CASE STUDIES FOR BETTER DEPLOYMENT

BEA WebLogic DEVELOPER'S JOURNAL

NOVEMBER 2002 - Volume:1 Issue:11

weblogicdevelopersjournal.com

NOW SHIPPING

WEB SERVICES RESOURCE CD
THE SECRETS OF THE WEB SERVICES MASTERS
MORE THAN 1400 EXCLUSIVE WEB SERVICES & JAVA ARTICLES

SEE DETAILS ON PAGE 41

RETAILERS PLEASE DISPLAY UNTIL DECEMBER 31, 2002
$15.00US $16.00CAN

11

0 71486 03424 7

SYS-CON MEDIA

In Search of Operational Excellence

Precise

www.precise.com/wldj

Intel

www.intel.com/ad/bea

Wily Technology

www.wilytech.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 PHOTOCOPY OR ANY INFORMATION STORAGE AND RETRIEVAL SYSTEM, WITHOUT WRITTEN PERMISSION. FOR PROMOTIONAL REPRINTS, CONTACT REPRINT COORDINATOR. SYS-CON PUBLICATIONS, INC., RESERVES THE RIGHT TO REVISE, REPUBLISH 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 DEVELOPER'S JOURNAL.

BEA WebLogic DEVELOPER'S JOURNAL

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 EDITORS
JAMIE MATUSOW, JEAN CASSIDY

ASSISTANT EDITOR
JENNIFER STILLEY

WRITERS IN THIS ISSUE
KUMAR ALLAMRAJU, PETER HOLDITCH, ANBARASU KRISHNASWAMY, JOSEPH KROZAK, DWIGHT MAMANTEO, SHYAM NAGARAJAN, FRANCO NEGRI, SAM PULLARA, DAVE TRUMAN, JASON WESTRA

SUBSCRIPTIONS
For subscriptions and requests for bulk orders, please send your letters to Subscription Department.
SUBSCRIPTION HOTLINE:
888-303-5282
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

COO/CFO
MARK HARABEDIAN

VP, BUSINESS DEVELOPMENT
GRISHA DAVIDA

SENIOR VP, SALES & MARKETING
CARMEN GONZALEZ

PRODUCTION CONSULTANT
JIM MORGAN

ART DIRECTOR
ALEX BOTERO

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

ASSISTANT ART DIRECTOR
TAMI BEATTY

VP, SALES & MARKETING
MILES SILVERMAN

ADVERTISING SALES DIRECTOR
ROBYN FORMA

ADVERTISING ACCOUNT MANAGER
MEGAN RING-MUSSA

ASSOCIATE SALES MANAGERS
CARRIE GEBERT • ALISA CATALANO
KRISTIN KUHNLE • LEAH HITTMAN

VP, SYS-CON EVENTS
CATHY WALTERS

CONFERENCE MANAGER
MICHAEL LYNCH

REGIONAL SALES MANAGERS
MICHAEL PESICK • RICHARD ANDERSON

ASSISTANT CONTROLLER
JUDITH CALNAN

ACCOUNTS PAYABLE
JOAN LAROSE

ACCOUNTS RECEIVABLE
KERRI VON ACHEN

ACCOUNTING CLERK
BETTY WHITE

WEBMASTER
ROBERT DIAMOND

WEB DESIGNERS
STEPHEN KILMURRAY • CHRISTOPHER CROCE
CATALIN STANCESCU

ONLINE EDITOR
LIN GOETZ

CUSTOMER SERVICE MANAGER
ANTHONY D. SPITZER

CUSTOMER SERVICE REPRESENTATIVE
MARGIE DOWNS

JDJ STORE MANAGER
RACHEL MCGOURAN

EDITORIAL OFFICES
SYS-CON Publications, Inc.
135 Chestnut Ridge Road, Montvale, NJ 07645
Telephone: 201 802-3000 Fax: 201 782-9637
SUBSCRIBE@SYS-CON.COM
BEA WebLogic Developer's Journal (ISSN# 1535-9581) is published monthly (12 times a year)
Postmaster: Send Address Changes to
BEA WEBLOGIC DEVELOPER'S JOURNAL,
SYS-CON Publications, Inc.
135 Chestnut Ridge Road, Montvale, NJ 07645

SYS-CON MEDIA

Looking Back to See the Future

BY JASON WESTRA
EDITOR-IN-CHIEF

I recently upgraded a small WebLogic 6.1 application to WebLogic 7.0. The process was really quite simple. I attribute this smooth transition to the application's standard use of J2EE components and to WLS 7.0's backwards compatibility! I really only had to do a few configuration changes to get it working. In particular, the JMS store and JMS paging store are no longer allowed to have the same JDBC persistent storage prefix. They want their own tables in WLS 7.0. Next, there was a minor, new element identifying the version of the configuration, ConfigurationVersion="7.0.0.0". My biggest worry was security because the security in WLS 7.0 changed dramatically and the application's fileRealm was "deprecated" so to speak. However, the server started in backward-compatibility mode and ran all of the EJBs out of the gate despite the fact that their DTDs were still for WLS 6.1 and the server was using the old fileRealm.

Backward compatibility provides application development teams with the ability to migrate at their own pace, taking advantage of new features of WLS as they can or as required by the application. If new versions of WLS did not provide backward compatibility, then many installs would fail to upgrade. This would be a disaster. IMHO, not upgrading is setting yourself up for failure in the future when new features such as Web services, JMX, and advanced clustering are required. I'm weary about time spent on migrating to new platforms and versions. Thankfully, with WLS we don't have to be concerned.

This month's ***BEA WebLogic Developer's Journal*** includes some success stories using the BEA WebLogic Platform and products that integrate with the platform. I'm fond of this issue because these articles provide a real-world basis for WebLogic applications across numerous industries, not just e-commerce. Likewise, it's a look back to see how WebLogic has been used to solve unique problems, or perhaps not so unique problems. For instance, looking back and analyzing case studies can be a great benefit in solving future problems. They often reveal benefits and disadvantages of particular approaches and designs that application architects or project managers might not have prepared for otherwise.

This month, we have case studies covering successes on the WebLogic Platform utilizing products such as PANACYA's BeX application management agents, Cacheon's Migrator, