make sure you read about them in this issue. This month, BEA publicly announced its vision for an "application infrastructure platform." In the words of Will Lyons, "Our vision of an application infrastructure platform is a package that will enable all enterprise developers to build integrated, multitier applications more easily and rapidly." In the past, WebLogic projects were hampered by high learning curves for J2EE development. A decent IDE for WebLogic developers was unheard of! This is going to change.

In the past, BEA's products have been about as "tight" as a rock band that hasn't played together in decades. This will change too. This month, Will discusses the new BEA WebLogic Platform, an integrated technology stack that creates a seamless experience for developers and administrators across the WebLogic Server 7.0, WebLogic Portal, and WebLogic Integration products. Benefits of an integrated platform include: a single license, a single install, and common packaging, release, and maintenance of the platform. The new platform boasts a modular design to allow BEA partners to integrate their products seamlessly into the platform.

WebLogic Portal, a major component of the WebLogic Platform, is also making news. Dimitry Dimov, product manager of BEA's E-Commerce Application Components Division, provides a tech-

nical overview of the Portal product's new features. For instance, Portal's WebFlow technology has been enhanced with a graphical editor that allows you to visually "wire" together the flow of your Web application. Other features include a unique approach to configuration management, performance improvements to the unified user profile, the ability to allow your application to present profile data from disparate sources in a unified manner, enhanced group-based security, and dynamic, business entitlement support for personalized portlets.

Perhaps the most exciting news at eWorld, however, is the announcement of "Cajun," a development tool that puts the power of J2EE in the hands of all developers, regardless of experience. "Cajun" lowers barriers to entry into the J2EE environment by simplifying complex portions of J2EE development such as developing EJBs, Web services, JMS, and clients (proxies) to these and other J2EE services. It also allows experienced developers to work directly with J2EE APIs, Web services, and Java.

"Cajun" is an IDE for Web services that includes the ability to write and run your Web services from within a single development environment. It includes a full-featured editor, a debugger for Web services, both local and distributed on other machines, and "Cajun" controls – an innovative wrapper around normally complex J2EE and runtime services. "Cajun" Controls allow "Cajun" developers to easily access J2EE services with little knowledge of the underlying implementation. In his article, "Introducing Cajun – Building Web Services for the Enterprise," Carl Sjogreen, a product manager at BEA who represents the developer community on the "Cajun" product management team, provides us with a glimpse of "Cajun" and how it integrates into the WebLogic Platform.

There you have it... WebLogic Platform, the application infrastructure platform of the 21st century!



**AUTHOR BIO...**

Jason Westra, CTO of Verge Technologies Group, is the editor-in-chief of *BEA WebLogic Developer's Journal*. Jason has written for SYS-CON's publications for several years and has substantial knowledge of WebLogic Server.

CONTACT:jason@sys-con.com

© COPYRIGHT 2002 BY SYS-CON PUBLICATIONS, INC. ALL RIGHTS RESERVED. NO PART OF THIS PUBLICATION MAY BE REPRODUCED OR TRANSMITTED IN ANY FORM OR BY ANY MEANS, ELECTRONIC OR MECHANICAL, INCLUDING PHOTOCOPYING OR ANY INFORMATION STORAGE AND RETRIEVAL SYSTEM, WITHOUT WRITTEN PERMISSION. FOR PROMOTIONAL REPRINTS, CONTACT REPRINT COORDINATOR. SYS-CON PUBLICATIONS, INC. RESERVES THE RIGHT TO REUSE, RE-PUBLISH AND AUTHORIZE THE READERS TO USE THE ARTICLES SUBMITTED FOR PUBLICATION. ALL BRAND AND PRODUCT NAMES USED ON THESE PAGES ARE TRADE NAMES, SERVICE MARKS OR TRADEMARKS OF THEIR RESPECTIVE COMPANIES. SYS-CON PUBLICATIONS, INC., IS NOT AFFILIATED WITH THE COMPANIES OR PRODUCTS COVERED IN WEBLOGIC DEVELOPERS' JOURNAL.



Dear **WLDJ** Readers

At BEA, we know that the WebLogic community of developers – which currently numbers more than 350,000 people worldwide – is one of the main reasons behind our success. We are committed to making your work easier, more productive, more rewarding, and more informed. The new **WebLogic Developers Journal**, a great source for in-depth, independent information on developing with WebLogic, is just one example of how we are doing that. I am very pleased to have the opportunity to welcome you to this new publication, and encourage you to subscribe so that you can keep abreast of the latest developments in the BEA WebLogic product line.

There are many more examples of how we're investing in you, the WebLogic developer. This week at BEA eWorld 2002, the 7th Annual BEA Developers Conference, some 3,000 of you will learn about these plans in detail. And, for those of you who weren't able to make it to eWorld, this issue of WLDJ provides a great overview of what you can expect to see from us in the coming months. You'll see first-hand how BEA is revolutionizing Java – making it radically easier to use and accessible to potentially four times J2EE programmers. You'll learn how WebLogic advancements are radically unifying application infrastructure, making your life easier and helping you have an even greater impact on your business. And you'll get an inside look at our new developer portal, designed to fast-track your development success.

So let's get into some of the details. This week we are launching the beta of our new integrated development environment and runtime framework for building and managing Web services. Code-named "Cajun," this product is nothing short of a technological breakthrough that will make both the WebLogic Platform and J2EE accessible to every developer in your company. If you work with COBOL, PowerBuilder, or Visual Basic and have little knowledge of Java, it no longer matters – now you can build and deploy robust, mission-critical applications and Web services on the WebLogic platform. And, for you seasoned J2EE developers, "Cajun" provides a framework that can accelerate your development processes. You can download a free evaluation copy of "Cajun" from [www.bea.com/products/index.shtml](http://www.bea.com/products/index.shtml).

At eWorld, we also announced our radically unified enterprise software platform, and our vision and roadmap for delivering the application infrastructure that businesses need to radically simplify their IT architecture. The BEA WebLogic Platform combines everything you need to develop enterprise-class applications and Web services – including new tools, third-party support, and several important integration enhancements – in a unified offering provides a more productive out-of-the-box experience.

Also coming up this year are new versions of BEA WebLogic Server, BEA WebLogic Portal, and BEA WebLogic Integration. BEA WebLogic Server 7.0, launching this spring, features a completely redesigned security framework, scalable and highly available JMS, enhanced support for enterprise-class Web services, and new features (including "Cajun") to enhance your productivity. Each product, which will be offered separately as well as part of the BEA WebLogic Platform, is designed to simplify enterprise application development, deployment, management, and integration — making your life easier and ensuring maximum value out of your IT choices. In this issue of **WLDJ**, you'll find detailed articles on these products, on the platform, and on "Cajun."

Finally, at eWorld we announced the launch of dev2dev, the center of gravity for the BEA developer community. BEA dev2dev pulls together all the resources you need to be as productive as possible. The dev2dev community is where developers can get the latest BEA technologies, share best practices, learn from the experts, and find the people, tools, and resources you need to get the job done. Here you'll find daily updated technical content, including articles and regular columns from experts throughout the BEA developer community, software downloads, demos and code samples, and a ton of helpful material for developers of all levels. And this is just the start. Register today at <http://dev2dev.bea.com> and join the community.

These are just a few of the ways we're investing in you. We are committed to making your work easier, more productive, rewarding, and informed. And we hope you find the **WebLogic Developers Journal** exactly that, rewarding and informative. If you're not already a subscriber to **WLDJ**, sign up now. I'm looking forward to another exciting year with you in the world of WebLogic!

Thank you for your continued support.

Alfred Chuang  
Founder, CEO, and President, BEA Systems, Inc.

# Altaworks

[www.altaworks.com](http://www.altaworks.com)





As I understand Western ideas about the world, there seem to have been three distinct phases through which they have passed. In the beginning, people believed that the world was flat, and at the center of the universe.

## Transactions: What are they, anyway?

### THE BEAUTY OF BUILDING ON AN APP SERVER

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](mailto:peter.holditch@bea.com)

Eventually, this view was confounded by the likes of Christopher Columbus discovering the New World while singularly failing to sail off the edge, as many were convinced he would do. So the world was spherical, and in the center of the universe. Then Galileo spent a long time in a David and Goliath-type battle with the Church, trying to tell everyone that the world revolved around the Sun, not the other way around. Eventually, he prevailed and now nobody doubts that the world is a spherical planet revolving around the Sun. Throughout this, people have always used the word “world” to refer to the planet (which has remained untrammelled by the debate as to its nature) while the word conjured different pictures in their mind’s eyes, depending on the age in which they lived.

So what’s a transaction? The definition of a transaction, unlike the definition of the world, changes in two dimensions, both over time, like the definition of the world, and depending on who the observer is. If you ask a bank teller what a transaction is, the answer will probably be that it’s a debit or a credit; a salesman will view it as a purchase of his wares, while a techie propeller-head may well start talking in a speedy staccato

mumble about two-phase commits and heuristics.

So which of them is right? Of course, they all are. They just look at the question from a different background. Wherever there is a difference between how a user and a techie view a concept, code is written to bridge the gap. It is that gap that is filled by the application server, whose intent is to provide an environment to the app. Designer that is more closely related to the business problem they are asked to solve and less related to the idiosyncrasies of how computer systems work.

#### Web Services! Where Did They Come From?

The latest attempt to move the bar between technology and users is Web services. The concept is that atomic-style business functions, implemented using various technologies, can be advertised via a repository to be assembled together into useful systems by a ‘drag and drop’-type business person, rather than by a ‘lines of code’-type technical person. The other promise that Web services offers is that the atomic functions that are composed together into ‘business transactions’ can potentially be geographically dispersed in nature. Note well that you have just seen a sentence with the “transaction” word in it coupled with components that are potentially dispersed. I didn’t just introduce Web services as a red herring; there is really a relationship between them and transactions. In fact, both could even be called transactions. Web services are what the user community might call transactions – debit, credit, transfer, etc. – while their realization will require what the technical community would call a transaction (two-phase commit and all that jazz) since a piece of architecture is needed to provide some level of consistency between the atomic operations that have been composed together.

#### Quick, Show Me!

So let’s take an example. A manufacturer needs to procure 30,000 widgets to complete an order for 300 of their new best-of-breed whatsits. They have three suppliers with whom they regularly deal – electronically of course – via Web services. So the Web service within the manufacturer is called “procure parts” and someone duly calls that with the necessary parameters, the required delivery date, and the number and type of parts required.

The implementation of this Web service calls the “supply parts” Web service of each of the suppliers, selects the most competitive quote, and then calls



the “order parts” service of the successful supplier.

On the face of it, this seems like something that could easily be implemented using Enterprise Java Bean components, called using RMI with data updates coordinated using Java Transaction APIs (JTAs). On closer inspection, however, the impedance mismatch between the low-level technical view of a transaction and the business level view of a transaction is revealed.

For a start, the JTA model of a two-phase commit transaction assumes that a transaction can complete successfully if and only if all the component parts of it succeed. In our example, we can successfully complete the business level transaction if any of the three suppliers can fulfill the order, so the model is already different. Secondly, let's look at the physical distribution of the systems and the length of the operations. It's likely that producing a quote will not necessarily be instantaneous – for instance, it may involve a manual step such as management approval. If you read last month's column, you'll think that the wide distribution of the three suppliers and the potential length of time to respond to a quote could present a problem in using two-phase commit-type transactions. You'd be right, since other assumptions made by the two-PC model are that each transaction is of short duration and the resources are relatively close together.

So in the Earth-centric world of that article, you would be using Java Message Service (JMS), to decouple the systems and writing application logic that could cope with multiple updates and compensating transactions. This month, however, the Sun is at the center of the universe

## Welcome to the Oasis

It's clear that when no one organization has end-to-end control over a business transaction, and that transaction could well be long-lived, a new model of a technical transaction is needed. This model is known as the “saga” (or sometimes, cohesion) A saga is like a JTA in the sense that it ties a set of related actions together, allowing a system to effect rollbacks, commits, and so on. However, it differs from the JTA view of a transaction in that it makes no assumptions about being able to hold data in a locked state, due to the potential length of time it may be active. A saga can succeed even if some operations within it should fail. To achieve this, the `xa_start`, `xa_end`, `xa_prepare`, and `xa_commit` operations are replaced by `prepare`, `cancel`, `confirm`, `vote`, `enroll`, and `resign`, allowing updates to be associated and disassociated with a saga as well as expressing their wish as to the outcome. The final say as to the outcome, however, lies with the coordinator, who can selectively choose to ignore no votes according to the business semantics.

Using this transaction model, reliability can be

brought to highly distributed transactional systems. The other distinction between JTA/XA and sagas is that where XA provides a programmatic API for the transaction manager to call because they are closely coupled, sagas are widely distributed and a network protocol is needed rather than an API.

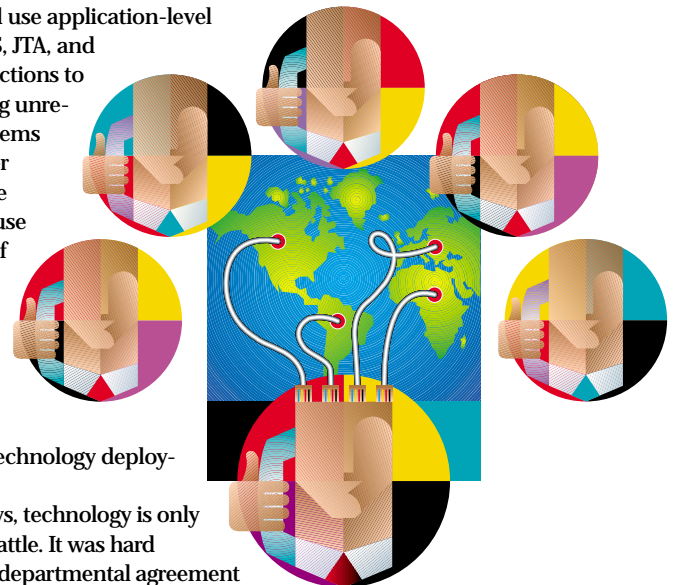
Since sagas are intended to span Web services, XML is a natural representation of the protocol elements needed to propagate them and their XML tags are added to business messages by transactional infrastructure, avoiding the need for application-level coding for transactional semantics. Since we are talking about interoperability here, there needs to be a standard for this protocol, and there is.

BEA's implementation of the saga concept is in the business-to-business component of the WebLogic Integration product. This supports a protocol, called XOCO, invented by BEA just for the purpose of propagating sagas. As well as implementing it in a product, BEA has submitted the specification to the Oasis standards body ([www.oasis-open.org](http://www.oasis-open.org)) where it is known as Business Transaction Protocol (BTP) and will soon be ratified as a release 1.0 standard (if it hasn't happened already).

## In Conclusion...

Today there are two choices for implementing widely distributed transactional systems. On one hand you can take the 'tried and trusted' Earth-centric approach and use application-level technologies like JMS, JTA, and compensating transactions to avoid closely coupling unrelated businesses' systems together. On the other hand, you can revolve around the Sun and use the new generation of infrastructure support for sagas and allow the infrastructure bar to be raised again to gain more business value, more quickly, from your technology deployments.

Either way, as always, technology is only a small piece of the battle. It was hard enough getting inter-departmental agreement within a company, now imagine that problem spread over multiple organizations. In the end, however, the business drivers for agreement and standardization – reduced cost and increased efficiency – will become irresistible, and that is the beauty of building on an app server – your logic is built and deployed now, and at the same time positioned for the future. ●



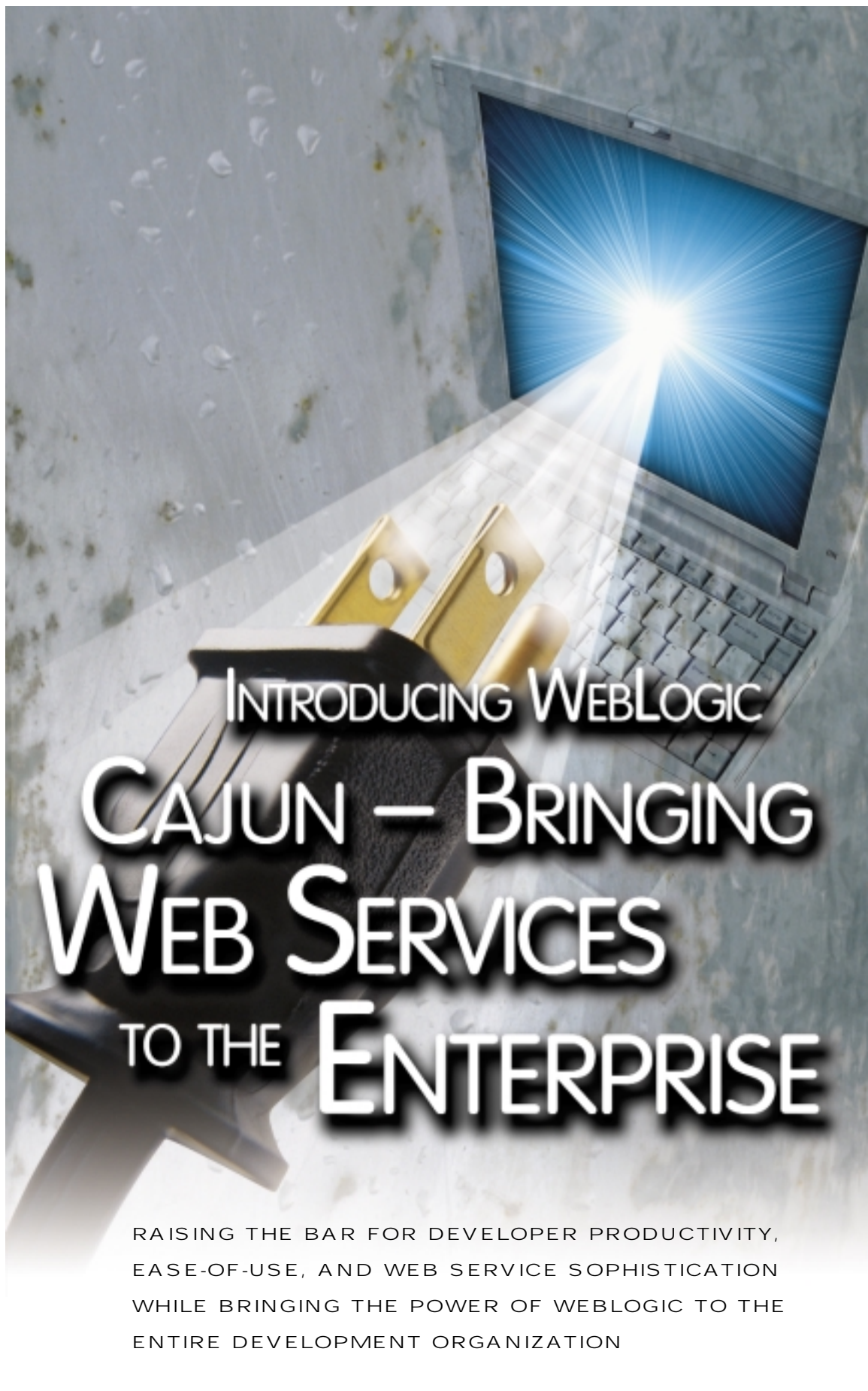
BY  
CARL SJOGREEN

**AUTHOR BIO...**

Carl Sjogreen, a product manager at BEA Systems, has been involved with XML, Web services, and developer tools since 1998 when he founded Transformis, a software startup specializing in XML tools. Passionate about bringing new technologies to the masses, Carl represents the developer community on the WebLogic Workshop product management team.

**CONTACT...**

CarlS@bea.com



# INTRODUCING WEBLOGIC CAJUN — BRINGING WEB SERVICES TO THE ENTERPRISE

RAISING THE BAR FOR DEVELOPER PRODUCTIVITY,  
EASE-OF-USE, AND WEB SERVICE SOPHISTICATION  
WHILE BRINGING THE POWER OF WEBLOGIC TO THE  
ENTIRE DEVELOPMENT ORGANIZATION

While Web services possess the potential to completely change how applications and organizations are integrated, capitalizing on this innovative technology hasn't been easy. To truly leverage the potential of Web services, you need both an architecture that can handle enterprise integration challenges and a framework that enables developers possessed of varying skillsets to work together. BEA Systems is launching a new product designed to solve these problems – "Cajun". Cajun makes it incredibly simple for application developers to build sophisticated Web services.

This article introduces "cajun", and illustrates some of its major features and concepts. You'll see from the examples that "cajun" sets a new bar for developer productivity, ease-of-use, and Web-service sophistication.

### Enterprise Class Web Services

SOAP, WSDL, and UDDI are the three major standards that form the foundation of Web services. Leveraging XML and existing Internet standards, these specifications describe how XML messages can be sent between applications (SOAP), provide a mechanism for describing the operations an individual Web service supports (WSDL), and establish a registry of Web services (UDDI).

Based on these standards, the Web services marketplace is flourishing. You can leverage Web services that let you check stock quotes, exchange profile information between Web sites, and check the traffic on your favorite highway – regardless of the platform you're using.

To date, these simple, synchronous Web services have commanded the attention of the marketplace. But Web services represent much more than just a way to exchange simple information over the Internet; their real power lies in the enterprise. Web services are a cross-platform, standards-based way to integrate systems (inside and outside a company's walls) in a way that's flexible and designed for change. Building Web services for the enterprise requires the right architecture and approach, as well as the infrastructure necessary to support it. In particular, enterprise class Web services have three main characteristics:

**1. They're loosely coupled.** Loosely coupled applications are flexible and can accommodate change over time. In an IT environment in which you need to integrate different applications built by different teams on different schedules using different technologies and platforms, you need a way to minimize the impact of change. With Web services, you can integrate applications based on a "public contract" that describes the XML messages applications will exchange, but leaves the underlying implementation details to each application. As long as applications continue to honor their "contract" they can change at will without breaking the integration.

**2. They accommodate asynchrony.** The real world doesn't consist entirely of systems that can immediately respond to any request. Any integration architecture must encompass human users, outside resources, and legacy applications – all of which are inherently better suited to asynchronous communication in most application scenarios.

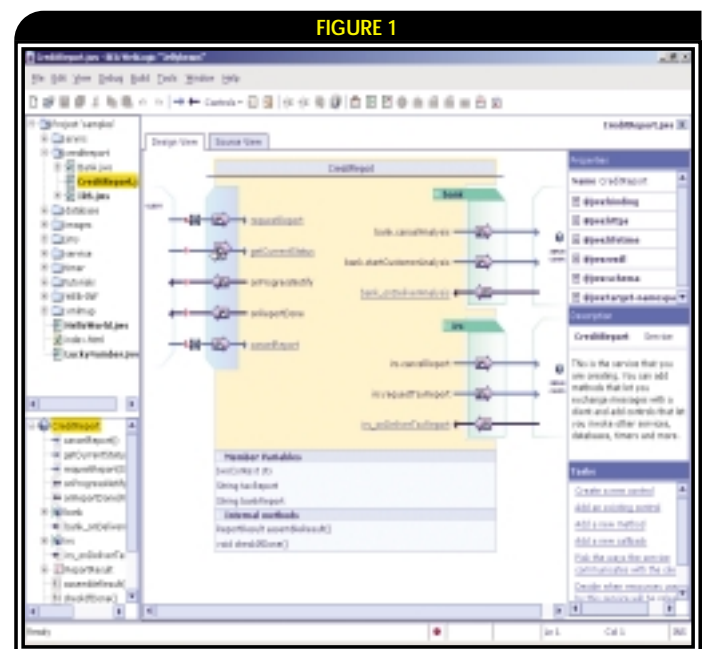
**3. They provide coarse-grained interfaces.** By integrating at a business level – exchanging documents like POs and invoices instead of moving one piece of data from one application to another – Web services can have greater flexibility when the underlying implementation changes. This coarse-grained exchange of data is also much more network friendly, unifying the internal and external integration worlds.

### The Enterprise Developer Organization

When used as an integration technology, Web services must be deployed on a trusted and secure, reliable, available, scaleable platform. "Cajun" provides the infrastructure to easily build Web services with these characteristics and deploy them on the WebLogic platform.

Bringing Web services to the enterprise also requires a platform that appeals to all constituents in the development organization. In most enterprises a small set of expert developers and architects is responsible for the overall enterprise architecture, creating reusable software components and building core business applications. For these users, the J2EE platform is the natural choice. The majority of developers, however, are focused on application and business logic rather than the architectural underpinnings of an enterprise. They're closer to the line of business, understand the requirements of an application, and are best at getting applications produced. These developers typically will write the code necessary to integrate existing components and expose new functionality through Web services.

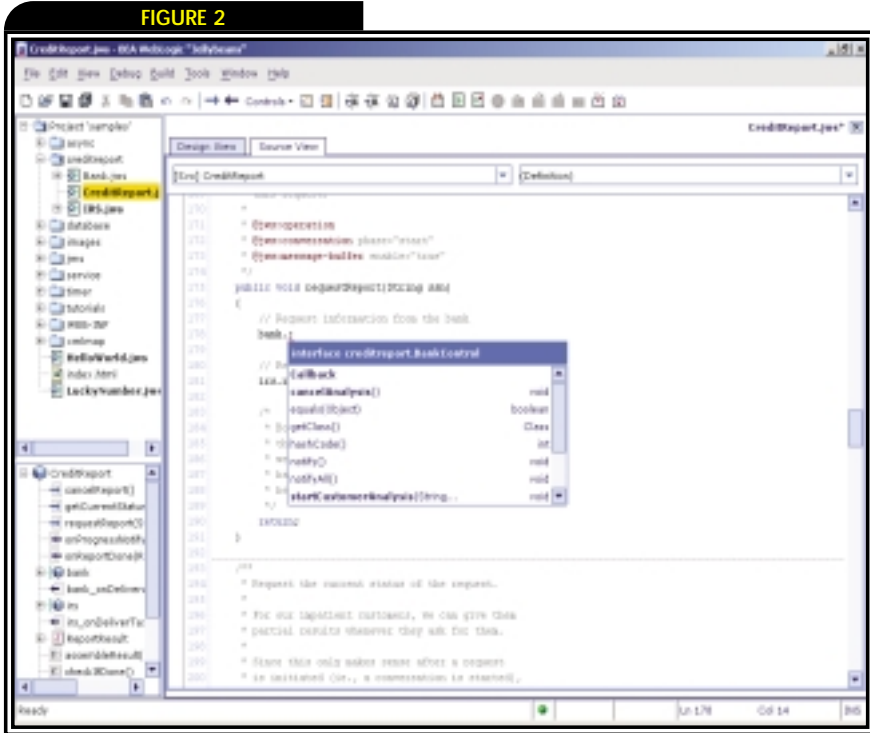
By reducing the barriers of entry to programming with J2EE, "Cajun" brings the entire development organization onto the same platform. Moreover, "cajun" enables developers to focus on the tasks they do best. J2EE experts can focus on building enterprise components and applications with the knowledge that all developers in the organization



The "Cajun" Design View



FIGURE 2



The "Cajun" source view provides all the standard Java IDE editing features and exposes the JWS annotations.

can easily access them. The rest of the development organization can then focus on creating the Web services and procedural code necessary to build composite applications from these components. Time and energy can be directed to the real meat of the application, rather than the details of the J2EE APIs necessary to access components or the steps involved in calling an EJB.

**"CAJUN" ARCHITECTURE**

"Cajun" includes two major components: a design-time tool that lets developers write Java code to implement Web services and a runtime framework that provides the Web services infrastructure,

testing, debugging, and deployment environment for "cajun" applications.

**THE "CAJUN" TOOL**

The "Cajun" Integrated Development Environment (IDE) provides a complete environment for developing a Web service application. Standard features such as project management, syntax highlighting, code completion, and integrated debugging are all included. In addition, "Cajun" provides a unique visual approach to Web services. Figure 1 shows the "Cajun" Design View. The Web service under development appears at the center of the screen. Messages sent from the Web service client appear as events that can be handled by writing business logic code and additional resources are exposed as controls on the right.

The design view lets users graphically create new methods, set properties on controls, and specify the overall structure of a Web service and its relationship with the outside world. The goal is to enable developers to focus on writing business logic – the code that's executed in response to each incoming method – not the machinery of typical Java programming. The "Cajun" IDE supports two-way editing so any changes made through the graphical environment are reflected immediately in code and vice versa.

**"Cajun" Controls**

Controls are a key innovation that give developers easy access to enterprise resources and J2EE APIs. These APIs provide a tremendous amount of power and flexibility but are typically overkill for common tasks. The sophistication of J2EE is one of the main barriers to entry for most developers.

"Cajun" controls overcome this problem by simplifying APIs and reducing the amount of object-oriented programming necessary to access external resources. Instead of creating new objects or interfaces or having to learn the necessary Java to access a database or look up an EJB, controls appear as local objects with a simplified set of methods that can be called, and properties that can be set.

For example, "Cajun" has a database control that simplifies the JDBC API. Database administrators can create a reusable database control that links SQL statements with Java methods in a simple declarative fashion. Developers can then use these components to access database resources with a simple function call. These controls can be further customized or extended simply by adding a new SQL statement.

A code example of a method in a database control that executes a simple select statement follows:

```
/**
 * @jws:sql statement:--
 * SELECT SCORE
 * FROM CUSTOMERS
 * WHERE SSN = {ssn}
 * :--
```

" To date. . . simple, synchronous Web services have commanded the attention of the marketplace. But the real Web services revolution is yet to come. "

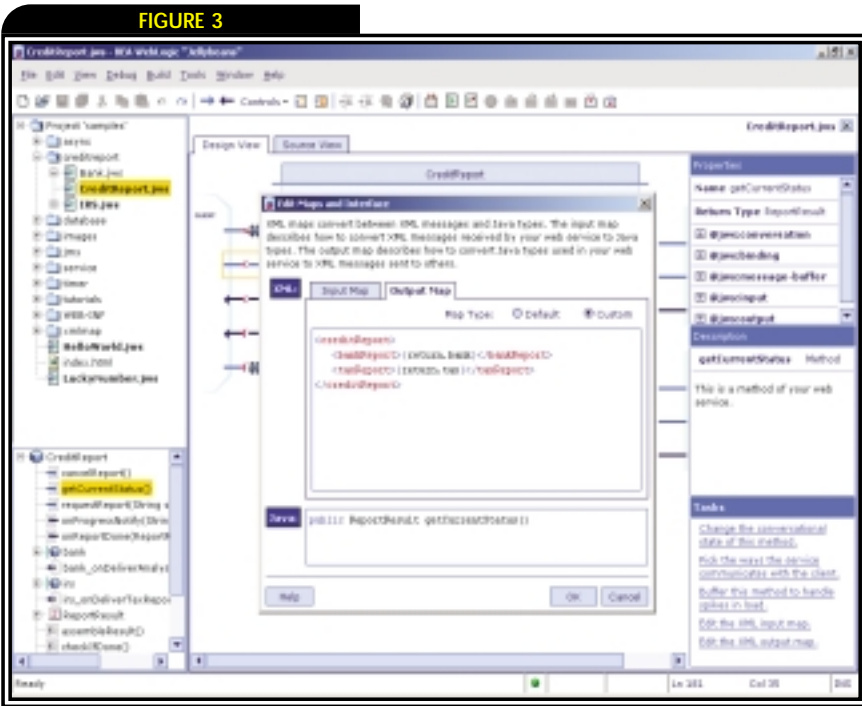
# Sitraka

[www.sitraka.com](http://www.sitraka.com)

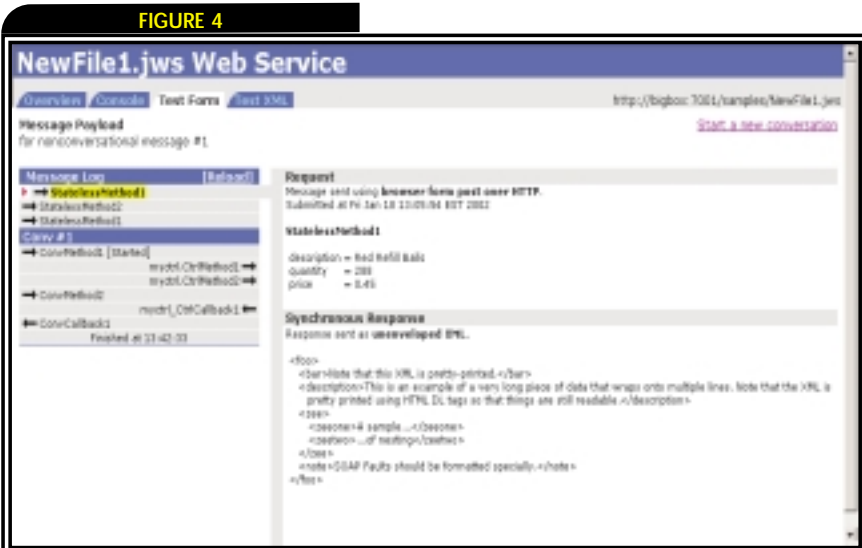
```

*/
int getScore (String ssn)
throws int getScore(String ssn);
    
```

A special “Cajun” javadoc annotation associates a SQL statement with a Java method. Once a method is on a control in this way, users of the control simply call the function to execute the SQL command. “Cajun” currently provides controls to access databases, other Web services, EJBs, JMS queues, and applications exposed through the J2EE Connector Architecture. It also provides a timer control to help manage asynchronous events. You can see how both a Database and a Timer control are used in Listing 1.



The Workshop map editor helps you map XML messages to Java objects.



This Web page is automatically generated for all “Cajun” Web services to enable easy testing.

## Java Web Service Files

The meeting place between the design time tool and run-time framework is the Java Web Service (JWS) file. JWS files are standard Java files with annotations (expressed in the JavaDoc syntax) to express additional functionality. This format enables developers to write standard Java code, but provides a mechanism for the “Cajun” tool and framework to assist with more sophisticated tasks. Annotations are used to display the Web service and its properties graphically, and by the framework to generate the EJB and J2EE code to execute the Web service. By moving code generation out of the tool and into the framework, developers never have to manage and maintain code they didn't write.

“Cajun” uses annotations in a JWS file to encode everything from what SOAP style a Web service uses to how XML messages are converted into Java objects (see Figure 2). This approach is also a simple but powerful way to smooth the ramp between developer skill sets. Developers work in a simplified programming environment but still write standard Java code. This allows a J2EE expert to easily add value over time. Instead of rebuilding an application written in a procedural language when it hits performance or scalability limitations, expert developers can tune the original Java code.

## The “Cajun” Framework

The “Cajun” framework is where the magic happens. Once a JWS file that contains all the business logic for a Web service has been created, the “Cajun” framework is responsible for generating the standard EJB code needed to implement it. Moreover, this framework exposes (through the annotations) functionality specifically designed to support building enterprise class Web services. In particular, “Cajun” assists by:

- **Managing asynchronous communication with a conversational metaphor.** The “Cajun” framework automatically manages asynchronous message correlation and state management across messages in a conversation. Users can simply mark methods as starting, finishing, or continuing a conversation and the “Cajun” framework takes care of the details. A unique ID is automatically generated to identify the conversation, and any state (class member variables) defined in the Web service is managed persistently with entity Java beans. Conversations can be either two-way, in which the client sends SOAP messages to the “Cajun” Web service and that Web service executes an asynchronous callback to the client (again with a SOAP message), or one-way through a polling approach.
- **Enabling loose coupling with XML Maps and XML Script.** “Cajun” uses simple, declarative XML maps to map between internal Java code and XML messages exchanged between Web services. Users simply indicate the structure of the desired method and associate XML fields

with Java variables. For more sophisticated mapping tasks, developers can include ECMA script code in a map definition. “Cajun” has extended ECMA script with XML extensions to simplify access to XML data and enable complex structural or data transformations.

- **Enabling availability with JMS queues.** To ensure availability under high load, Web services can take advantage of message buffers. Users simply mark a method as requiring a buffer, and the “Cajun” framework will handle creating a queue and wiring it to the Web service. This enables one-click access to sophisticated J2EE functionality.
- **Supporting the control architecture.** The “Cajun” framework instantiates the implementation of any control used by the JWS. The framework also hooks up any asynchronous callbacks using a simple naming convention.

The “Cajun” framework also enables a “write-and-run” approach to creating Web services. Much like Java Server Pages (JSP), JWS files are deployed the moment they’re placed in the WebLogic Server applications directory. The framework is responsible for creating the necessary EJB and J2EE components (session beans to host application logic, entity beans to manage state, message queues to enable scalability, and so on) and deploying them. This entire process is transparent to users, giving those unfamiliar with WLS or J2EE the ability to create a Web service that takes advantage of all the major performance and scalability attributes of the WebLogic platform – connection pooling, caching, security, transaction management, and so on.

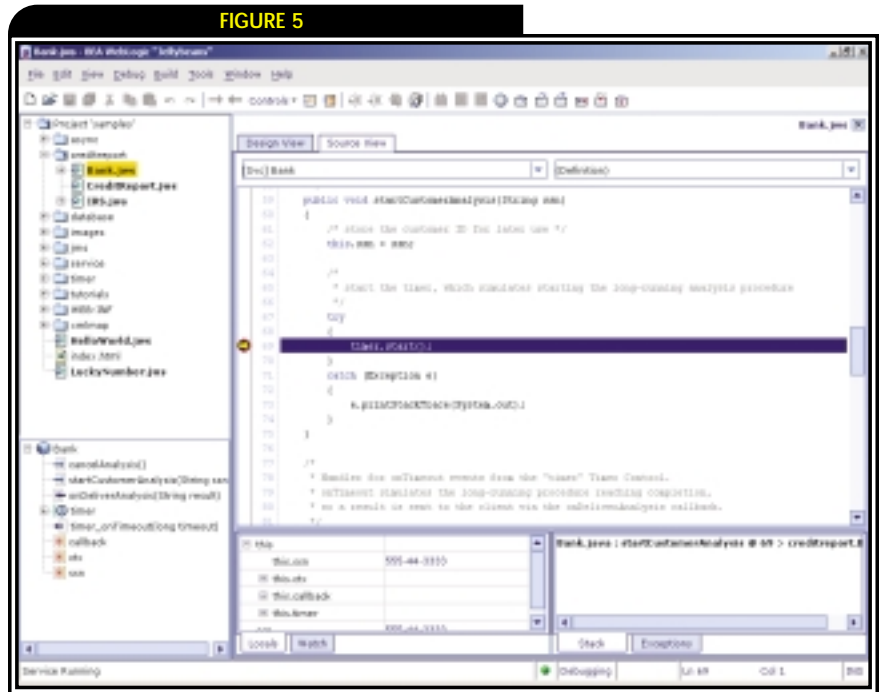
In addition to deploying the Web service, the framework creates a Web page for each service that provides all the information necessary to use the service in another application – a Web Services Definition Language (WSDL) file – and a Java proxy that makes it easy to invoke the Web service from another Java application. Part of this Web interface is a complete test and monitoring system so that developers can immediately test a Web service for correctness and monitor messages as they are received by the service.

Finally, the “Cajun” framework exposes a complete debugging environment that’s accessible through the IDE. This lets a developer execute a Web service (that may be hosted on another machine) and step through the Web service code as it executes, watch variable values, and test expressions.

## Final Notes

Listing 1 shows the complete source code for a very simple “Cajun” Web service.

Investigate.jws defines a Web service at a credit bureau. This service provides both a synchronous method – `investigateApplicant()` – and an asynchronous method – `investigateApplicantAsynch()` – to



“Cajun” provides an integrated debugging environment that allows developers to step through their code in response to incoming XML messages

determine whether an applicant should be offered credit. The synchronous method always returns a negative result immediately.

`investigateApplicationAsynch()` is the more interesting version. In this method the code starts a timer (using the “Cajun” timer control) to simulate the delay in accessing a back-end system. When the timer fires 10 seconds later, the `delayTimer_onTimeout()` is invoked by the framework. In this function, we look up the credit score of the customer using a database control and make an asynchronous callback to the client to return the result.

`CreditReportDB.jwi` includes a definition for a database control. Like the JWS files discussed above, JWI files are standard Java files with “Cajun” annotations that are used to define controls. Each method in the database control associates a Java method with an annotation used to define a SQL statement. The SQL statement can use variable names in squiggly braces to substitute parameters from the Java method.

## Conclusion

As you’ve seen here, “Cajun” provides a complete tool set for building Web services in a fast and easy-to-use fashion. With the JWS format, control architecture, and write-and-run approach to deployment, “Cajun” brings the power of WebLogic to the entire development organization.

Even more importantly, it provides developers with the infrastructure necessary to bring Web services into the enterprise – where the full potential of the technology can be realized.



## Listing 1

```

import weblogic.jws.*;
import weblogic.jws.control.*;

/**
 * This Web Service demonstrates some of the basic features of Cajun
 * using a credit bureau web service example.
 *
 * This Web Service supports two methods, a synchronous method
 * called investigateApplicant() that always returns a negative result
 * and an asynchronous method that uses a credit score retrieved from a
 * database to determine the result returned to the user.
 */
public class Investigate
{
    /**
     * This is a reference to a "Cajun" database control that store's
     * credit scores. We use the "Cajun" annotation below to signal to the
     * framework that this is a control instance.
     */
    @jws:control
    /**
     * private CreditReportDB creditReportDB;

    /* A local variable to store the applicant's SSN */
    private String m_applicantSSN;

    /**
     * A Timer control. This is used to provide an artificial delay to
     * simulate asynchronous processing. After its start() method is
     * called it will respond with a callback 2 seconds later.
     */
    @jws:control
    @jws:timer timeout="2 seconds"
    /**
     * private TimerControl delayTimer;

    /**
     * The callback interface used to invoke methods on the client of
     * this Web service.
     */
    public Callback callback;

    public static interface Callback
    {
        public void resultReady(String result);
    }

    /**
     * This is the simple, synchronous method that always returns a
     * positive response
     */
    @jws:operation
    @jws:conversation phase=stateless
    /**
    public String investigateApplicant(String applicantSSN)
    {
        return("applicant: " + applicantSSN + " is approved.");
    }

    /**
     * This method is asynchronous and starts a new conversation. It
     * stores the SSN in a class member variable and starts the timer.
     *
     * It also takes advantage of JMS queues and message driven beans
     * to ensure high availability using the message-buffer feature of the
     * "Cajun"
     * framework.
     */
    @jws:operation
    @jws:conversation phase=start
    @jws:message-buffer enable=true
    /**
    public void investigateApplicantAsync(String applicantSSN) throws
    Exception
    {
        m_applicantSSN = applicantSSN;

        delayTimer.start();
    }

    /**
     * This method is called when the timer fires and is responsible
     * for calling back using the Callback interface to the client of this
     * web service.
     * Depending on the result returned by the database, an appropriate
     * message
     * is returned
     */
    private void delayTimer_onTimeout(long arg0)
    {
        int score = creditReportDB.getScore(m_applicantSSN);

        if (score > 5)
            callback.resultReady("applicant: " + m_applicantSSN + " is
            approved.");
        else
            callback.resultReady("applicant: " + m_applicantSSN + " is
            denied.");
    }
}
Content-Type: text/plain; name="CreditReportDB.jwi"
Content-Description: CreditReportDB.jwi
Content-Disposition: inline; filename="CreditReportDB.jwi"

import weblogic.jws.*;
import weblogic.jws.control.*;

/**
 * Defines a "Cajun" database control.
 *
 * The @connection tag indicates which WebLogic data source will be used
 * by this database control.
 */
@jws:connection data-source-jndi-name="cgSampleDataSource"
/**
public interface CreditReportDB extends DatabaseControl
{
    static public class Customer
    {
        public String ssn;
        public String name;
        public int score;
    }

    /**
     * Each of the following methods contain an @sql annotation that
     * defines a SQL statement and associates it with a Java method
     * signature.
     * Braces {} in the SQL statement are substituted with variables from
     * the Java
     * method signature.
     */
    @jws:sql statement:--
    * CREATE TABLE customers (
    * ssn VARCHAR(40),
    * name VARCHAR(40),
    * score INT)
    * :--
    /**
    void createTable();

    /**
    @jws:sql statement="DROP TABLE customers"
    /**
    void dropTable();

    /**
    @jws:sql statement="SELECT SSN, NAME, SCORE FROM CUSTOMERS WHERE
    SSN = {ssn}"
    /**
    Customer findCustomer(String ssn);

    /**
    @jws:sql statement="SELECT SCORE FROM CUSTOMERS WHERE SSN =
    {ssn}"
    /**
    int getScore(String ssn);

    /**
    @jws:sql statement="INSERT INTO customers (ssn, name, score)
    VALUES ({ssn}, {name}, {score})"
    /**
    int insertCustomer(String ssn, String name, int score);
}

```

# Precise Software

[www.precise.com/wldj](http://www.precise.com/wldj)

# Caching

## INCREASE PERFORMANCE WITH A CAREFUL INVESTMENT IN DESIGN AND ARCHITECTURE

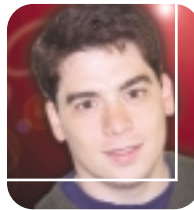
BY SAM PULLARA

**A**PPPLICATION SERVER PERFORMANCE. Database performance. Hardware performance. These are numbers measured in the popular press, although in most situations they have little to do with your application's real-world performance. The number one way to increase performance, the thing that gives you the biggest boost, is caching. Caching in every tier is becoming more and more prevalent. On the front end, we have caching proxy servers like Squid and AOL. On the back end, we have databases and file systems that are caching our data. This month, I'm going to talk about caching in the Web application layer of the middle tier, where your application meets the Internet.

In WebLogic Server 5.1, the JSP cache tag was introduced for caching in the Web application tier. This tag allows you to cache both input (values of variables) and output (typically, the generated HTML). Ideally, these tags can be used so you have what I call "exact caching." In other words, even though caching was introduced, no change in the functionality of the application was made because the data in the cache is always the same as the real data. This is similar to how caches are used on CPUs, databases, and file systems. Just because something is cached, it doesn't mean that it's out of date. This isn't the only way to architect your caching layer. It's also possible to analyze the system and find places where you'd rather have an out-of-date answer right now, rather than an up-to-date answer after blocking the user for a time.

Here's an example of "exact caching." On my Web site I built a DVD check-out service for the people who live in my loft building. At the time, I was experimenting with XML and wanted to use it to store all the information about the registered users and the DVDs. Since I was just using the file system, parsing the XML all the time was time consuming. However, since I knew when the application was updating the XML files, I could use the cache tag to store the results of the parse and only flush the cache when they changed. This led to an order-of-magnitude increase in speed, with no loss of functionality. Voila, "exact caching."

Similarly, in the first application of the cache tag in production, a customer used the tag to cache the content from their content management system. Since they could cache based on variables in the



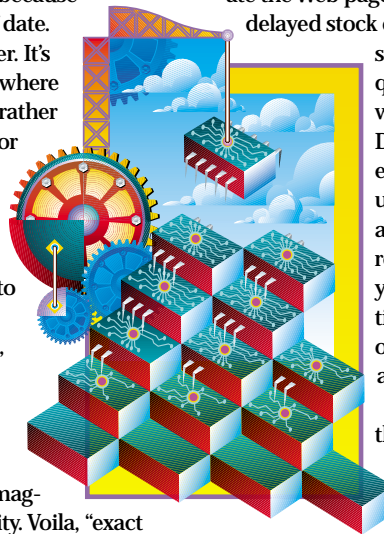
page, they were able to only query the update time of the document in question on each request, instead of pulling the whole document over to the file system each time. This also garnered them an order-of-magnitude improvement in performance for the affected operations.

Much more difficult to architect properly is timeout-based caching. This is the normal caching that your browser and proxy servers do. When you get a Web page back from a Web site, a time stamp is returned that tells you when the page might be out of date; it's stored in the "Expires" header. If you request the page before that time, your browser will first go to its local cache for the contents (unless you force it to return to the Web site). Obviously, this method assumes that all content on the page expires at the same time and that your primary concern is reducing the bandwidth load on the server. For many application server deployments, the primary concern isn't bandwidth, but reducing the load on the servers themselves. That's where caching in the middle-tier has its sweet spot. Timeout-based caches assume that the application developer has special knowledge about the time sensitivity of the data that is being used to generate the Web page. For instance, you might be offering 20-minute delayed stock quotes and you might decide that the most popular

stocks should be cached so that when users request a quote they are given the same one as everyone else who asks within some specified period of time. Determining that period is the important part of the exercise. Let's say you decide that you're only going to update the cache every 5 seconds; the quotes are already 20 minutes old anyway. If you're getting 10 requests per second for a particular ticker, that means you get 50 requests in 5 seconds, so you're doing 50 times less work to get each quote minus the cache overhead! Quite a savings for very little loss in functionality.

As you can see, caching can tremendously improve the performance of your Web applications. Using cache tags and cache filters (new in 7.0), you can build very robust, performant, scalable applications with only a bit more careful investment in design and architecture. *We have only touched the surface* of what you

can do with caches in the Web application layer in this article; you can find out much more about caching using these tools on my Web site at [www.sampullara.com/caching](http://www.sampullara.com/caching).



### AUTHOR BIO...

Sam Pullara has been a software engineer at WebLogic since 1996 and has contributed to the architecture, design, and implementation of many aspects of the application server.

CONTACT: [sam@sampullara.com](mailto:sam@sampullara.com)

# Borland

[www.borland.com](http://www.borland.com)

# Resonate, Inc.

[www.resonate.com](http://www.resonate.com)

# Resonate, Inc.

[www.resonate.com](http://www.resonate.com)



BY  
JIM RIVERA

**AUTHOR BIO...**

Jim Rivera is a product manager for WebLogic Server focusing on Web services, XML, EJB, and caching. He is lead product manager for the WebLogic Server 7.0 release.

**CONTACT...**

[jimr@bea.com](mailto:jimr@bea.com)

**IT'S HERE!**

INTRODUCING  
**BEA WEBLOGIC SERVER 7.0**



Of the many challenges facing today's IT system architects, two stand out as being the most common and strategic:

1. Integrating disparate applications and platforms to fully leverage data and software investments
2. Providing an enterprise-class framework that ensures reliable, available, scalable, and secure applications.

BEA WebLogic Server 7.0, scheduled for release early this spring, gives developers the tools necessary to face such IT challenges. This article discusses how these challenges are solved by some of the new features in WebLogic Server 7.0.

### Application Integration

Over the last couple of decades, there have been massive investments in enterprise software applications. In recent years, we've seen the adoption of ERP (Enterprise Resource Planning) packages by companies looking to automate their back-office processes and provide operational transparency to enterprise data. These concepts have also driven the adoption of CRM (Customer Relationship Management) and SCM (Supply Chain Management) packages to extend a company's reach to include its customers and partners.

Today, the typical enterprise is made up of a vast number of dis-

parate systems purchased and implemented throughout various stages of software evolution. These systems are implemented on a variety of software and hardware platforms without a standard protocol or data model for communication between applications. The inability of these systems to share services and data has resulted in fractured, disconnected silos of applications.

The successful company of tomorrow is the one that's able to integrate these applications to share data and services, leverage existing investments, and provide a complete 360-degree view of the business. To solve these problems, WebLogic Server 7.0 provides significant enhancements to the existing WLS suite of integration capabilities.

### WebLogic Web Services

Everyone is talking about Web services...but what exactly are they? At a high level, Web services enable application integration by leveraging commonly understood technologies like XML and HTTP. Web services – and associated specifications like SOAP (Simple Object Access Protocol), WSDL (Web Services Description Language), and UDDI (Universal Description, Discovery, and Integration) – define the conventions used for exposing and defining a service interface, invoking a service, and discovering what services are available for consumption.

Since version 6.1, WebLogic Server has supported an "export from Java" model in which users write a Web service implementation as an Enterprise Java Bean (EJB) and the server handles the details of generating the WSDL file that describes the service and marshalling and unmarshalling the SOAP request to the back-end EJB. This eliminates the need for J2EE developers to be experts in SOAP, WSDL, or XML.

WebLogic Server was also the first platform to address the need for asynchronous Web services – a critical aspect of application-to-application communication, particularly when you must communicate with a legacy application that may have significant constraints on throughput. In this scenario, a synchronous RPC (Remote Procedure Call) invoke can have a devastating impact on application performance and scalability. WebLogic Server allows developers to implement Web services using Stateless Session EJBs (for RPC invokes) and JMS consumers such as Message Driven EJBs (for asynchronous processing).

In WebLogic Server 7.0, the Web services container has been enhanced to provide additional flexibility and to ensure interoperability with other key Web service vendors.

Web Services are implemented using three types of components:

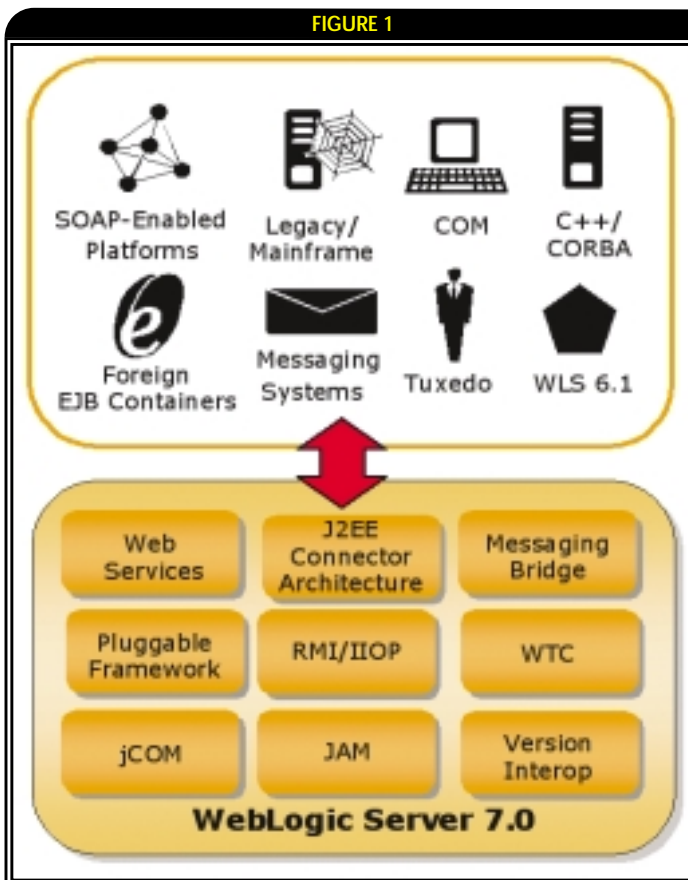
- **Back-end J2EE components:** for example, EJBs
- **Interceptors:** for pre- and post-processing SOAP requests
- **Serializers/deserializers:** for converting data between XML representation and Java objects

### BACK-END J2EE COMPONENTS

In WebLogic 7.0, you can implement a single Web service with multiple backend components. Each operation on the Web service is associated with a particular method on a back-end component via the Web service deployment descriptor. This provides greater flexibility and a looser coupling between back-end logic and the interface that's exposed.

### INTERCEPTORS

Interceptors are optional components that provide additional flexibility to help ensure the separation of business logic from the details of the messaging infrastructure. In general, WebLogic Server shields developers from having to work directly with SOAP and XML; however, some applications will require access to the message level (for example, to read and write SOAP headers). For these, interceptors can be



INTERGRATION

invoked before and after the back-end component and provide complete access to the SOAP and HTTP requests and responses through a simple API.

#### SERIALIZERS/DESERIALIZERS

Since a Web service implementation is written in Java, the mapping of XML parameters to Java objects is a critical aspect of the Web services container. WebLogic Server provides built-in support for simple Java types that can be straightforwardly mapped into SOAP-specified types as defined in the JAX-RPC specification. For applications that use more complex or user-defined types, the WebLogic Autotyper utility can be used to generate a serializer and deserializer that will handle the conversion at runtime. As input, WebLogic Autotyper will accept XML (or Java) representation of the types and will generate the equivalent Java (or XML) representation along with the serializer/deserializer.

The client-side model is based on the JAX-RPC specification. WebLogic Server can generate a thin Java proxy that will marshal and unmarshal SOAP requests and handle XML/Java conversion. An SSL client can also be generated for secure communication.

WebLogic Server's new Pull Parser improves the performance of Web services applications by allowing developers to target specific portions of an XML document and avoiding the need to parse the entire document, as is the case with SAX and DOM parsers.

Additionally, WebLogic Server 7.0 includes the ability to host a UDDI registry for publishing your Web services and a graphical utility for searching and modifying UDDI registries.

#### J2EE CONNECTOR ARCHITECTURE

The J2EE Connector Architecture (CA) defines a standard way to integrate J2EE application servers with an enterprise's existing legacy applications. The CA specification defines a set of system-level contracts between the application server and a legacy application-specific resource adapter - i.e., a software driver that's specific to an underlying legacy application.

WebLogic Server 6.1 provided an implementation of the pre-final J2EE CA 1.0 specification. In WebLogic Server 7.0, this implementation is updated to comply with the final version of the specification. There are also additional enhancements to connection and security management for the J2EE CA adapters.

#### WebLogic jCOM

WebLogic Server 7.0 includes WebLogic jCOM, a bi-directional COM-Java bridging tool. Using WebLogic jCOM you can access Component Object Model (COM) components as though they were Java objects, and you can access pure Java objects as though they were COM Components.

#### Foreign Message Service Bridge

A bridge is provided for a connection between

WebLogic JMS (Java Message Service) and foreign messaging service providers. This pluggable architecture supports store-and-forward between two messaging systems, mapping of message destinations, reconnection policies, and synchronous and asynchronous support. An adapter for interoperating with MQSeries is also included.

#### WebLogic RMI/IIOP

The new version includes many enhancements to the RMI/IIOP implementation, such as bi-directional support for transaction and security context propagation. This support is leveraged for interoperability with foreign EJB containers, with BEA Tuxedo via the WebLogic-Tuxedo Connector (WTC), and with legacy CORBA applications. Also available is a thin C++ client that supports clustering and failover of WebLogic Server services.

#### WebLogic JAM (Java Adapter for the Mainframe)

WebLogic JAM now supports bi-directional transactions with the CICS and IMS applications on the mainframe.

#### Version Interoperability

Since WebLogic Server 6.1, BEA has made a commitment to interoperability between server versions. WebLogic Server 7.0 is the first version able to interoperate with the previous version, WebLogic Server 6.1, over both T3 and IIOP protocols. This means that WebLogic Server 7.0 clients will be able to invoke 6.1 services, such as EJBs, using RMI, and vice versa. Additionally, a single administrative domain can contain a mix of 6.1 and 7.0 servers. This provides flexibility in determining a migration schedule for your applications.

#### Enterprise-Class Services

J2EE has come a long way in providing the building blocks necessary to develop enterprise applications. With the additions of JMS, J2EE CA, and significant enhancements to EJB, J2EE 1.3 provides a rich set of APIs. These services offer sophisticated component models, asynchronous processing and messaging, integration, distributed transactions, data access, and presentation capabilities. WebLogic Server 7.0 is fully compliant with the J2EE 1.3 specification.

However, enterprise applications also require security, performance, reliability, availability, and scalability. It's critical, too, that the sheer complexity of implementing and managing these services doesn't end up inhibiting their use - today's enterprises must maximize the productivity of engineering resources. WebLogic Server 7.0 provides these enterprise-class services within a single, highly usable platform.

#### New Security Framework

The security framework has been completely redesigned to offer a level of flexibility and control never before available with any application server



platform. The pluggable, modular design exposes a set of Service Provider Interfaces (SPIs) for authentication, authorization, auditing, and PKI management. Not only does this version provide fully featured implementations of each SPI, but it also allows modules from third-party security vendors to plug right into the WebLogic Server framework. Administration tools from these third-party security vendors can be seamlessly integrated within the extensible Administration Console. Multiple modules can be associated with different resources within your system, allowing you to take full advantage of the value-adds offered by your favorite security vendors and manage them all within a single management system.

WebLogic Server 7.0 offers a dynamic, role-based authorization scheme that can be applied to all J2EE and non-J2EE resources. You're no longer constrained by the limitations of J2EE's declarative security model. The authorization module includes an embedded entitlement engine that allows you to easily create prose-based rules for dynamically assigning roles and calculating access privileges. The application developer is freed from having to write code to implement complex security policies, leaving the task of securing applications to administrators and security experts. Entitlement rules can be based on an extensible list of operations and parameters such as time, identity, user profiles, and invocation parameter values.

All user profile and entitlement data can be stored in WebLogic Server's internal system data store, a scalable data store optimized for quick reads. In addition to the system data store, users can easily configure one or more LDAP stores to provide a single unified user profiling system from multiple back-end stores. WebLogic Server 7.0 also supports JAAS (Java Authentication and Authorization Service) and bundles an SSL implementation from Certicom for full JSSE (Java Secure Sockets Extension) support.

## Scalable and Highly Available JMS

WebLogic Server 7.0 introduces the concept of distributed destinations. Administrators can set up a single virtual destination that is mapped to multiple destination instances deployed on multiple servers within a WebLogic cluster. When a message producer sends a message to the virtual destination, WebLogic Server will determine the actual destination based on configurable load-balancing algorithms. Message consumers are "pinned" to a given destination at creation time. If a server fails, incoming messages will continue to be sent to other destinations hosted on healthy servers.



## Migratable Services

Some services, by definition, must exist as a singleton within a WebLogic Cluster. Things like transaction logs and JMS destination instances must maintain data integrity and there-

fore exist in only one place within a cluster. This helps to avoid the need to synchronize data across servers, which can result in excessive server-to-server cross-talk that could adversely affect scalability. In the event of a server failure, singletons can be migrated from a sick server to a healthy server, ensuring the availability of the service.

## Administration Enhancements

Cluster management has been greatly simplified. Configuration wizards are available to easily create new domains, clusters, and configurations. WebLogic Server 7.0 also introduces new server health monitoring metrics. These metrics and events can be used by the WebLogic Server Node Manager and/or HA (High Availability) frameworks to determine when a given server instance should be considered "dead." Some of the metrics that can be reported are heartbeat status, execute queue growth rate, thread availability, and denial-of-service monitoring. Using standard JMX interfaces, administrators can set up a notification system to inform them of specific events, such as when a server needs to be restarted or when a particular service should be migrated to a healthy server.

The WebLogic Server Administration Console interface has been enhanced for ease of use – and to support massive deployments. In addition, many new configuration properties can be set dynamically without restarting the server. Extensibility of the Administration Console allows for products offered by BEA partners to fully integrate their management tools into the WebLogic Server Administration Console. Providing a common framework and interface for all administration tasks throughout your system can significantly reduce system management complexity.

Additional administrative control over server startup and shutdown is now available. The concept of a "server life cycle" that clearly defines the various stages a server goes through as it boots up and shuts down has been introduced. Each of the server subsystems has been designed to behave appropriately within the various stages. Using the Administration Console or command-line tools, a server can be brought up in a suspended state. In this state, the server is fully administrable but doesn't claim any resources that may be shared with other servers and doesn't process requests from clients. It can, therefore, act as a hot standby and be brought online very quickly.

Life cycle support also allows a server to be grace-

" The successful company of tomorrow is the one that's able to integrate [its] applications to share data and services, leverage existing investments, and provide a complete 360-degree view of the business."



fully suspended without dropping in-flight work. The server will reject new requests while completing existing ones. Also, managed servers have greater independence from the Administration Server in WebLogic Server 7.0. They can be started and shut down using simple command-line utilities, and cache a copy of the most recent configuration, which can then be used if the Administration Server is unavailable.

Support for multiple Network Interface Cards (NICs) allows for an administration port to be defined on each server. This isolates administration traffic from other network traffic, thereby improving scalability.

### Caching Enhancements

Caching is a powerful tool for improving the performance and scalability of applications. Since WebLogic 6.0, WebLogic JSP Cache Tags have provided the ability to cache specific portions of JSP pages in memory and avoid unnecessary computation or trips to the back end. For very dynamic or personalized applications, this provides a flexible way to isolate data and calculations that can be safely cached.

WebLogic Server 7.0 provides additional functionality to easily configure caching for entire pages, URLs, and file types. Without requiring any code changes to the application, administrators can turn on caching and see immediate performance and scalability improvements.

Together with the WebLogic JSP Cache Tags, WebLogic Web Cache provides caching capabilities that are both easy to implement and extremely flexible.

Entity EJB caching has been a part of WebLogic Server since version 5.1. The familiar read-only and read-mostly caching strategies provide significant performance improvement when EJB data doesn't change often and the consequences of stale data are insignificant. The enhancements allow for concurrent access of read-only entity beans, thereby improving scalability.

Optimistic concurrency and automatic cache invalidation allow for a read-write caching strategy. With optimistic concurrency, EJBs can be cached between transactions. If the EJB is already cached in memory, the container won't read from the database at the start of a new transaction. Prior to committing the transaction, the container will verify the freshness of the data and roll back the transaction if data was changed in the database while the transaction was in progress.

This prevents the container from updating stale data.

To help prevent optimistic rollbacks, the EJB container identifies when an EJB has been updated and automatically invalidates any instance of a cached EJB within the cluster using multicast messages. These enhancements significantly increase the applicability of EJB caching technology.

### Developer Productivity Enhancements

"Cajun," the codename for a new development tool introduced in Carl Sjogreen's article (see page 10) in this issue, provides a new framework for the development of enterprise-class Web services applications on WebLogic Server. "Cajun" is an IDE targeted at the "corporate developer" who prefers a GUI-based approach to application development. "Cajun" developers work within a higher level of programming abstraction, which shields them from the complexities of J2EE and object-oriented programming. This exposes the power of J2EE and WebLogic Server to a new class of developer.

For J2EE developers, the WebLogic Server Application Builder is a graphical utility that helps developers prepare Java files for deployment to the application server. The Application Builder introspects on Java implementation classes and through a guessing framework, derives default deployment descriptors. Developers can quickly validate and add to the deployment descriptors through a series of property sheets, preventing the need to edit the underlying XML files by hand. All of this gets wrapped into a deployable package and sent into the runtime environment for quick and easy deployment.

Many files make up an EJB: the implementation class, the home and/or remote interfaces, the deployment descriptors, and so on. WebLogic EJBGen (see the related article in *WLDJ*, Vol. 1, issue 2) is a utility that reduces the number of files to one, which greatly simplifies development and maintenance and allows the developer to focus on the business logic.

### Conclusion

The evolution of software technology has left many enterprises with a vast collection of discrete, disconnected systems. The complexities of application integration require a wide variety of solutions. From Web services to J2EE CA to RMI/IIOP, WebLogic Server 7.0 has the tools to help developers seamlessly integrate disparate systems and leverage their existing IT investments.

For mission-critical applications, you need a framework that will provide enterprise-class services in an easy-to-use package. J2EE 1.3 compliance ensures that you have a rich set of APIs to develop your applications. With the pluggable, flexible framework of WebLogic Server 7.0, you can easily build and manage applications that are reliable, scalable, and secure.



" WebLogic Server 7.0 is the first WebLogic Server version that will be able to interoperate with the previous version, WebLogic Server 6.1 over both T3 and IIOP protocols "

# ReportMill Software

[www.reportmill.com](http://www.reportmill.com)



INTERNATIONAL WEB SERVICES CONFERENCE & EXPO



INTERNATIONAL JAVA DEVELOPER CONFERENCE & EXPO



INTERNATIONAL XML CONFERENCE & EXPO



# FUNDAMENTALLY IMPROVING THE SPEED, COST AND FLEXIBILITY OF BUSINESS INTEGRATION

Sponsored by:



Sponsored by:



Produced by:



## Web Services – Skills, Strategy, and Vision

For the developer, the latest tools and techniques...

For the architect, the latest designs...

For the VP/CIO, technology management issues

- Invaluable information in the form of discussions, presentations, tutorials, and case studies
- Unmatched Keynotes and Faculty - gurus in the Java, XML, .NET, and Web Services world
- The largest independent Web Services, Java, and XML Expos
- An unparalleled opportunity to network with over 5,000 i-technology professionals

## New in 2002!

Each track will feature "Hot Breaking" Sessions to keep attendees up-to-the-minute on Emerging Technologies and Strategies!

Featuring 2 Keynote Panels and Industry Perspectives from the Visionaries shaping Next Generation Technologies and Business Strategies



# 2002

## ONLINE EARLY BIRD REGISTRATION NOW OPEN! SAVE \$400

**JUNE 24-27**  
**JACOB JAVITS  
CONVENTION  
CENTER**  
**NEW YORK, NY**

**OCTOBER 1-3**  
**SAN JOSE  
CONVENTION  
CENTER**  
**SAN JOSE, CA**

### FOR MORE INFORMATION

SYS-CON EVENTS, INC  
135 CHESTNUT RIDGE RD.  
MONTVALE, NJ 07645

201-802-3069  
WWW.SYS-CON.COM

#### Who Should Attend...

- Developers, Programmers, Engineers
- Senior Business Management
- Senior IT/IS Management
- i-Technology Professionals
- Analysts, Consultants

#### Who Should Exhibit...

Java, XML, Web Services, and .NET Technology vendors, staking their claim to this fast-evolving marketplace



Web Services Edge Conference Committee



Alan Williamson  
Java Track Chair  
Editor-in-Chief  
*Java Developer's Journal*

#### Java Track

- Java 1.4: What's New?
- Building Truly Portable J2EE Applications
- J2ME: MIDlets for the masses
- WebScripting Technologies: JSP/CFML/Velocity
- Where is Swing in this New Web Services World?



Ajit Sagar  
XML Track Chair  
Editor-in-Chief  
*XML-Journal*

#### XML Track

- XML Web Services: Standards Update
- Using XML for Rapid Application Development and Deployment with Web Services
- Unlocking the Promise of XML: From Hype to How-To
- The Use of XML Technologies to Enhance Security
- Content Management with XML



Sean Rhody  
Conference Tech Chair  
Web Services Track Chair  
Editor-in-Chief  
*Web Services Journal*

#### Web Services Track

- Starting Out in Web Services: Fundamentals of Web Services (WSDL, UDDI, SOAP)
- Exploring .NET myServices Initiative
- Standards Watch: Reviews and Discussions of the Interactions of All the Relevant Standards
- Guarding the Castle: Security and Web Services
- The Microsoft Way: .NET, Passport, and other MS Technologies for Web Services
- Laying Down the Rules: Web Services and Business Process Engines
- Practical Experiences with Web Services and J2EE
- The Odd Couple: Making .NET and J2EE Work Together



Andy Astor  
IT Strategy Track Chair  
International Advisory Board  
*Web Services Journal*

#### IT Strategy Track

- Key Architectural Issues in a World of Web Services
- .NET vs J2EE – From Religious Wars to Fact-Based Decision Making
- Getting Started with Web Services
- Selecting a Framework: Toolkit, Platform, or Roll Your Own?
- Infrastructure Vendors—A Map of the World
- Extreme Programming and Web Services Projects: Defining ROI
- Standards You Need to Know About
- Best Practices You Need to Insist On
- Introduction to Business Rules and Rules Engines
- Panel Discussion- Web Service Business/Economic Models

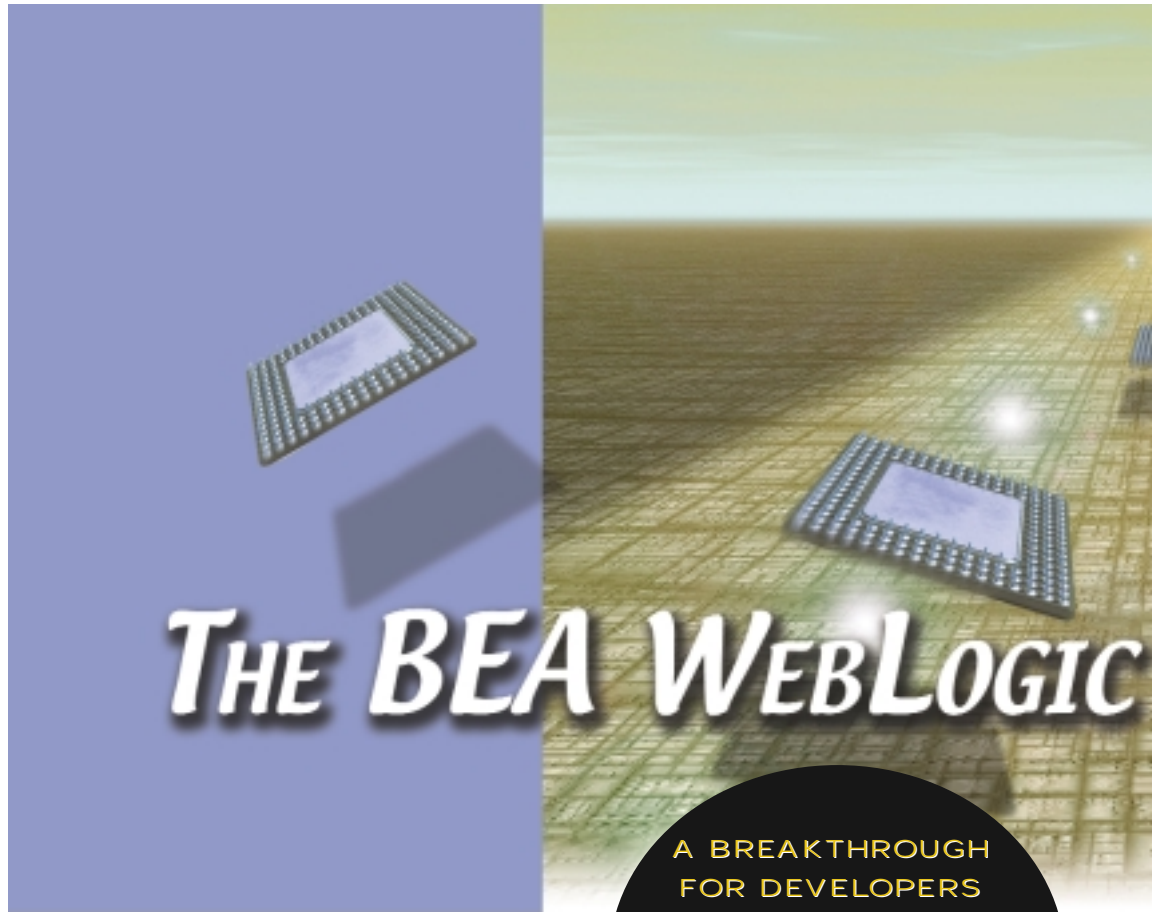


Jim Milbery  
Vendor Track Chair  
Product Review Editor  
*Java Developer's Journal*

#### Vendor Track

- .NET, J2EE, JMS and other Messaging
- Case and Development Tools
- The Use of Testing Tools
- How-to Demonstrations





A BREAKTHROUGH FOR DEVELOPERS WISHING TO LEVERAGE RICH, EASY-TO-USE APPLICATION INFRASTRUCTURE FUNCTIONALITY



BY  
WILL LYONS

**AUTHOR BIO...**

Will Lyons is a senior product manager for BEA Systems in the Servers and Integration Division and is the product manager for BEA's new application infrastructure platform product. He 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

**T**his spring, BEA will deliver a new, unified application infrastructure platform. The new product name had not been announced as of this writing, so we refer to it here simply as the BEA WebLogic Platform.

The new platform represents an integration and extension of the current WebLogic suite of products. Its delivery is an important milestone and opportunity for the WebLogic developer community. This article provides an overview of the release of the BEA WebLogic Platform, describing the process of technology platform innovation and adoption, and the technology disruptions that enable the emergence of an application infrastructure platform. It also discusses the components that are being integrated into the platform, current plans for new integration features, some directions for BEA's platform strategy going forward, and what this means for WebLogic developers.

**Technology Platform Innovation and Adoption**

A technology platform is an extensible technology package that, through breakthroughs in functionality and ease-of-use, enables new types of solutions to pervasive business problems, and achieves massive

adoption in the user community. It emerges when a new technology, or set of technologies, is created and becomes viable for volume usage. The essential innovation in the creation of the technology platform is the combination of these technologies in an easy-to-use package. This innovation becomes disruptive when a user community recognizes the extensibility of the package and its broad applicability to business applications. Killer applications drive mass adoption of the package and acceptance as a pervasive technology platform. Massive adoption leads to a positive feedback loop where an ecosystem of developers and technology partners enhance the value of the platform, further promoting its adoption by end-users and other developers and partners.

An example of the emergence of a technology platform is the emergence of the Windows desktop platform. In this case, technologies such as low-cost microprocessors, graphical user interfaces, and laser printing technologies were combined into affordable packages that made it easy for individuals



# PLATFORM

working on desktop systems to create and edit content using a WYSIWYG interface, and print output on local printers. Desktop publishing became one of the killer applications that foreshadowed widespread adoption of PCs and Windows-based technologies in business. This was followed by creation of Windows-based productivity applications and development tools, delivery of thousands of Windows-based packaged and enterprise applications, and the massive adoption of Windows-based desktop systems as a technology platform over the past decade.

## The BEA WebLogic Platform as THE Application Infrastructure Platform

The WebLogic Platform represents the emergence of an application infrastructure platform that is a disruptive innovation, offering explosive opportunities for enterprise developers and the software channel partner community. Our vision of an application infrastructure platform is a package that will enable all enterprise developers to build integrated, multi-tier applications more easily and rapidly. The ability to build highly reliable, available, and serviceable applications will be extended not only to professional developers with system programming and enterprise

architecture expertise, but also to corporate developers who are less familiar with advanced system programming techniques and more comfortable with GUI-based development.

The application infrastructure platform will enable enterprises to deploy and manage these applications more easily. It will provide a tightly integrated package with the functionality required to solve enterprise business problems, creating a seamless experience for developers and administrators.

The platform will be widely adopted and inherently extensible, enabling significant opportunities for developers and partners to customize and extend it with value-added services. Developer and partner value-add will contribute to platform adoption, leading to further opportunities and a reinforcing cycle of massive adoption.

The time has come for the emergence of an application infrastructure platform. Some of the key technologies and standards required for the application infrastructure platform exist today – Java and J2EE for application development on a wide range of databases, operating systems, networking protocols, hardware vendors, and legacy systems; infrastructure technologies that provide for application reliability, availability, and serviceability; tools for the creation and management of portal applications and business process management; standards for enterprise application integration; and B2B integration. Combined with these technologies is the emergence of Web services for loosely coupled application integration, and GUI-based tools for development of Web services applications. These technologies, taken together, offer the promise of easy development of robust applications that meet enterprise business requirements and can run across diverse enterprise IT environments.

However, to achieve the massive adoption of a technology platform, these technologies and their functionality must be integrated in an easy-to-use package, so the developer can focus on application development rather than technology integration. The requirements of such a package are well known in the software industry. They include coexistence and inter-operation of the software technologies provided in the package, physical packaging of the software into a single CD or download, a single installation program, a single licensing scheme, unified documentation, and a simple maintenance and release model. The integration of these technologies into a package that meets the above requirements is the disruptive innovation of the BEA WebLogic Platform.

## Features and Components

The WebLogic Platform integrates the features provided by current WebLogic products, including new enhancements, into a single product. The current products will become platform components, and the platform will become the focus of new product development and design. In the first release, the functionality is the sum of the product components described below, plus the integration features summarized in the next section. Note that the platform components will remain orderable and installable as individual products, and will provide a seamless upgrade path for users of current product releases.

### BEA WEBLOGIC SERVER 7.0

The next version of BEA WebLogic Server is a major release of the product, incorporating major enhancements in areas such as Web services, performance, usability, J2EE support, enterprise messaging, security, and administration. The WebLogic Platform provides the foundation for WebLogic Server 7.0.

“Cajun,” a new framework for the development of enterprise-class Web services applications (see Carl Sjogreen’s article on page 10), is targeted at “applications developers” who prefer a GUI-based approach to application development. “Cajun” abstracts away complex system programming functions and enables developers to focus on solving business application functionality and enterprise resource integration issues. Applications developed using the “Cajun” framework leverage the BEA WebLogic Server 7.0 runtime environment. (A related article on WebLogic Server 7.0 is on page 22.)

### BEA WEBLOGIC PORTAL

WebLogic Portal is a product based on WebLogic Server for creating a variety of portals, including employee, consumer, and partner portals. It provides a comprehensive set of foundation services that simplify the creation of complex portal sites, help to improve the user experience, and enable measurement of targeted user interaction to achieve business success. Intelligent administration capabilities save time and give users quick access to resources they need. Integration services enable a single view of the visitor with profile information gathered from distributed sources.

### BEA WEBLOGIC INTEGRATION

BEA WebLogic Integration, based on WebLogic Server, delivers application integra-

" The BEA WebLogic Platform represents the emergence of an application infrastructure platform that's a disruptive innovation, offering explosive opportunities for enterprise developers and the software channel partner community "

tion, business process management, and B2B integration functionality for the enterprise. It provides a standards-based "build to integrate" approach that enables companies to deploy new applications, and integrate them with existing ERP, CRM, and MRP applications, streamline complex business processes and connect with business partners.

The WebLogic Platform treats the above products as components of an unsurpassed application infrastructure platform that provides:

- Accessibility to developers with a range of system programming and architectural expertise
- Leadership reliability, availability, and serviceability
- Rich functionality, simplifying development of complex business applications
- Support for a wide range of industry standards
- Support for heterogeneous environments including a variety of hardware, operating systems, networking protocols, web servers, firewalls, and databases
- Out-of-the-box integration, ease of use and developer productivity

**BEA WebLogic Platform Integration**

To a significant degree, the above products already represent a highly integrated offering. Both WebLogic Portal and WebLogic Integration are based on the WebLogic Server foundation, and include WebLogic Server as part of their product offerings. In addition to the natural integration provided by sharing a common foundation, BEA product integration testing ensures that these products coexist and interoperate. They share a common installation and licensing technology, a common look and feel in documentation, and a common support infrastructure.

The enhancements provided in the WebLogic Platform represent an evolution of the existing integration provided by BEA, creating a more unified offering for developers, administrators, and partners. These integration features, described below, will be extended in future releases, along with functional enhancements to the software components, to ensure that the WebLogic Platform remains the leading application infrastructure platform in the industry.

**INTEROPERABILITY AND COEXISTENCE**

The application infrastructure platform software development process ensures that platform software components can coexist and interoperate in integrated applications. Developers can use the platform to build applications that seamlessly leverage the services offered by the individual components. For example, a developer can build a portal that will run in a clustered environment with standards-based database access and messaging, leveraging the proven technologies offered by WebLogic Portal and WebLogic Server. The "Cjun" development framework can be used to build reusable Web serv-

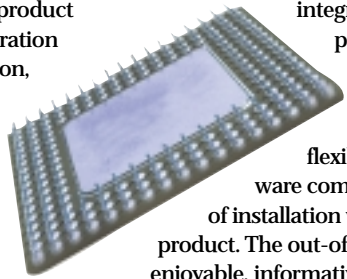
ices applications that are callable from the portal. The portal can also initiate workflows across packaged applications, or external trading partners, using features provided in WebLogic Integration. By using an integrated application infrastructure platform, developers can focus on building business logic and addressing business requirements, rather than integrating technology.

**COMMON PACKAGING**

The WebLogic Platform will be delivered as one set of software bits in a single software package. Users will be able to obtain the platform as a single Web download. BEA will also distribute the platform on a single CD, in a single physical package. Users won't be required to understand complex product combinations, and will not be required to order and physically procure multiple independent products that must be integrate a on target systems.

**COMMON INSTALLATION**

The entire application infrastructure platform offering will be installable via a single installation program. A typical installation will install all the platform software components, but users will be able to select individual platform components, or combinations of platform components to meet their individual business requirements.



The installation program will be integrated into the download process, enabling installers to download only those components they require. The installation program will support flexible configuration of the software components, and upon completion of installation will introduce the user to the product. The out-of-the-box experience will be enjoyable, informative, and comprehensive, covering the full range of application infrastructure platform functionality.

**COMMON LICENSING**

Use of application infrastructure platform components will be managed by a single license. The license will control usage of individual components, and, in some cases, usage of specific features offered by individual components. This provides a single point of control for management and administration of platform licenses, while enabling a rich functionality.

**COMMON DOCUMENTATION**

For users wishing to explore topics such as installation, application development, deployment, administration, or security, the platform will provide an integrated view of documentation across software components and features. Easy navigation for particular topics will be provided, as well as integrated search capabilities and a common look and feel to facilitate development of applications that require use of multiple application platform components.



**COMMON MAINTENANCE AND RELEASE MODEL**

The platform and its components will be maintained and upgraded as a single product offering. Service packs will combine maintenance of features across multiple components, with integration testing to ensure that platform components continue to interoperate and function together. Common upgrade and release migration policies will be applied to ensure predictable and manageable change as new features and functions are added.

These features represent some of the more obvious integration points that will promote product ease-of-use during the out-of-the-box experience, during the development of sophisticated applications, and during long-term maintenance and support. Underlying these features is a detailed product technology integration “under the hood” that reflects the inherent extensibility being built into the platform and being leveraged through reuse of software components across the platform.

**Strategic Directions**

The first release of the WebLogic Platform represents a breakthrough for developers building enterprise applications and wishing to leverage rich, easy-to-use application infrastructure functionality. Going forward, developers can expect to see continued enhancements both to the functionality provided by each of the components, and the integration provided across the components. Some of these enhancements will include a greater commonality of look and feel across platform components and user touch points, tighter integration of security and administration across the platform, extension of the Workshop development environment to leverage additional functionality, and further integration of end-user tools to provide a common user experience.

In particular, BEA will seek to extend the inherent extensibility provided in the WebLogic Platform. In addition to the obvious capability of building applications that run on the platform, example extensibility features provided in the release include the ability to plug

in alternative Web servers; the ability to use alternative development tools; replaceable XML parsers; an application integration framework; a replaceable, pluggable security infrastructure; and an extensible administration infrastructure. These features and others have enabled the emergence of a robust ecosystem of developers and partners who add value to the platform. BEA intends to place even greater focus on extensibility features that can be leveraged for our own development, and that can also be leveraged by our developer and partner community.

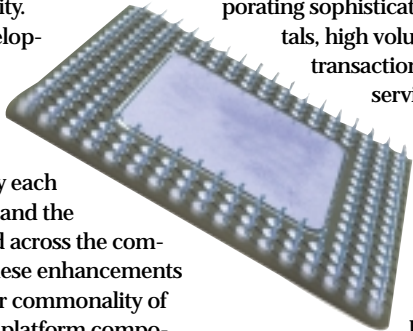
**What This Means for the WebLogic Developer**

A major focus of our platform strategy is to enable and unleash the talents of the WebLogic developer community. The platform will not only reinforce the WebLogic product features and capabilities you are leveraging today, and support seamless migration from existing WebLogic products, but will also offer new opportunities for you to leverage and extend the value of the platform in the future.

At a more detailed level, this means that BEA will provide you with the leading application infrastructure platform for a range of applications, from the most simple Web services application to the most complex application incorporating sophisticated end-user portals, high volume distributed transactions, composite Web services, complex workflows, and integration of existing enterprise applications or trading partner applications. BEA will make these capabilities accessible to

a range of developers, from the corporate developer requiring easy-to-use visual programming tools to the professional developer seeking to optimize application design through use of sophisticated architectures, system programming techniques, and sophisticated programming APIs.

On a strategic level, BEA's goal is to lead in the application infrastructure platform market. Enhancing the extensibility of the platform on an ongoing basis means a tremendous long-term opportunity for the WebLogic developer community.



**SUBSCRIBE AND SAVE**

**WebServices JOURNAL**

Offer subject to change without notice

<b>ANNUAL NEWSSTAND RATE</b>
<del>\$93.88</del>
<b>YOU PAY</b>
<b>\$69.99</b>
<b>YOU SAVE</b>
<b>\$13.89</b> Off the Newsstand Rate

**DON'T MISS AN ISSUE!**

Receive 12 issues of **Web Services Journal** for only **\$69.99!** That's a savings of **\$13.89** off the annual newsstand rate. Sign up online at [www.sys-con.com](http://www.sys-con.com) or call 1 800 513-7111 and subscribe today!

**In March WSJ:**

**Making Second-Generation Web Services Secure**  
First-generation Web services will use existing technologies... but what's next?

**Securing Web Services**  
Security is vital if Web services are ever to achieve mass deployment

**Scalable Web Services Using JMS and JCACHE**  
A solution to the problems inherent in delivering large-scale distributed systems

**Web Services' Impact on Business Process Management**  
The promise of a single solution for integration across multiple enterprises



# STRATEGIES & SOLUTIONS FOR UNWIRING THE ENTERPRISE

## International Wireless Business & Technology Conference & Expo

Wireless Edge will provide the depth and breadth of education and product resources to allow companies to shape and implement their wireless strategy. Developers, *i*-technology professionals and IT/IS management will eagerly attend.

### Who Should Attend

Mobile & Wireless Application Professionals who are driving their enterprises' wireless initiatives:

- Program Developers
- Development Managers
- Project Managers
- Project Leaders
- Network Managers
- Senior IT and Business Executives

### Plan to Exhibit

Provide the Resources To Implement Wireless Strategy  
The conference will motivate and educate. The expo is where attendees will want to turn ideas into reality. Be present to offer your solutions.

## C O N F E R E N C E T R A C K S

### TRACK 1: DEVELOPMENT

- WAP
- i-Mode
- Bluetooth / 802.11
- Short Messaging
- Interactive Gaming
- GPS / Location-Based
- Wireless Java
- XML & Wireless technologies

### TRACK 2: CONNECTIVITY

- Smart Cards
- Wireless LANs incl. Bluetooth
- UMTS/3G Networks
- Satellite Broadband

### TRACK 3: WIRELESS APPS

- Education
- Healthcare
- Entertainment
- Transport
- Financial Services
- Supply Chain Management

### TRACK 4: HARDWARE

- Cell Phones/WorldPhones
- PDAs
- Headphones / Keyboards / Peripherals
- Transmitters / Base stations
- Tablets

### TRACK 5: MANAGEMENT

- Wireless in Vertical Industries
- The WWW
- Unwired Management
- From 3W to 4W: Issues and Trends
- "Always-On" Management
- Exploiting the Bandwidth Edge
- Unplugged Valueware
- Wireless Sales & Marketing

FOR EXHIBIT & SPONSORSHIP  
INFORMATION PLEASE CALL  
**201 802-3004**



**Arie Mazur**  
President, Chief Executive Officer, and Chairman, Slingsoft



**Chris Bennett**  
Architect, Freedom Technologies



**Daniel Elliott**  
Senior Vice President, Mobile Business, CompuCom



**Douglas Lamont**  
Professor of Marketing, DePaul University



**Kenneth Leung**  
Business Development Manager, Retail and Emerging Technologies, IBM



**Keith McIntyre**  
Vice President and Chief Technologist, Stelcom



**Steve Milroy**  
Wireless Technologist, Immedient



**Tony Wasserman**  
Director, Mobile Middleware Lab, Hewlett-Packard Company

# Exclusive Sponsorship Available

Rise above the noise. Establish your company as a market leader. Deliver your message with the marketing support of



# ONLINE REGISTRATION NOW OPEN

[WWW.SYS-CON.COM/WIRELESSEGE2002](http://WWW.SYS-CON.COM/WIRELESSEGE2002)

Or, To Register By Phone  
Call: (201) 802-3069



	Development	Connectivity	Wireless Applications	Hardware	Management
<b>M A Y 8 , 2 0 0 2</b>					
8:30-9:45	Embedded Java (Bill Ray, Network 23 Limited)	A Real Time Model of (3G) UMTS Access Stratum (Niloy Mukherjee, MIT Media Laboratory)	Using Mobility to Streamline Traditional Business Processes (Dan Elliott, CompuCom)	Java Software Performance On Wireless Devices: Myths and Realities (Ron Stein, Nazomi)	LMDS - The Last Mile Enabler of Class 5 Revenues with VoIP (Ed Peters, Ensemble Communications, Inc.)
10:00-11:15	KEYNOTE				
11:30 A.M.	EXPO FLOOR OPEN				
LUNCH BREAK					
1:30-2:30	XML & Wireless Technologies (Karl Best, OASIS)	Securing Wireless Data Via Smart Card (Joseph Smith, New Dominion Software)	Collaboration for Wireless Warriors (Timothy Butler, SiteScape, Inc)	User Interactivity for Information Appliances (Arie Mazur, Slangsoft)	Leveraging Wireless In Customer Acquisition and Retention (Kenneth Leung, IBM)
2:45-3:45	KEYNOTE				
3:45-4:45	Developing Mobile Web Applications (Tony Wasserman, Hewlett-Packard Company)	In-building Wireless, the Next Frontier (Mary Jesse, RadioFrame Networks)	Turbocharge Mobile Applications with J2EE (Dr. Jeff Capone, Aligo, Inc.)	Cell Phones/ WorldPhones (Speaker TBA)	Mobilize Your Enterprise (Chris Bennett, Freedom Technologies)
5:00-6:00	VoiceXML Workshop (Bryan Michael, BeVocal)	Satellite Broadband (Speaker TBA)	Mobile Portals - The First Step Towards the Mobile Enterprise (Tony Wasserman, Hewlett-Packard Company)	PDAs (Speaker TBA)	Wireless Solution Return On Investment (ROI) In the Real World (Steve Milroy, Immedient)
<b>M A Y 9 , 2 0 0 2</b>					
8:30-9:45	Brew, I-Mode, WAP, and J2ME: How the Battlefield Is Shaping Up at the Start of the Mobile War (Reza B'Far, eBuilt, Inc.)	The Future of IP Mobility (Antti Eravaara, NetSeal, Inc.)	Multimodality: Revolutionizing Our Wireless Lifestyles (Arvind Rao, OnMobile Systems)	Tablets (Speaker TBA)	Marketing Value in Automotive Telematics Through Mobile Communications (Douglas Lamont, DePaul University)
10:00-11:15	KEYNOTE				
11:30 A.M.	Guide to Developing Applications with Micro Java (Kurt Baker, Kada Systems)	EXPO FLOOR OPEN			
LUNCH BREAK					
1:45-3:00	Building Secure Mobile Solutions (Keith McIntyre, Stellcom)	Smart Card Communications (Bill Ray, Network 23 Limited)	Developing New Applications Using VoiceXML (Jonathan Taylor, Voxeo)	PDAs (Speaker TBA)	Policies and Profiles: The Key to Mobile Data Services (Doug Somers, Bridgewater Systems)
3:15-4:30	Bluetooth™ Wireless Technology and Java™ Technology (Michael Portwood, Exuberance, LLC)	UMTS/3G Networks (Speaker TBA)	Instant Messaging and Wireless Computing Collide (JP Morgenthal, iKimbo)	Transmitters / Base Stations (Speaker TBA)	Total Business Solutions B2E (Speaker TBA)
4:45-6:00	Creating Carrier Optimized Wireless Internet Applications (David Young & Victor Brilon, Lutris)	Wireless LANs/Bluetooth (Speaker TBA)	Rapid Development of Successful Wireless Applications (Rod Montrose, AVIDWireless)	Headphones / Keyboards / Peripherals (Speaker TBA)	Wireless Sales & Marketing (Speaker TBA)

**REGISTER BEFORE**

**March 1**

**And**

**Save**

**Up To**

**\$400**

**Plan to Attend the 3-DAY Conference**

PRODUCED BY



May 7 will feature full-day tutorials. Consult [www.sys-con.com/wirelessege2002](http://www.sys-con.com/wirelessege2002) for up to-the-minute conference news.



# BEA WebLogic Portal TECHNICAL OVERVIEW



BY  
DMITRY DIMOV

### AUTHOR BIO...

Dmitry Dimov is the product manager for theE-Commerce Application Components Division at BEA Systems. He is part of the team responsible for the design and release of BEA WebLogic Portal. Previously, Dmitry was responsible for the design and release of versions of BEA WebLogic Commerce Server and BEA WebLogic Personalization Server.

### CONTACT...

ddimov@bea.com

NEW RELEASE  
SIMPLIFIES COMPLEX  
PORTAL DEVELOPMENT,  
MAINTENANCE AND  
SECURITY

**W**ith the release of WebLogic Portal 4.0 in October 2001, BEA introduced a major update to its portal functionality and added

significant new features. In this article, we'll give you a technical overview of the new product, and provide a glimpse of some of the new functionality you'll find in it.

Although the portal market is still very fragmented and not well-defined, and the concept of a portal remains vague, we can say that a portal comprises a number of concrete and useful paradigms: the user interface paradigm, the content and data aggregation paradigm, the application development paradigm, and the enterprise architecture paradigm. On the user interface side, the model has been firmly established by consumer portals such as MyYahoo! present content and information to the user in a compact way on a page and let the user customize the look and feel of that page. The content and data aggregation model of portals is also familiar to many: present the users with a set of content and data sources, let them choose what they want to see, and present the information to them through a browser interface with small, window-like elements we'll call portlets. As an application development approach, the portal allows developers to introduce new content and functionality incrementally, as portlets, which can be added without disrupting the rest of the portal.

Finally, in enterprises with complex IT environments, with a multitude of both legacy and new applications, the portal becomes the unifying platform that must allow aggregation of and integra-





## Portal User Interface and Presentation Services

WebLogic Portal provides a user interface framework with prebuilt presentation

elements and templates, which simplifies the creation of personalized portal sites. The interface centers on portlets – compact windows arranged on portal pages that provide access to content and applications – as its primary model for presentation and content aggregation (see Figure 1). While our experience shows that it is impossible to provide a Web interface framework that will satisfy everyone's needs, we think we've been able to find a good middle ground for those who would like to stay within the portal interface paradigm. The user interface framework works out of the box, but exposes enough to the developers that they can customize it as necessary. The interface consists of the following key elements:

- **Portal Main Page:** The main portal template includes the portal header, portal footer, page navigation template, and the main portal content area. The main page is a JSP page that includes other JSP pages responsible for rendering their respective area of the portal.
- **Portal pages:** The main portal content area may contain one or many pages that are “stacked” on top of each other. A tabbed navigation bar is used to quickly switch between pages by bringing a page to the front. Each page consists of one or more layouts, and contains portlets within a layout.
- **Portlets:** Windows into applications and content that can be added to a portal page and presented as one or several rectangular windows.

tion with existing applications, as well as enable new application development. These paradigms apply regardless of the type of portal at hand, be it an internal employee portal, a consumer portal, or a partner and customer extranet portal.

BEA WebLogic Portal is a platform for creating a variety of portals, including employee, consumer, and partner portals. One way to look at it is as a J2EE application on top of WebLogic Server. On the other hand, one may look at it as a natural extension to WebLogic Server. In any case, WebLogic Portal will prove interesting to any developer creating complex, application server-based systems with portal-like characteristics.

- **Portal WebFlow:** A mechanism for specifying the user interaction and navigation flow in the portal declaratively, as a set of events and actions, defined via a graphical WebFlow editor tool
- **Page layouts:** “Wireframes” that define how portlets will be arranged on a portal page.
- **Skins:** These determine the look and feel of the portal and are defined using Cascading Style Sheets.

### PAGE LAYOUTS AND PORTLET PLACEHOLDERS

Each page in a portal page set can use one or more layouts to determine how portlets are arranged on the page. A layout consists of a number of portlet placeholders. Placeholders are rectangular areas that can accommodate one or more portlets, arranged vertically. Portlet placeholders can be arranged in a layout in any way – as a set of columns, as columns and rows, as rows spanning some columns, and so on.

Under the covers, a layout template is implemented as an HTML table definition. Each table cell includes a JSP tag. At the time of page rendering, the tag makes calls to the portal engine to determine which portlets should be included within the cell.

Because the layout definition is clean and concise, an HTML developer, even if unfamiliar with JSP, can easily create a new layout by drawing up an HTML table in a graphical HTML editor, and then cutting and pasting the JSP tag into table cells. Attributes that determine the width of the cells, the thickness of borders between cells, whether the cells have absolute or relative dimensions, and others, are determined by standard HTML table cell attributes and can easily be specified by an HTML developer.

Additionally, not all of the cells in the layout table need to include portlets. A site designer may choose to have some areas that are not portlet-based and include static HTML content or a JSP page, making the layout mechanism flexible and customizable.

### ANATOMY OF PORTAL RENDERING

Most of the portal rendering is in the JSP layer. The main JSP template, `portal.jsp`, communicates with the portal engine to set up the rendering environment, including determining what skin to use, and delegates the rendering to sub-templates by including additional JSP templates. The sub-templates in turn include templates for specific

FIGURE 1



An example of a portal with multiple pages and a variety of portlets.

areas of the portal, such as the header, the page navigation bar, the page layout area, and the footer.

The JSP template for the page layout area is determined dynamically by consulting with the portal engine, according to group and user layout personalization settings, as there may be several layouts available for a given page. As mentioned above, the layout template contains an HTML table with a JSP tag in its cells – the portlet placeholder. At rendering time, the tag consults with the portal engine on which portlets to include, according to portlet entitlements and group and user portlet personalization. When a portlet is included in a layout, all of its JSP elements are included in the page, as well as the appropriate portlet view. At that time, the business logic implemented in the portlet view (or referenced from it) executes. Figure 2 illustrates the page aggregation and rendering process.

**ANATOMY OF A PORTLET**

A portlet mimics a window control of a Windows-based user interface system, but within a portal page in a browser. A portlet window can be added, removed, minimized, maximized, and detached from the portal page. It is assembled from several JSP fragments: the portlet titlebar, the banner, the header, the content area, and the footer. The most important element of a portlet is its content area, provided by a portlet developer. The portlet content is a JSP page and may include simple HTML or

dynamic data resulting from the execution of JSP tags or scriptlets, which call back-end components. The portlet content JSP can access any functionality that is available within the portal and is exposed via JSP tag libraries or Java APIs.

Because portlets in WebLogic Portal can leverage WebFlow (see section below), the content area isn't limited to a single page, and may in fact be comprised of a set of pages, if required by the portlet interaction model.

A portlet is a convenient component development paradigm. Once it is created, adding it to the portal interface and managing it is simple and done using administration tools. We see the portlet paradigm developing into a new component model, a sort of mini-application that includes both business logic and the user interface. Web services will figure prominently in this new model.

**INFRASTRUCTURE SERVICES AND FEATURES**

Of course, a portal isn't just a pretty façade – it needs solid infrastructure; features that cut down development time; and a scalable, standards-based architecture. WebLogic Portal uses the infrastructure provided by the underlying WebLogic Server, and adds features critical to creating full-featured, personalized portals.

**Built on J2EE and WebLogic Server**

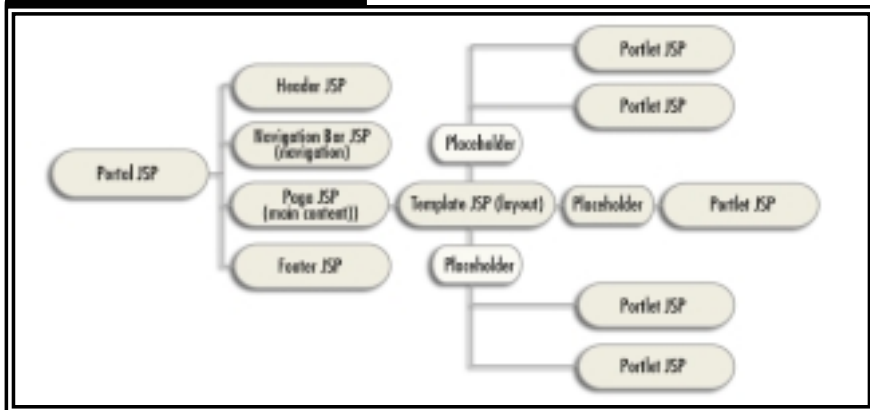
WebLogic Portal is implemented using the J2EE architecture, and is in fact a J2EE application that runs in the WebLogic Server environment (see Figure 3). In J2EE terms, WebLogic Portal is an Enterprise Application. It consists of a collection of Enterprise Java Bean (EJB) components and a set of Web applications, which are collections of servlets, Java Server Pages (JSPs), JSP tag libraries, and supporting Java classes. Both the portal functionality itself and the portal management tools are part of the J2EE Enterprise Application.

Because WebLogic Portal is a J2EE WebLogic Server application, it leverages the infrastructure provided, such as security, JDBC connection pooling, caching, clustering for failover and load balancing, J2EE application deployment, Web services support, system-level administration and management. For example, the WebLogic Portal Enterprise Application can be deployed over a set of clustered servers. This is in stark contrast to other Java-based portal implementations, which are typically confined to the servlet engine and the Web container and take little advantage of a J2EE application server.

Moreover, because the WebLogic Server provides a complete J2EE application environment, new application components can be developed and added to the WebLogic Portal application. These components can range in complexity from JSP pages to Java classes to EJB components.

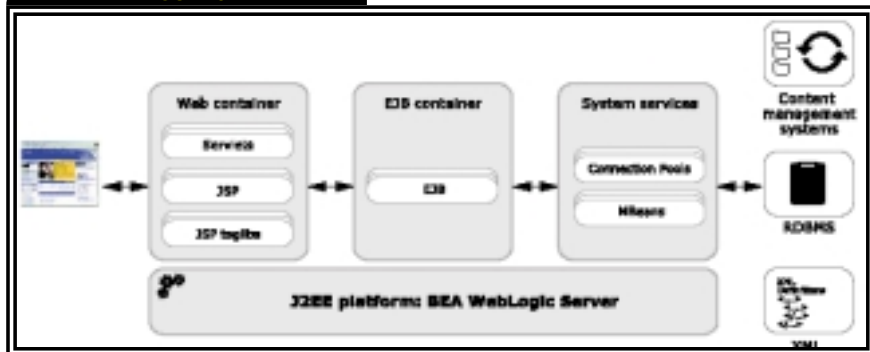
You can see the J2EE structure of WebLogic Portal in the standard WebLogic Server administration

**FIGURE 2**

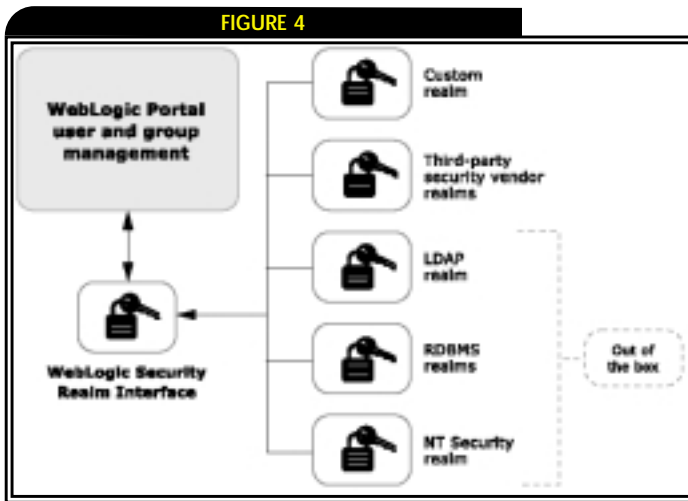


Simplified diagram of portal page aggregation

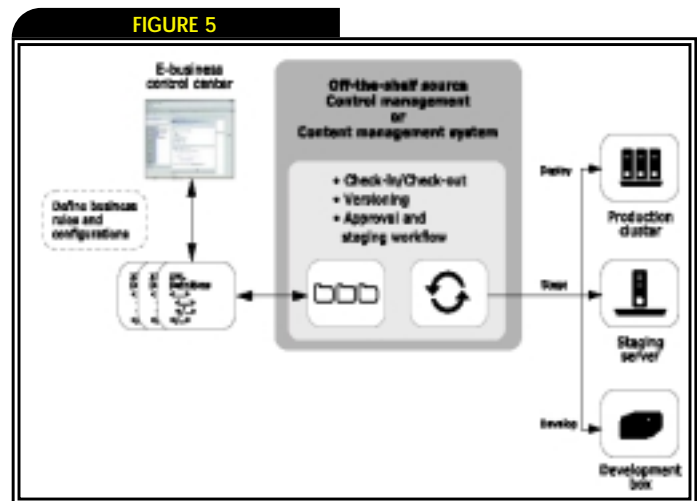
**FIGURE 3**



WebLogic Portal J2EE architecture



WebLogic Portal uses WebLogic Server security realm for users and groups



Files can be placed under version control and easily moved between development and testing environments.

console, where you can browse through the portal Enterprise Application and look at its constituent Web applications, EJBs, JSP tag libraries, and so on. You'll find the PortalManager Stateless Session Bean, which is the central component of the portal engine, and responsible for retrieving and persisting information about what users will actually see when they log into the portal—portlets, pages, layouts, skins, and so on.

#### Security and User and Group Management

The WebLogic Portal main authentication, authorization, and personalization model is built upon the concept of user groups. User access to a portal, and that user's primary view of the portal, is determined by the user's group membership. WebLogic Portal provides infrastructure and tools to define the structure of groups and the membership of users within groups. This design leverages the J2EE role and principal concepts provided by the underlying WebLogic Server.

WebLogic Portal uses the security realm mechanism provided by WebLogic Server for user and group information. It offers a plug-in security interface, which has been used by leading third-party security vendors to enable access to their user repositories. This enables WebLogic Portal to use such third-party systems for user and group information.

#### Application Deployment Model

Many products that provide portal, personalization, or commerce functionality are database-driven. The application configuration, business rules, attributes, and parameters are stored in relational database tables, along with the actual application transactional information.

Administration and configuration tools typically operate on the server instance and modify database records. One advantage of this approach is the fact that changes in the application behavior can be effected immediately; however, what if the changes require review and approval? How are

those changes moved between development, staging, and production instances? How can the changes be versioned and rolled back if necessary? How can concurrent development be performed? Systems that are completely database-driven attempt to address these issues by either implementing proprietary version control systems inside the database, or by resorting to database scripts to move information between staging databases.

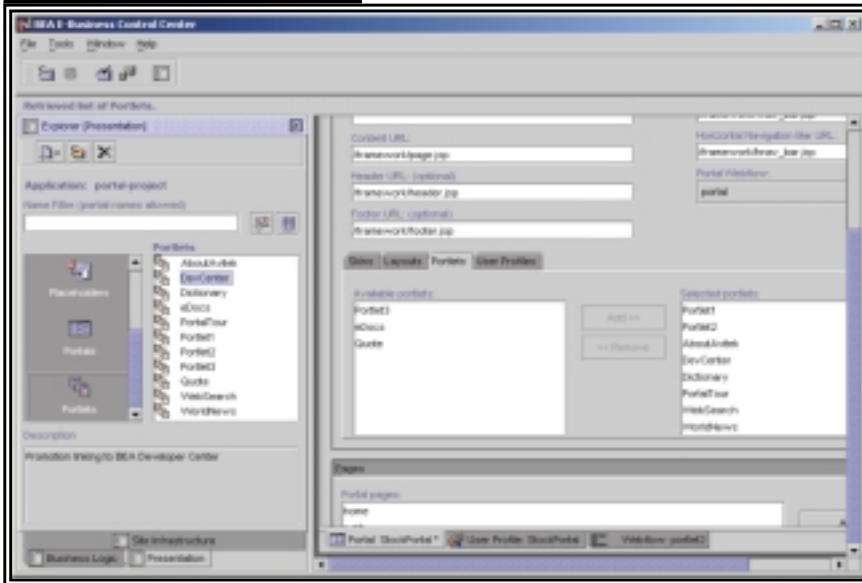
WebLogic Portal introduces an application deployment model that uses XML files. With this model, the application business configuration data, such as business rule definitions, user profile definitions, navigational and process WebFlows, along with portal and portlet definitions, is expressed in XML format and stored in the file system. Of course, frequently updated transactional application data, such as customer information, order information, and user profile attribute values, remains in a relational database.

With this model, on the development side, the definitions created in the WebLogic Portal tool suite, E-Business Control Center, result in XML files created on the local file system. On the server side, a special set of deployment services takes the XML files and deploys them to the server. A source-control management system or a content management system can be introduced between the development and the deployment steps, to implement versioning, check-in and check-out, and approval workflow, as well as to restrict access to specific deployment files (see Figure 5).

#### Portal Tools

You will find two sets of tools in WebLogic Portal: a Java client-based tool suite, the E-Business Control Center (EBCC), and the browser-based set of Portal Administration tools. The EBCC provides rich graphical interfaces that simplify complex tasks such as rule definition, WebFlow editing, and portal creation and management. EBCC plays within the

FIGURE 6



EBCC Portal module

Application Deployment architecture described earlier: as users work with its point-and-click interface, it generates XML files that are synchronized with the server. The browser-based Portal Administration tools focus primarily on administering and managing the portal at runtime. Figure 6 shows the EBCC Portal module.

## DELEGATED ADMINISTRATION

Browser-based Portal Administration tools support delegated administration. Delegated administration, sometimes called decentralized administration, is a necessity in today's portal environments. It's common to delegate authority not only to members of IT personnel, but to other users to enable them to perform tasks related to their duties. Delegation takes the load off the central administrator, without granting sweeping administrative authority to every user.

WebLogic Portal provides the infrastructure and tools for setting up the delegated administration structure and specifying the specific tasks that are delegated and the scope within which the delegated administrator will operate. For example, it's possible to set up a delegated administrator who is authorized to perform certain portal management tasks (such as managing users, managing look and feel, or managing portlet permissions) only for a certain group of users, or for several groups, or for all groups in particular portal. Figure 7 shows the Delegated Administration tool.

## Rule-Based Entitlements

WebLogic Portal provides group-based access control as a default way for controlling access to a portal. Group-based access control is typical of many portal products; it's most useful when the user

# Performant

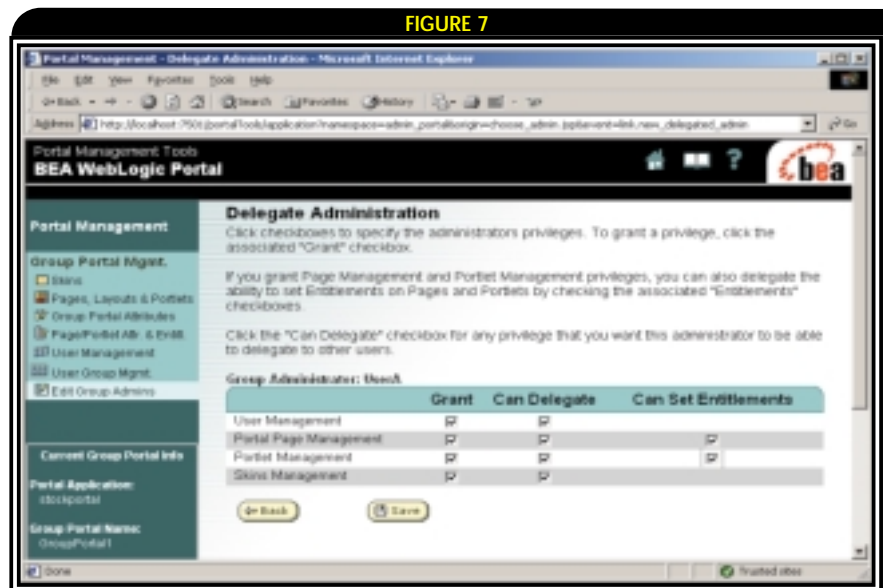
[www.performant.com](http://www.performant.com)



group structure changes infrequently, and the access to portlets and pages maps well to user groups. However, group-based access control is not sufficient if access to portlets or pages is dependent on a more complex set of conditions than simply the group membership, or if it may need to change dynamically based on changes to the user profile, without the involvement of an administrator.

With traditional group-based or role-based access control, the placement of a user in a group or a role rests with an administrator. Where a more dynamic, flexible entitlement behavior is needed, rule-based entitlements become helpful. With rule-based entitlements, the business rule behind an entitlement becomes a self-documenting business policy that governs whether a user is given access to portal content through portlets or portal pages.

An example of a rule-based entitlement is to imagine an investment portal with a portal page that contains portlets and information related to option trading. The portal page has an entitlement associated with it that specifies that the page is visible only to users that fall under the "OptionTradingEligible" entitlement segment. The "OptionTradingEligible" entitlement segment is defined by a dynamic rule that executes against the profile of the user and verifies that "OptionApplicationFiled" attribute is set to "true" and that the "AccountBalance" attribute has a value over \$25,000. The "OptionTradingEligible" enti-



Delegated Administration tool

ment segment thus represents a business policy that this portal chose to use to govern access to one of its pages.

Rule-based entitlements govern not only whether a given portlet or page is accessible to a given set of users, but also whether a given portlet is mandatory, or whether the users can edit it.

# Corda Technologies

[www.corda.com](http://www.corda.com)



### WebFlow

WebFlow is designed to help you build Web applications that maintain the much-desired separation between presentation logic and underlying business processes. Because WebFlow's centralized XML configuration files specify the order in which pages are displayed to your Web site's visitors, use of the WebFlow mechanism may reduce the work necessary to create and modify the flow of your Web site. At appropriate times during a visitor's interaction, the WebFlow may also invoke predefined, specialized components to validate data or to execute back-end business processes. Therefore, using it

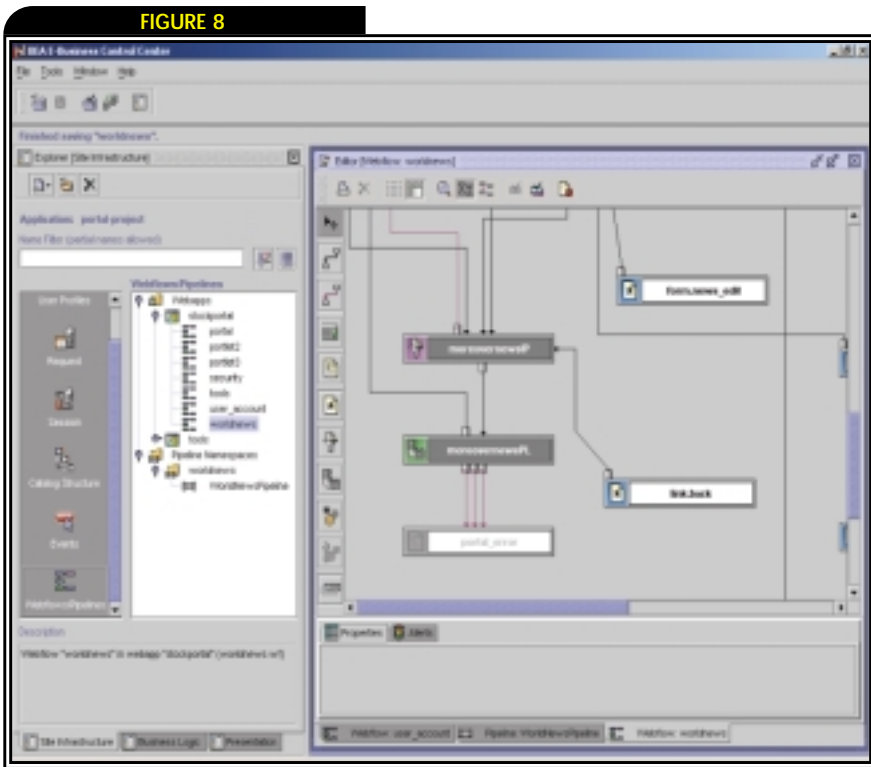
may also make it faster and easier for your development team to complete modifications that require back-end programming (see Figure 8).

#### WEBFLOW AND PORTLETS

Portlets in a portal may range in complexity from simple read-only views of content to complex "mini-applications" with sophisticated interaction flows. If a portlet requires nontrivial interaction with the user, it will benefit from using the WebFlow mechanism. WebLogic Portal allows each of the portlets in a portal to have its own WebFlow. For example, an HR benefits sign-up portlet may consist of a number of screens that are different forms a user needs to step through and fill out. The interaction flow with these screens may be controlled by WebFlow.

In addition, several portlets may interact with one another via the WebFlow. Suppose that a suite of portlets provides customer and order information through a portal. Entering a customer number in one portlet results in a display of the status of the latest orders in this portlet, and details of the customer account in another. This is achieved by having portlets examine events posted by other portlets in the course of WebFlow execution. The inter-portlet communication enabled by WebFlow is invaluable in portals that go beyond simply exposing the view into content and aim to present an interactive window into applications.

In conversations with customers, developers, and analysts, I've found that people find WebFlow is another step toward doing away with the grunge of coding complex navigation and validation into Web applications. It makes simple things easy, but it also makes more advanced processing possible—if you need to introduce complex data validation code between transitions, just add an Input Processor, and write your validation logic in Java. If you need to invoke back-end functionality, such as EJB components,



WebFlow Editor tool.

**RECEIVE \$150**  
DISCOUNT OFF FULL CONFERENCE  
REGISTRATION  
WEB SERVICES EDGE



2002WORLDTOUR

**Learn How to Develop  
SOAP Web Services NOW!**  
at a One-Day Tutorial... Coming to a City Near You!

add a Pipeline Component, and write your client code there.

### Personalization

In addition to the end-user and group personalization of the portal content and interface, typical for many portal frameworks, WebLogic Portal supports fine-grained, rule-based personalization.

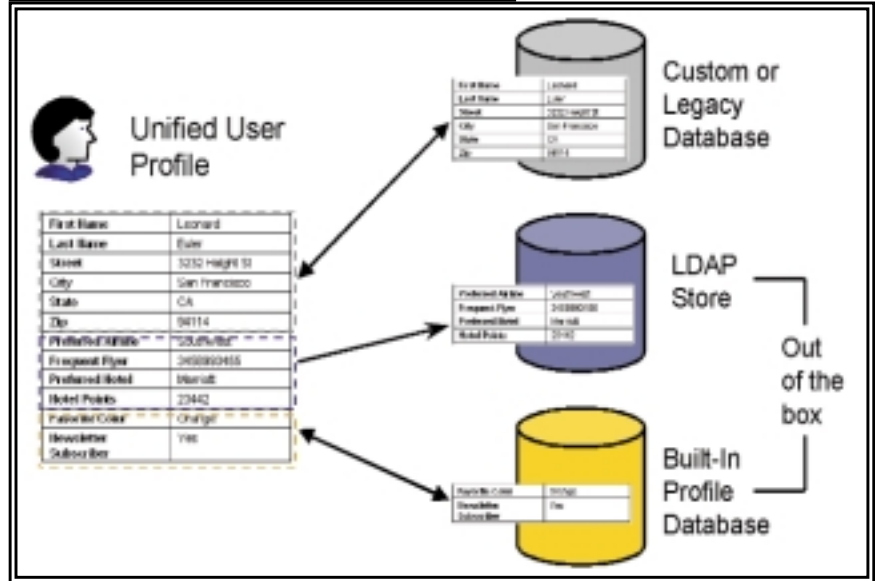
Two key pieces of personalization are the user profile and the rules engine. Based on the values of attributes in the user profile, business rules can be defined and used to personalize the content presented to the user and the interaction with the user. The business rules may also be based on other parameters, such as session attributes, request attributes, and time. For example, a rule may be defined to place users in a segment when their Zip Code attribute is 94104 and the time is between 9 a.m. and 5 p.m.

A business rule may also be used to directly relate a user segment to content metadata, creating a content selector. When a content selector is executed, it returns a collection of content with matching metadata. For example, a content selector may specify that only financial content be returned whose metadata matches the investment preference attribute values in the user profile. The content personalization functionality ties closely with the Content Management interface (described later) provided by WebLogic Portal.

The JSP tags that provide access to rule-based content personalization can be used anywhere within the portal user interface: inside a portlet, within the portal header and footer, or within the page layout template. This allows for fine-grained personalization.

WebLogic Portal provides a tool for defining personalization rules, very similar to the rule-based entitlement tool described earlier. Both features rely on the same base technology, the BEA rules engine.

FIGURE 9



Unified User Profile

### Unified User Profile

While some attributes in a user profile will be newly created as part of the portal application, our customers almost always have to incorporate user information from existing data stores and applications. The Unified User Profile (UUP; see Figure 9) provides the architecture for presenting a single view of the profile across disparate data sources.

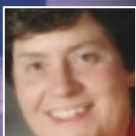
The UUP is implemented as a Stateless Session EJB, which acts as an aggregation interface to multiple stateless session EJBs responsible for retrieving their portion of the user data. On the portal side, the attributes in the Unified User Profile can be accessed via an administration tool, as well as programmatically, via an API or a JSP tag library.

To map user attributes from an existing system to the UUP, a developer implements an interface that

# Jump-start your Web Services knowledge Get ready for Web Services Edge East and West!

AIMED AT THE JAVA DEVELOPER COMMUNITY AND DESIGNED TO EQUIP ATTENDEES WITH ALL THE TOOLS AND INFORMATION TO BEGIN IMMEDIATELY CREATING, DEPLOYING, AND USING WEB SERVICES.

EXPERT PRACTITIONERS TAKING AN APPLIED APPROACH WILL PRESENT TOPICS INCLUDING BASE TECHNOLOGIES SUCH AS SOAP, WSDL, UDDI, AND XML, AND MORE ADVANCED ISSUES SUCH AS SECURITY, EXPOSING LEGACY SYSTEMS, AND REMOTE REFERENCES.



#### PRESENTERS...

**Anne Thomas Manes, Systinet CTO**, is a widely recognized industry expert who has published extensively on Web Services and service-based computing. She is a participant on standards development efforts at JCP, W3C, and UDDI, and was recently listed among the Power 100 IT Leaders by Enterprise Systems, which praised her "uncanny ability to apply technology to create new solutions."



**Zdenek Svoboda is a Lead Architect** for Systinet's WASP Web Services platform and has worked for various companies designing and developing Java and XML-based products.

EXCLUSIVELY SPONSORED BY



**systinet**

WASHINGTON, DC (Tysons Corner Marriott) .....**FEBRUARY 26**  
 NEW YORK, NY (Doubletree Guest Suites) .....**MARCH 19**  
 SAN FRANCISCO, CA (Marriott San Francisco) .....**APRIL 22**

REGISTER WITH A COLLEAGUE AND SAVE 15% OFF THE \$495 REGISTRATION FEE.

Register at [www.sys-con.com](http://www.sys-con.com) or Call 201 802-3069



accesses the source system and retrieves specific properties, and registers this implementation with WebLogic Portal. The UUP provides a high degree of flexibility in mapping attributes from multiple sources. For example, the total set of attributes in a user profile may comprise a subset of attributes stored in LDAP, a subset of attributes stored in a default RDBMS profile store, and a subset that comes from a custom database.

As the Java 2 CA takes hold, more and more legacy systems will be exposed via standard adapters, which means less custom coding will be needed to access the existing profile information. UUP then will evolve into an aggregate representation of these adapters that creates a single view of the profile.

### Interoperability

BEA WebLogic Integration (WLI) is a suite of products for application integration, business process management, and B2B integration functionality for the enterprise. Both WLI and WebLogic Portal leverage the same WebLogic application server platform. Combining WLI and WebLogic Portal allows enterprises to roll out solutions that address end-to-end portal requirements, from the personalized portal front-end to back-end application integration. WebLogic Portal and WLI can work together to enable fully integrated enterprise applications. WebLogic Portal's WebFlow, UUP, and business services can work with WLI's business-to-business integration (B2Bi), business process management (BPM), and application integration (AI) features to facilitate process-level communication and data flow between Web applications and other enterprise systems.

ation of portlets that allow end-users to interact with Web services (see Figure 10). The tool can introspect a WSDL file and automatically generate JSP code for a portlet to allow the end user to interact with the Web service.

For Web service creation, the foundation of WebLogic Portal – the WebLogic application server – includes comprehensive Web service support, including the ability to automatically expose functionality that runs within the application server as Web services. Thus, developers creating new functionality as part of a portal application may expose that functionality not just via the portlet user interface, but also as a Web service to be accessed by other applications.

### Additional Features

In this article, I've reviewed some of the features of WebLogic Portal but haven't done justice to many others, including built-in commerce capabilities, campaign and interaction management, content management, and event and behavior tracking. However, if you evaluate WebLogic Portal, you'll find that it's equipped with a fully-featured commerce server that will be helpful in creating a portal that needs commerce functionality. You'll also find an impressive point-and-click tool for designing campaign scenarios that govern how end-users will interact with the site. The event and behavior tracking will allow you to track any important interactions that occur within the portal. Finally, the content management interface enables integration with many off-the-shelf content management systems, but also includes a reference implementation content repository that will help you get started with a prototype.

### Summary

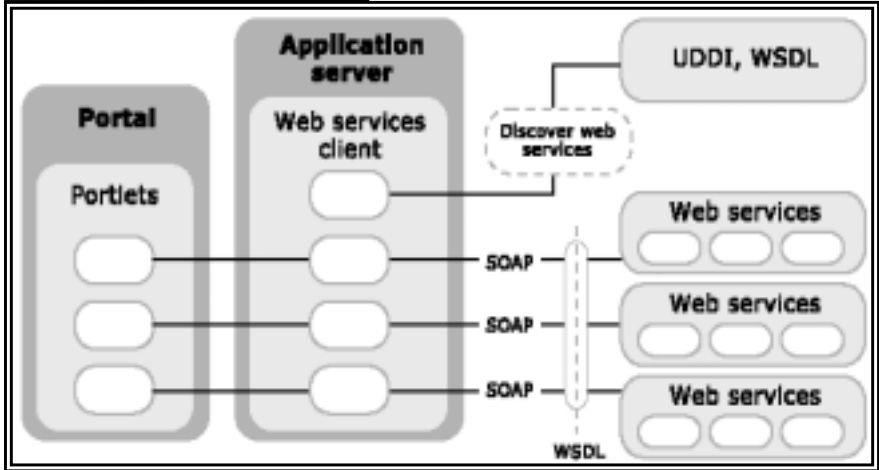
BEA WebLogic Portal provides a high level of base portal functionality to simplify complex portal development, maintenance, and security. It simplifies the rapid creation, customization, and administration of enterprise, partner, and customer portals.

Developers will find a breadth of functionality that will save time in developing a portal solution. At the same time, the product exposes many extensibility points where the developers can customize the default behavior.

You will also find that WebLogic Portal is a true implementation of the J2EE component model in the WebLogic environment, and represents best practices in developing large applications in J2EE environments. It was built from the ground up on WebLogic Server, and our engineers had no choice but to fully utilize all that WebLogic Server has to offer. After all, portals are complex applications, so they should be exploiting the application server to the fullest.

Download a copy of WebLogic Portal from [www.bea.com](http://www.bea.com).

FIGURE 10



Web services aggregation in WebLogic Portal

### Web Services Integration

With WebLogic Portal, developers can create Web services for consumption by other applications, as well as aggregate existing Web services into the portal. For Web service aggregation, WebLogic Portal provides Web services SOAP-client functionality and a tool for automatic cre-

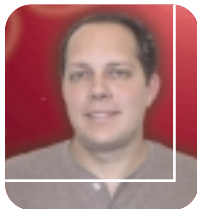
# Divine

[www.divine.com](http://www.divine.com)



# Holistic Infrastructure Monitoring and Management

MAKING INFRASTRUCTURE MANAGEMENT A TRUE BUSINESS ENABLER



BY

GREG PETERS & LANCE PETERSON

## AUTHOR BIOS

Greg Peters, VP of Engineering at NOCpulse, has over 10 years of experience in software development, system engineering, and program management. He leads the company's core product development strategy and engineering teams.

Lance Peterson, program manager at NOCpulse, is responsible for interface design and engineering program management. Previously, he worked as a programmer analyst at Emory University and in an earlier life taught literature, film, and linguistics.

## CONTACT...

gpeters@nocpulse.com  
lpeterson@nocpulse.com

**W**ebLogic Server, like most applications, provides robust and detailed monitoring tools bundled with the basic application. The embedded monitoring and management provided by the WebLogic Console is extremely useful when diagnosing and repairing a problem once it has been isolated in the WebLogic Server. But this embedded point solution is of limited use in most real-world situations where the application server is just a single component in a system of components that are all vital to providing the end-user application. When trying to quickly diagnose a general problem with the end-user application, it is much more powerful and effective to view data from each component holistically as a part of the system rather than evaluating each component by itself.

This holistic view becomes even more powerful when correlated with end-user application and business transaction performance. If business priorities and processes can be included in this view of the infrastructure, operations management becomes more valuable to the enterprise.

## Point Solutions

A typical Web infrastructure is composed of a multitude of hardware and software components: hosts running various operating systems, Web servers, application servers, databases, legacy systems, and network devices. Each element of the infrastructure usually includes a point monitoring and management solution.

The command line utilities available to operating systems in the Unix family are typical point solutions. Point solutions provide power and flexibility to the experienced administrator. At the application level, vendors package tools for the monitoring and management of their applications. The WebLogic Server Console provides detailed information about many aspects of WebLogic (current connections, JVM memory usage, database connection pool load, etc.).

The common benefit of these point solutions is that they enable real-time system monitoring, troubleshooting, and resolution by the experienced administrator. Moreover, these tools are ultimately used to solve problems once they have been identified.

One difficulty with point solutions, however, lies in their heterogeneity. It is not always self-evident to an experienced Unix administrator how to identify an errant process on a Windows platform, although the operation is nearly identical to that used on Unix. Similarly, an administrator familiar with WebLogic Server Console might have difficulty using a Web server or another application server's point solution. Point monitoring solutions demand a high level of specialization and domain expertise from administrators.

A second difficulty is in their dispersion. With point solutions there is no visibility across the components, which can be of various types. Unfortunately, an operational problem with WebLogic Server may be related to an underlying problem with the host, the database server it's connected to, or the network layer. Because specialists are hired to fix particular problems, each using their own tools, problem isolation is slow and labor intensive, especially in large and highly specialized IT organizations.

A third drawback to point solutions is that they tend to be reactive. They're used when the system is already down or performing poorly and the administrator is responding in real time to the problem as the business suffers the consequences. Lacking an aggregate tool, inventive teams have crafted scripts and other tools to facilitate these activities, but the maintenance of these tools requires even more expertise.

## Infrastructure as a Whole

Infrastructure managers quickly realized that the information they were getting from their various point solutions was much more valuable when viewed together as part of a larger whole. They realized that fault isolation and correlation became much easier once they had this larger view. In the network world, SNMP became the de facto standard for aggregating all of the element data and enabled master management consoles.

Suddenly, infrastructure managers could do more to support the enterprise because they were spending less time in the isolation phase of problem resolution. Infrastructure management was still princi-



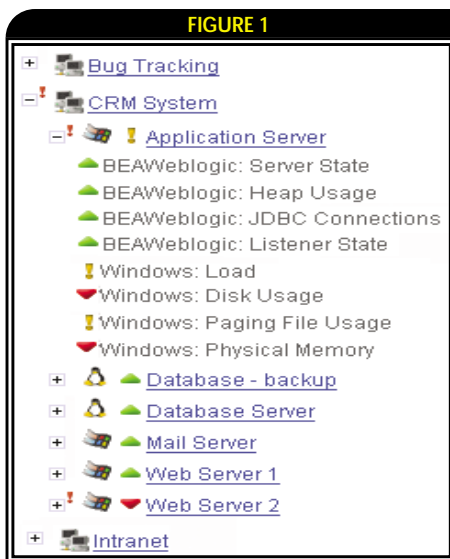


pally reactive, but it was faster. For the most part, though, these goals were only achieved at the network layer. State models of IP-based networks are much easier to understand and manipulate than what is happening at the application level (see Figure 1).

As network management solutions attempted to integrate infrastructure management and monitoring above OSI Layer 3, the problem of resolving a common view of the health of the infrastructure from disparate data sources becomes even more pronounced. Application vendors extended SNMP support to their products (WebLogic, for example, can be monitored via SNMP) as an attempt to enable a common infrastructure view.

But SNMP can suffer from reliability and complexity issues, as well as being incompletely supported by some components in the infrastructure. Access to command-line utilities and Windows tools like Performance Monitor also remain critical in the complete management of infrastructure resources. Also, databases typically make most statistics available only via SQL.

Agent-based systems are often proposed as an alternative comprehensive monitoring technique, but are difficult to manage and tax system resources unnecessarily. They also often lack access to some of the components you might need.



A view into the layers of an n-tier architecture via the NOCPulse Command Center.

Common access to system information from heterogeneous sources is vital. Most existing approaches to consolidation involve a Manager of Managers (MoM) system, which receives monitoring data from multiple sources. These systems tend to be prohibitive-

ly expensive to purchase, customize, and support. They also take considerable time to implement and require significant training to be used effectively.

NOCpulse Command Center uses a plug-in framework to separate monitoring data from the access methods used to gather that data, providing the benefits of the agent-based and MoM systems without suffering the disadvantages. Lacking a cross-industry standard to reconcile the very different issues encountered when monitoring an application like WebLogic versus an operating system (and such a standard seems unlikely, even considering the distant future), a flexible, extendable product-based standard (like NOCPulse Command Center's plug-in framework) is the next best thing.

Also key is a common data repository and interface. Command Center plug-ins access required metrics via multiple protocols, but the results are presented in a common format via a single Web-based user interface. Performance metrics collected from the infrastructure are gathered in a common data store, allowing easy data mining, historical event correlation, and root cause analysis through a shared report engine (see Figure 2).

The result is a holistic view of all the components that make up an end-user application: up the stack from the operating system through the application layer to the network and vertically from server to server.

### Infrastructure and the End User

Too often the focus of monitoring is attention to system problems without regard to the real end-user impact. What is ultimately important is not the specific health of all of the individual components of an Internet infrastructure, but the performance of the application at the other end. Can our customers currently purchase a CD from our site? Is our billing system too slow? Does the customer service section of our site offer the help our customers need? The era of Web applications has given rise to point solutions for end-user monitoring. These products either quantify end-user experience or monitor site accessibility.

Unfortunately, the limited end-user monitoring approach ignores the infrastructure. Web site slow? Administrators go to another solution to solve the problem. Web infrastructures grow quickly in complexity; an administrator might not be able to expediently and effectively correlate end-user performance issues with a particular component of their infrastructure.

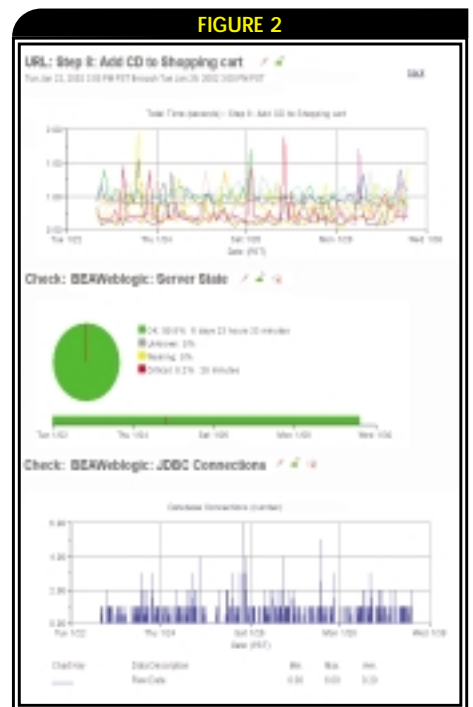
Holistic monitoring requires a common

interface to both infrastructure health and end-user application performance. NOCPulse Command Center provides this common interface and allows a user to model a multi-step browser-based transaction through a point-and-click configuration tool for both remote and local monitoring.

It may be of interest to see the performance of my e-commerce site at a sample of locations on the public network. But this needs to be triangulated against local performance. Customers were unable to place orders for a time. Do I need to complain to my network provider or do I need to scale up my Web server farm? Point end-user monitoring solutions cannot answer this question; a holistic approach that correlates user experience to infrastructure health can.

### Infrastructure and the Enterprise

Once we have a single view of the infrastructure that can be correlated to the performance of our end-user applications and business transactions, resources can be efficiently applied to quickly solve problems. The infrastructure can be tuned to provide better performance with a strong feedback loop of the metrics that matter, ensuring that changes actually have the intended effect. Service Level Agreements can be managed proactively; problems become defined by what the end user is experiencing or by metrics associated



Command Center 7-day report correlating end-user timing of a single Web-application transaction.

with the business transaction rather than by an arbitrary definition of "problem" at the infrastructure level.


The final stage in the development of infrastructure management involves connecting business management to the infrastructure. It involves making the priorities and concerns of the business transparent within the view of the infrastructure. Now precious operations resources are not only able to resolve problems quickly, but they can be applied efficiently to where they matter most: to the most urgent problem, where urgency is determined by business priorities. Fundamentally, it amounts to quickly getting the right people, with the right tools and information, to the most important problem (as defined by the priorities of the business).

For example, we might have problems with two applications: our CRM system is down and our credit card approval process is running slowly. From an operational perspective, the first problem may seem more severe, but when tied to business priorities, the risk of lost revenue mandates that effort be applied to the second problem first. A truly holistic management solution enables these types of decisions automatically.

NOCpulse Command Center allows users to build arbitrary groups of components that corre-

spond to business processes, transactions, customers, or end-user applications. The behaviors of each of these groups (thresholds, notification destinations, escalation procedures, etc.) can be set in accordance with the importance of each group to the business. Critical customer issues get raised and resolved while lower priority or tolerable problems wait until people are free to deal with them.

### Conclusion

Fundamentally, infrastructure management adds more and more value to the enterprise as it evolves from an inefficient, slow, reactive, element-focused approach to an efficient, responsive, proactive approach that is able to see infrastructure holistically and understand the relative importance of each end-user application to the business. When that level is achieved, infrastructure management becomes a true business enabler, allowing service level management, efficient customer problem reporting and resolution, and prioritization of fault response in accordance with business priorities. These benefits require a holistic operational view that provides the ability to correlate what is happening at the infrastructure level with what is happening at the business level. 

EnginData Presents

# 2002 Developer Market Survey Reports



Our comprehensive reports offer insight and strategy to guide your most critical business decisions in today's fastest growing technologies...

- ✓ Establish your product and marketing strategy
- ✓ Understand your customer's needs
- ✓ Evaluate Technology & Trends

Preview and order reports at [www.engindata.com](http://www.engindata.com)

[engindata.com](http://engindata.com)

**engindata** ✓  
RESEARCH

# WebGain

[www.webgain.com](http://www.webgain.com)



## The Subtlety and Power of Entity EJBs

### WHY YOU SHOULD USE EJBs ON YOUR NEXT IMPLEMENTATION

BY TYLER JEWEL

The first in a series of columns focused on architecture with the BEA WebLogic Platform.



#### AUTHOR BIO...

Tyler Jewell is BEA's director of technology evangelism and the coauthor of *Mastering Enterprise JavaBeans 2.0* and *Professional Java Server Programming (J2EE 1.3)*. He writes widely on the Web on the subject of both J2EE and Web Services.

#### CONTACT...

tyler@bea.com

For software developers, architecture is about making choices and tradeoffs. Solid architectures are the result of careful analysis of the problem and its solutions. There's no such thing as a perfect design for a distributed system..

All distributed systems adhere to the three "RAS" constraints (Reliability, Availability, and Serviceability): anything done to increase the availability of a system subsequently reduces its reliability and serviceability; anything done to increase the reliability of a system decreases its availability and serviceability; and anything done to increase the serviceability of a system reduces its availability and reliability.

This series will provide insights into various microcosms of the WebLogic system. None of them individually makes or breaks an individual implementation, but together they equate to a more robust implementation all around. The columns will always focus on the tradeoffs being made and how they impact the relative weights of the RAS characteristics for a system.

This column focuses on the power of entity EJBs and why you should consider using them on your next implementation.

#### Unlocking the True Power of Entity EJBs

There has been considerable debate in the development community about how flawed the entity EJB model is and how its performance isn't on a par

with that of a stateless session EJB (SLSB) with JDBC. The conversations have centered around performance, the meaning of distributed component architecture, the relevance of JDO, the cumbersome nature of EJBs, and so on...and on. These discussions were insightful, but they never focused on the primary reason that entity EJBs can be a powerful implementation tool. This article provides a discussion of the write once, deploy n-times model and how it might be incorporated into your next project.

(This article uses the EJB 2.0 specification for its arguments, as referring only to the EJB 1.1 entity EJB model would invalidate some of the points.)

#### Write Once, Deploy n Times

EJBs have transactional, security, and interoperability support at the container level. It's been repeatedly demonstrated, however, that these capabilities are more relevant at the SLSB facade layer that accesses a data layer. Indeed, most architectures that employ EJBs have the SLSB facade handle these aspects. They configure their data access layer to use entity EJBs as a pass-through for these services (i.e., if an SLSB method is configured to start a new transaction, the entity EJB method invoked by the SLSB method will be configured to accept the already started transaction).

Although important, these EJB features distract people from appreciating the subtle and powerful nature of entity EJBs – especially their ability to be tailored to every conceivable data scenario using a "write once and deploy n-times" model.

Every system has data that's used in different ways. There's data that's read-only nontransactional, read-write transactional, read-write transactionally clustered cache, read for update over multiple requests, modified through batch updates, and used in bulk data retrieval. The number of ways that data is accessed/used is limitless. The complexity of these operations increases when relationships are factored in. If you disagree, think about how difficult it would be to implement a transactionally aware clustered cache that would perform eager relationship caching for entity EJBs with relationships three levels deep. One of the goals of the WebLogic Server entity EJB container implementation is to provide the services required to support these varying data usage scenarios.

It would be foolish to think that an entity EJB deployed in its basic configuration is best suited to handling data for each of these scenarios. The out-of-the-box configuration for entity EJB engines, such as WebLogic, is designed to handle read-write trans-

# Web Services Journal

[www.wsj2.com](http://www.wsj2.com)





SUBSCRIBE AND SAVE

**XML JOURNAL**

Offer subject to change without notice

ANNUAL NEWSSTAND RATE	
<del>\$83.88</del>	
YOU PAY	
<b>\$77.99</b>	
YOU SAVE	
<b>\$5.89</b>	Off the Newsstand Rate

**DON'T MISS AN ISSUE!**

Receive 12 issues of *XML-Journal* for only \$77.99! That's a savings of \$5.89 off the annual newsstand rate.

Sign up online at [www.sys-con.com](http://www.sys-con.com) or call 1 800 513-7111 and subscribe today!

**In March XML:**

**XML Glue**

Using an XML Workflow and integration layer for Telecommunication Providers

**XML messaging**

Use JAXM to exchange SOAP messages (Part 1 of 2)

**A client for testing server side XML applications**

How to speed up the process of building and debugging server based XML applications.

**ANT: An Introduction by Example**

A platform independent alternative to shell script based build tools

**XML Is a Treasure My Heart Cannot Deny**

The magic of an XML stylesheet



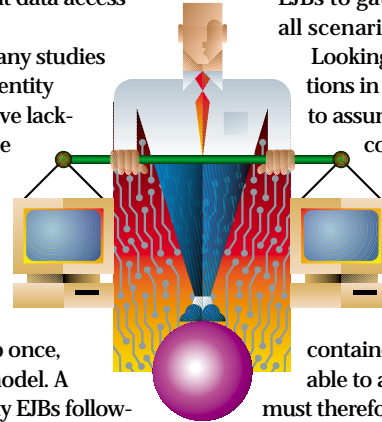
actional data with the best possible performance. This is done because read-write data is the easiest way to represent life cycle data: data that's created, used for a time, and later destroyed. It allows EJB developers to quickly test the range of functional operations of the data. This functional robustness is often interpreted as the suggested production deployment configuration, when in fact it should be rarely used as such.

The real power of entity EJBs, especially robust container managed persistence (CMP) engines, is the ability for a developer to develop a single entity EJB and then deploy that component multiple times, with each deployment tailored for a different data access scenario.

Yes, there are many studies that demonstrate entity EJBs with CMP have luster performance when compared with an SLSB with JDBC. But, there isn't a single study that leveraged entity EJBs using the "develop once, deploy n times" model. A system using entity EJBs following this model will have greater overall performance, robustness and reliability than an SLSB with multiple JDBC operations. Here are three reasons why:

1. An SLSB with JDBC will never be as robust in its ability to handle different data in a single server or a cluster. CMP engines already support lazy loading, aggressive loading, batch updates, optimized writes, field groups for CMP fields, field groups that incorporate relationships, optimistic locking policies, and transparent cache management through activation and passivation. What level of effort would it require to write an SLSB with JDBC to correctly implement all of these scenarios? You could painstakingly accomplish this for a single-server environment, but what about trying to coordinate actions for a clustered cache? SLSB with JDBC implementations have upper limits to their capability that

- entity EJB CMP containers have already exceeded.
2. The configuration of entity EJBs allows a developer to rapidly change the way a container behaves without any code modification. This provides a level of productivity not available with other technologies.
3. Test cases used on entity EJBs for performance often don't factor in the relative weights of how the data will be accessed in production. For example, Web-based systems are primarily read-only systems, yet typical test cases tend to use read-write transactional entity EJBs to gauge the performance for all scenarios.



Looking at CMP implementations in particular, it's reasonable to assume that, at the core of the container, every database query made by the container will be the equivalent query made by a developer that had to hard-code the query. Given the addition of container overhead, it's reasonable to assume that entity EJBs must therefore be slower than using SLSBs with JDBC. However, entity EJBs start to show a return on investment when they start creating queries that couldn't be hard-coded, such as batch updates, optimized writes, dynamic joins, and so on. When used in a clustered environment, these "special" container queries not only make a system more robust, but also more performant.

**Introducing: Architecture by Usage Pattern**

There are all kinds of approaches to the design/architecture of a system. Some architects design from the data up and others design from the presentation view down. Other approaches have developers focus on transaction and request granularity.

If you want to make the most of entity EJBs, I propose a new methodology: architecture by usage pattern. This process requires an understanding of the business domain, a data model, and some expectations as to the patterns of data usage.

Now in More than 5,000 bookstores worldwide

subscribe **Now!**

**FORFAST**  
DELIVERY

Go  
Online  
and  
Subscribe  
Today!



Helping  
you enable  
intercompany  
collaboration  
on a global scale

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

**SPECIAL**  
INTRODUCTORY OFFER  
**SAVE \$31\***  
HURRY! DON'T DELAY! OFFER EXPIRES APRIL 30, 2002

**WebLogic**

**Journal.com**

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



## WLDJ ADVERTISER INDEX

ADVERTISER	URL	PHONE	PAGE
Altaworks	www.altaworks.com	603-598-2582	7
BEA	www.developer.bea.com	408-570-8000	2,3
	831-431-1000	19	
Corda Technologies	www.corda.com	801-805-9400	41
Divine	www.divine.com	773-394-6600	45
JavaOne	http://java.sun.com/javaone/	888-886-8769	55
Object Design	www.objectdesign.com	781-674-5000	63
Performant	www.performant.com	866-773-6268	40
Precise Software	www.precise.com/wldj	781-461-0700	17
ReportMill Software	www.reportmill.com	214-513-1636	27
Resonate, Inc.	www.resonate.com	408-548-5500	20,21
Sitraka	www.sitraka.com	416-594-1026	13, 64
Thought, Inc.	www.thoughtinc.com	415-836-9199	61
WebGain	www.webgain.com	877-Web-Gain x15858	49
WebLogic Developer's Journal	www.sys-con.com	800-513-7111	53
Web Services Edge	www.sys-con.com	201-802-3069	28,29 42,43
Web Services Journal	www.wsj2.com	800-513-7111	51
Wily Technology	www.wilytech.com	888-GET-WILY	4
Wireless Edge	www.sys-con.com	201-802-3069	34,35
XML Journal	www.sys-con.com	800-513-7111	52

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.

## NeXT MONTH



Standards-Based Integration:  
The Impact of Web Services  
and J2EE

*Compelling opportunities that might  
provide a competitive advantage*

Exploring WebLogic JMX:  
JMS and J2EE, part 1

*An incredibly simple way to make Java  
applications manageable*

A Simple ADK for WLI's  
Business Process  
Management

*Making the processing of workflows  
independent of the BPM app*

The Next Frontier for Web  
Services: Dynamic Assembly  
*Relief from the repetitive work associated  
with maintaining separate code bases*





## Step by Step

1. Understand your business requirements and model your entity EJBs in such a way as to create in-memory domain objects that encapsulate the data in a format most accessible by the rest of the system.
2. Implement your entity EJBs to this model, ignoring behavior of the data.
3. Test the EJB using a standard read-write transactional locking scheme to make sure that you can create, read, update, and destroy instances.
4. Compile a list of usage patterns for the system will need to perform on the data and their relative weights to one another. For example, I interviewed companies that have built Web-based systems (see Table 1).
5. Create a separate deployment for each entity EJB for each usage pattern. For example, in WebLogic Server, changing certain flags can implement different patterns (see Table 2).
6. Partition your presentation logic along the lines of the usage patterns you defined.

Different use cases will access different deployments. So, for the “Read data for display” use case, this might be implemented by a single SLSB that directly accesses the read-only deployment. Since each usage pattern is a different deployment, they’ll each have a different JNDI name that they are bound to. Your presentation logic will determine which deployment to use by the lookup performed on the naming server.

## The Power...and One Drawback

The example discussed above is, of course, contrived, but it demonstrates the possibilities. In fact, the number of usage patterns that you could derive for your system is quite extensive. For WebLogic Server’s EJB CMP container, I counted nearly 40,000 different ways that an entity EJB could be configured to access a persistent store by taking the permutation of available values for each data-access deployment descriptor tag. Of course, most of these combinations would never be used, but this still quantifies the possibilities.

Inevitably, there’s a penalty for using this methodology: memory consumption. Each deployment of an entity EJB will consume additional memory based upon your cache settings. If it isn’t configured carefully, you could have the same PK in many different caches (one for each usage pattern). If you design a system that has each entity EJB deployed in 10 different formats, you’d potentially be replicating the data 9 times!

There’s a solution, however: determine the overarching cache size available for a single EJB and then set the individual deployment limits to be a ration based upon their expected usage. For example, if you have an entity EJB X that can allocate 100,000 instances in its cache for all usage patterns, the “Read data for display” deployment would be set to 85,000 since its expected usage pattern is 85% of the activity in the system. The other usage patterns would have their cache sizes set appropriately. This is a simple, yet elegant, way to allocate the appropriate amount of memory to each operation based on its activity level in the system.

## Conclusion

BEA is very excited about EJB 2.0, not simply because it’s a new version of a popular specification, but because of the possibilities that a solid container implementation can fulfill. In WebLogic 7.0 we’re extending our CMP implementation by providing transactional clustered caches, optimistic container locking, aggressive relationship loading, and more.


WebLogic Server 7.0 beta is available for public download now. 

TABLE 1

Usage Pattern	Relative Weight
Read data for display	85%
Read-write data, inserting records, and deleting records (all requiring transactional support)	10%
Batch update data	5%

Sample application data access statistics

TABLE 2

Usage Pattern	WebLogic Configuration
Read data for display	read-only <concurrency-strategy> (weblogic-ejb-jar.xml) with invalidations from a separate read-write deployment and NotSupported transaction demarcation (ejb-jar.xml).
Read-write data, inserting records, and deleting records (all requiring transactional support)	Database <concurrency-strategy> (weblogic-ejb-jar.xml) with <finders-load-bean> set to true (weblogic-ejb-jar.xml) and Required transaction demarcation (ejb-jar.xml).
Batch update data	<finders-load-bean> set to false (weblogic-ejb-jar.xml), Required transaction demarcation (ejb-jar.xml), and only invoking setXXX(..) methods of EJBs. This will cause a findXXX(..) method to return a Collection of PKs, but no data will be loaded into cache. A client can iterate through the Collection, modifying fields by calling individual setXXX(..) methods. When the transaction is committed, an optimized query to write out all modified fields of all PKs will be generated.

Entity EJB configurations for different access scenarios

# JavaOne

<http://java.sun.com/javaone/>





# A STRATEGY FOR THE FUTURE



AN INTERVIEW WITH  
**SCOTT DIETZEN**  
CTO OF BEA SYSTEMS

**WLDJ recently sat down with Scott Dietzen to talk about BEA's strategy for the future, and some of their upcoming releases.**

**WLDJ:** *Could you tell us about your new role as BEA's overall CTO and what you think are your challenges in the next year?*

**SD:** I've been given a great opportunity to take the position of CTO for the overall company BEA is investing in areas on all levels of the application server that is becoming ever more critical to our enterprise customers, especially in the area of integration. With the emergence of J2EE standards around adapters and the new XML technologies around Web services and workflow, the industry is reaching critical-mass technology for standardizing the way the application and DB2 integration is done. Today, integration is a very fragmented, proprietary market, one that causes a lot of pain to our large end-user customers, our large systems inte-

grator partners, and in fact a lot of our independent software vendors. This is due to the complexity of integrating their applications with other applications, both custom and off-the-shelf in the enterprise. Customers want to solve these problems up front. You no longer get to just deploy the application and figure out how to integrate it later on. Customers are being much smarter about building in that solution up front.

BEA has been investing in the integration area for about three years, and also had a long-running initiative around portal personalization. I think the broad realization is that the user interface and business applications don't tie together one-to-one as often as they used to. What we see across our customer base is the desire to aggregate multiple

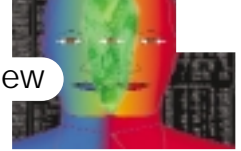
business applications together with a single user interface so that the people using those systems keep 4, or 20, different applications up on their desktop, cutting and pasting between those applications. They want to see a unified, personalized view of the back-end application infrastructure. I think BEA was present in investing here.

One of my charters is to take the innovations we've been doing over the past three years and integrating them together so they come under the same installation and management environment with a common look and feel. It will be a very natural extension for any customers using our application server.

That's the most critical challenge I face in taking on the CTO job – finishing the delivery of BEA's vision around the e-busi-

ness platform. These platforms are much more than just the platform for custom application development – which is what the application server is. There's a critical integration component which includes business process modeling, Web services, and EAI technologies around the adapters. There's a critical next-generation user-interface component which includes portal technology aggregation, personalization, declarative interface building, and ease-of-development components, bringing all of this e-business platform programming capability to the developer who doesn't want to learn all the intricacies of J2EE and distributed object modeling.

On the business side, this is a much more conservative market for IT spending. BEA has managed to continue our growth in



market share although the overall market is not growing nearly as quickly as it was a year-and-a-half ago. Customers are far more up front in wanting to demonstrate return on investment (ROI) business cases before they

head down this path. I think customers are taking a much more mature approach in terms of deciding what business imperatives they should focus on. At the end of the day, we're best served by delivering real value

to our customers and that always entails ROI. The challenge and the opportunity for BEA has been helping our customers document their ROI so we can transfer that to others.

**WLDJ:** *What are some of the challenges facing BEA in the coming year?*

**SD:** I think the challenge is that we have some very strong competitors. BEA is a small but rapidly growing company that's got some great strategic alliances, but we're clearly dancing among the giants and we need to continue our history of "co-opetition" where we complement our competitors. We work very closely with Microsoft, for example, on Web services initiatives, and making our product integrate very tightly with the Windows environment. We have to work closely with IBM. Today we work closely with Sun and Intel.

**WLDJ:** *How are you ensuring that BEA and its partners, as well as competitors' products, function smoothly together?*

**SD:** My role is to drive the strategy for our R&D organizations, help lay out the road map, investigate new areas of technology, expand our product line, and how we will integrate the products we have today. There are occasions when we have multiple approaches that are being surfaced inside of our engineering organization for tackling a particular problem. BEA feels strongly that it is our job -- and my job in particular -- that we help pick the best mechanisms and assure that they win so that we don't introduce complexity to our customers. I think some of our competitors have done a far less effective job of that. They've allowed internal competition to be surfaced to the customers -- four or five different workflow solutions combined into a single product. We don't believe that it serves our customers very well if the customers have to make the choice.

Fortunately, I have a team that I encourage to specialize in various areas of the products. On the outwardly focused side, my team is responsible for coordinating BEA's participation in standards bodies. BEA participates in over 20 different standards organizations. My team spends the most time on the W3C and XML/Web services standards and initiatives, and the Java platform standardization initiatives. Our team is focused on ensuring that the right protocols, and the right programming models, are defined, continuing our history in offering very strong investment protection for our customer base. It's how we have become the market leader. We were more aggressive than our competitors in embracing and delivering standards-based technology to the market, which we're convinced everybody wants.

**WLDJ:** *Considering the standards you've mentioned, do you consider standards bodies to be innovative or reactionary?*

**SD:** Actually, I think that each standards body has a very productive life. It's like a technology adoption curve. In the early stages they can make very steep progress; when we've got a compelling business case in technology transformation happening at the same time, you get a lot of excitement. I think both the Java standardization process and the W3C standardization process are on that steep curve, Java being a little higher up the ramp since the XML and Web services side -- the integration side -- are a little more recent effort. There are critical innovations that have already happened, such as SOAP and WSDL, and our efforts and those of Microsoft, IBM, Sun, and others to unify the industry around a single Web services stack. I think we're making good progress; I don't think we're done yet. There are key things to come. We have to be



able to insure reliable delivery guarantees for asynchronous Web services. We have to be able to secure those asynchronous Web services. Asynchrony is so critical because it drives most Web processing today. If you look at Charles Schwab's training site, or any of its competitors, they don't service trade requests while you wait. They capture your trade request and then return, and you see the results maybe a few seconds later, or it might take a while to process depending on whether the market's open. We're very bullish that the sweet spot for Web services is going to be similarly around messaging; reliable delivery guarantees and security for asynchrony are the two most critical areas for furthering the

W3C. BEA is very aggressive in this particular area.

**WLDJ:** *You said that you see Web services developing around asynchrony. Do you see any other areas where it is evolving? And what is BEA's strategy in attacking that market?*

**SD:** We've got to get a critical mass around the set of standards to make all of this work. For synchronous Web services, like getting a stock quote or the weather, we're already there. The combination of SOAP/WSDL that we have in the marketplace is very good at that. We've already tightly integrated the Web services model with our Java container. We can transparently attach Web services to Enterprise Java Beans (EJBs) and to JMS message queues and topics for pub-

lish/subscribe; to make the Web services model transparent. That means you can use Web services as your design center or come in with a WSDL definition and we can generate Java bindings for an EJB, or a message driven bean. You just need to fill in the business logic or we can go in reverse.

Most of our existing customers have J2EE applications or that's the design center they're used to building to. We can go in and automatically generate the WSDL that describes that service, and define a SOAP binding so that it is fully automated from the developer's perspective.

One of the principal challenges is about taking not just Web services but the greater e-business platform and making it substantially easier. If you go back and look at the history of client/server, the first wrappable user interface were very rich. If you go back to the MOTIF toolkit and the early SmallTalk products, you get a very rich programming model for richly interactive applications. But it took the great object programmers, the people with the computer science backgrounds generally, to produce and build these sexy user interfaces. Sometime later, PowerBuilder and Visual Basic brought the richness of the graphical user interface to the business programmers – people who are less comfortable with the intricacies of a complex object model and more inclined to think procedurally.

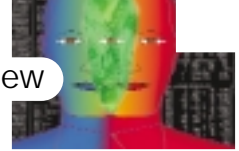
This is very much a portion of the market that the Java community needs to reach more effectively than we have to date. We've done a great job with the J2EE rock stars, now we need to turn our attention to people whose expertise is more in business applications, who are less inclined to want to go off and learn the intricacies of J2EE programming. I think this is perhaps BEA's single most crucial area of investment, making e-business easy, just like PowerBuilder and

Visual Basic made client/server easy. I don't think anyone has done that to date. If you look at the .NET architecture, that's clearly the path they're heading down, but VB.NET is quite a big leap for the average VB programmer to pick up and I don't think our other big competitors have figured out how to make e-business application development and integration in portals easy. We think we can make it easy, and we think we can take those integrations and share them with the W3C and the Java community so that we're very happy to compete on offering the best of breed in standards-based technology.

**WLDJ:** *Would you say that the following of standards is how BEA differentiates itself from Microsoft, Oracle, and IBM?*

**SD:** Historically, I think we've differentiated on adherence to standards. We continue to lead the market in delivering production implementations of J2EE standards. We've been shipping EJB 2.0, for example, for more than a year. And we've been tracking the spec very aggressively, something that most of our competing products are still not delivering today, especially with support for early production deployment. Adherence to standards before the rest of the market has been one of the things that made BEA the market leader. We will be very aggressive to continue with our innovation. The architecture of WebLogic Server is one of the things that makes this possible. We've produced a very general purpose of quality-of-service infrastructure for the performance and availability of fault tolerance through our clustering architecture. These generic services map to an orgy of sockets and spreads. It's been done in such a way that we can plug in underneath J2EE services so that we don't have to go into each of the J2EE personalities and engineer all of this stuff directly. We can





engineer an overall platform and then wire it in; that's allowed us to move very quickly.

Another area we've historically continued to differentiate on is scalability and performance. We're running workloads on WebLogic today that are approaching the workloads we run on Tuxedo. Thousands of business transactions per second where business transactions may equate to a three- or four-database operation. These are really large business-critical workloads on Java technology today, which is very exciting for us.

We've differentiated ourselves with our partnering strategy. We've got more systems integrators and more independent software vendors building their products on top of WebLogic. Many of our customers buy off-the-shelf applications from Peoplesoft or Vignette or Ariba, and so on. They're already running WebLogic products in-house and were able to win business for their customers as well. BEA is one of the few vendors that doesn't have an agenda in terms of a particular hardware; it's the only one of the leading players that doesn't have a hardware OS database agenda. We're quite comfortable plugging into all of those different platforms.

Looking forward, I see this ease-of-use as one of our key differentiators in addition to our focus on making integration and next-generation Web user interface around portal dramatically easier.

**WLDJ:** *We've heard about a new IDE called "Cajun." What can you tell me about it?*

**SD:** The "Cajun" product has been in limited alpha release for a couple of months and we will be entering a general availability beta shortly. It will be available on the BEA Web site for download. "Cajun" offers a technology to developers who don't want to deal with all the intricacies of J2EE. Some of the existing J2EE rock stars will look at

"Cajun" and say "It's a neat tool but it's not for me because I'm much more comfortable and empowered when I crank all the J2EE and JMS code directly." We view it as a great vehicle for taking existing EJB applications that have been developed and surfacing them to J2EE experts. There's no magic here. What PowerBuilder offered was a more tightly constrained framework; you could do 80 or 90% of the work you needed for user interface design with dramatically less complexity by providing a framework.

"Cajun" has a framework built around Web services and Java development paradigms like EJB and JMS in it. We're making it very easy to construct processes that consume existing Java apps, consume Web services, and transform XML without having to learn J2EE. It still has J2EE underneath the covers. We're mapping this to all of the Java standards that our existing customers know, but hopefully we're appealing to an audience that doesn't have the time or inclination to spend learning all of J2EE and we think we can broaden the marketplace for our own products and for Java by doing so.

One of the things I'm most excited about is being able to provide an environment that allows the systems programmers and the business programmers to collaborate much more intimately than they have historically. If you look at, for example, the history of Microsoft, developers tended to be Access people or SQL Server people, Visual Basic peo-

ple or VB C++. There was sort of an expert class and a business programmer class and more computer scientists got the heavy lifting done to run the enterprise. I think that BEA and the Java community are uniquely positioned to tie those two classes of users together in an environment that empowers both.

And that is exactly what "Cajun" is for. This is a long-running strategy and a major area of investment for BEA. We've already come up with some truly innovative ideas, like the notion of an annotation grammar for Java that makes it very easy in the source code to specify properties that will go off and generate code; define a class and say that you want this class to be invoked asynchronously; introduce a JMS store and a message driven bean that surround it without the programmer having to do it. It's these prepackaged stylized uses of J2EE that are cookie cutters so they can be plugged in. The J2EE developer is free to go in and add additional specializations should that developer feel they're necessary. For most of the "Cajun" programmers, the expectation is that they will be more comfortable working at a level where they don't have to understand the application programming interface.

"Cajun" is very much a Java tool. When you program in it you see and touch Java. There clearly is a visual component to the environment, but it is for Java programming. You use a procedural and object model to develop applications. What you don't have to do is learn all of J2EE. We're definitely trying to appeal to the Java programmer who either doesn't want to deal with J2EE for a particular app because their use of J2EE is pretty stylized, pretty standards-based or they don't want to learn it. We're not hiding Java; we are hiding J2EE, although in such a way that it's available for developers who choose to get active in it.

**WLDJ:** *You talked about how "Cajun" can help the developer who isn't quite the hard-core Java person. What about wireless development products? Are there plans to make J2ME or WAP development easier as well?*

**SD:** We have seen a movement toward multichannel applications. New applications are developed, they're not just being developed with a Web-user interface in mind, but also a user interface for wireless devices – the phones, the PDAs. We even have customers looking at appliance-type devices, even if they're not implementing for them yet. Voice is often neglected in the multichannel story but voice is a very powerful interface in the sense that it leverages existing technology. We've all experienced these really frustrating voice portals where we have to go through 18 layers of menus when we're selecting items four and five and it takes us 5 to 10 minutes to get to the information that we want. We have to make these voice portals much more powerful if we're ever going to sell users on leveraging them rather than talking to real humans, which is expensive. The technology here is around Voice XML, speech recognition, and personalization.

We've been distinguishing two models of the multichannel client. I would call one the generic client, one where voice applies, the phones, and text messaging, where we see a broad convergence around the need for text messaging, whether it's SMS or e-mail-based or even instant messaging on the Web I think that these things will come together over time into a text-messaging solution that works across all our multi-devices and page browsing. What they have in common is that there's no business logic whatsoever required on the client device. For typical enterprises, that looks like a really compelling solution



because if you're an Amazon, or a Charles Schwab, you don't want to force your customers to install special-purpose application software on your devices because you've got customers who run too many devices for that to be feasible.

There's also a rapidly growing market for programming clients. We're talking about the J2ME stack – the MIDP profile specifically targeted at phones – or building small Java apps that run on these small, slick grids on client devices and interoperate with app containers like WebLogic across the Web. BEA is collaborating aggressively to deliver technology for both models. We are an executive board member of the J2ME initiative working on the surfacing technologies like Web services, and SyncML for disconnected operations and a standard way for Java clients as well as a standard way for server-side apps. There's a new JSR for provisioning, it's a Java vending machine JSR that allows Java applications to be defined and tied into the server and then dynamically downloaded and installed on the client without the user having to do anything during the real-time provisioning of new application capabilities.

**WLDJ:** *With the introduction of "Cajun", is BEA going after the Visual Basic crowd?*

**SD:** I can't say I have a lot of insight into Microsoft. I think that from their perspective, Microsoft does see J2EE as a competitor. BEA values the fact that we've had a good working relationship with Microsoft. We've done a lot to make our product and our platform Windows friendly by tightly integrating it with COM, for example, so you can build beans and plug them into Excel spreadsheets very easily. We've tied in active directories and we've collaborated on Web services and SQL Server performance benchmarks. We feel that our customers want us to

allow them, when they're building and deploying on Windows to be able to take advantage of those native services. From that point of view, I think that Microsoft looks at us and says this is an important, value-added ISV for our platform technology. We have a good relationship there.

There is no doubt that .NET is going to compete with J2EE and the only scenario that I can imagine is long-term coexistence for both platforms. I think the Java community is already headed towards coexistence. This proposition of portability across platforms is critical to the enterprise ISV. If you're a Peoplesoft, or a Siebel, or an SAP, I don't believe that the current market conditions would allow you to choose a solution where you can only deploy on Windows, because customers want to run applications on UNIX, Windows, and even mainframes. That's part of the appeal of the Java platform. At the same time, Microsoft has its small enterprise marketplace already jumping on the .NET bandwagon. The industry is heading toward long-term coexistence of J2EE and .NET, and they will absolutely compete. Competition is good. The Java community shouldn't shy away from it. It will continue to drive innovation in the Java camp. We have to keep up with the innovations that are happening on the .NET side and stay ahead. You have to maintain your lead. It's one of the Java community's strengths: hundreds of companies innovating and sharing those innovations, and picking the best and growing our platform. I think that will be a challenge for a company as innovative even as Microsoft. Let me also add that the most compelling positive that has come out of this is our continuing convergence on a single Web services stack. As .NET and J2EE coexist we have a much more compelling value proposition than we did previously.



**WLDJ:** *Anything else you want to tell the readers?*

**SD:** We have to continually challenge the Java community. It has to stick together and help make our environment easier. I think it's one of the daunting challenges facing the Java community. Java has grown in so many different areas, continuing to innovate and standardize broadly. We have to sustain our pace of innovation, but also look at the work we've done and think about how we can simplify it and make it better. We've been pushing in a lot of different directions and I want us to maintain that pace of innovation, while at the same time not introducing a complexity that I think can plague later stage standards. Standards bodies seem to have a great early stage life where a lot of innovation happens, and then you get creeping complexity. I would charge the greater Java community with finding ways to tighten and simplify the Java platform that will allow it to continue its growth.

Let's aggressively reach out to developers who aren't comfortable with complex object forms because they are much more comfortable with the knowledge of the business rather than the knowledge of computer science. Java is going to grow and thrive. We have to continue to educate

the world in Web services and XML. These are great technologies. There's a very natural synergy between distributed objects component models like EJBs and Web services. Inter-application integration, I believe, is going to move away from binary protocols like IIOP toward technologies like Web services because it allows for a looser coupling and a less fragile integration model. It provides more flexibility to unplug solutions and plug in new ones.

Web services don't compete with J2EE; they're an integration model, not a component model. We need the Web service and multichannel innovation to allow our platform to grow and thrive, at the same time recognizing this long-term need to coexist effectively with .NET. We will compete as we coexist because without that coexistence the whole train of application integration across the Web won't happen. What the Web has meant to date is tying the browser together with our documents and our business applications but the second generation of the Web will be deployed applications that much more seamlessly and easily interconnect. That's a great opportunity for Java to be working very closely with those developing Microsoft .NET architecture to assure that they will interoperate.

# Thought, Inc.

[www.thoughtinc.com](http://www.thoughtinc.com)

# News & Developments

## Resonate Delivers New Solution for In-Depth Monitoring and Management of Applications

(San Diego, Calif) – Resonate, the provider of active service level management solutions for e-business applications, has announced a new Resonate Command Module for BEA WebLogic Server 6.1 using BEA's implementation of Java Management Extension (JMX) technology, the standardized specification for managing J2EE application server environments. The Resonate Command Module for BEA WebLogic 6.1 works in conjunction with Resonate Commander Solutions to deliver a comprehensive solution to identify, diagnose, and resolve problems for the application server and the custom applications built on top of the application server.

Resonate Commander Solutions closely monitor WebLogic Server and custom application metrics and correlate those metrics with infor-



mation on other critical components of the infrastructure. Resonate enables automated actions to proactively address any problems and mask end-users from experiencing any degradation in availability or performance.

[www.resonate.com](http://www.resonate.com)

## Sitraka Announces Performasure Version 1.1

(Toronto) – Sitraka, a leader in J2EE Performance Assurance, has announced the release of Sitraka Performasure version 1.1, a transaction-centric diagnosis tool for multi-tiered J2EE applications that enables e-

business performance teams to measure, analyze, and maximize performance prior to application deployment.

Performasure provides insight into the J2EE performance data stack for rapid prob-



lem identification and resolution. Featuring a rich user interface, Performasure provides highly intuitive graphical views into application server configuration and run-time information, JVM performance, database performance, network utilization, and OS and hardware platform metrics. Its unique Tag-and-Follow technology reconstructs the execution path of end-user transactions, highlighting performance hotspots to accelerate root cause diagnosis of unacceptable e-business response times.

[www.sitraka.com](http://www.sitraka.com)

## TeaLeaf Debuts IntegriTea

(San Francisco) – TeaLeaf Technology has unveiled TeaLeaf IntegriTea, the latest generation of TeaLeaf's software. IntegriTea is the first Web application management solution designed to contain the skyrocketing costs and bottom-



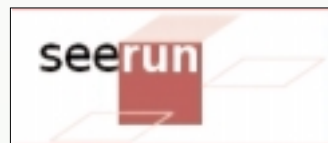
line impact of Web site failures, site maintenance, and problem resolution. By providing visibility into the functional integrity of Web applications, customers report increased reliability and accelerated problem resolution.

IntegriTea bridges the gap between traditional application performance and system man-

agement tools. Using patented technology, IntegriTea captures the composition of every Web page sent to every user alert, alerts IT to possible problems, resolves errors, and communicates problem resolution across the enterprise with visual documentation of Web application failures. [www.tealeaf.com](http://www.tealeaf.com)

## SeeRun Showcases Customer Lifecycle Management Suite

(San Diego, CA) – SeeRun has integrated its real-time CRM



analytics suite with BEA WebLogic Server to provide e-businesses with a solution that enables rapidly increasing customer acquisition conversion rates and significantly reducing customer acquisition costs. In addition, SeeRun has embedded BEA WebLogic Server in its platform to provide its customers with a solution that provides a proven, open-standards based, reliable e-business infrastructure. [www.seerun.com](http://www.seerun.com)

## iWay Software Announces JCA-Compliant Connector/Adapter Offering

(New York City) – iWay Software, an Information Builders company that accelerates business integration, has announced the release of iWay Enterprise Connector for J2EE Connector Architecture (JCA). This connector allows Java developers to rapidly create applications that incorporate functionality and data from more than 160 commonly used enterprise information assets on 40 platforms.



As part of the iWay Enterprise Integration Suite, the JCA Connector helps to further reduce the complexity of integration by easily allowing packaged applications – such as CRM, ERP, SFA – access to back-end data sources. The iWay Enterprise Integration Suite unifies more data sources and applications than any other single vendor with more than 140 adapters and connectors. The connectors provide an API version 2 interface that has been tested on BEA WebLogic Enterprise as well as all other major platforms.

[www.iwaysoftware.com](http://www.iwaysoftware.com)

## NEON Systems Delivers J2EE Connector Architecture Adapters

(Sugar Land, Texas) – NEON Systems Inc., a provider of data access, legacy renewal, and OS/390 enterprise system management solutions, has announced additional support and functionality for J2EE with the introduction and certification of Shadow Direct JCA adapters to support BEA WebLogic Integration.



Shadow Direct is an open, standards-based solution for integrating mainframe data sources and transaction environments to client/server and N-tier computing environments. [www.neonsys.com](http://www.neonsys.com)



# Object Design

[www.objectdesign.com](http://www.objectdesign.com)



# Sitraka

[www.sitraka.com](http://www.sitraka.com)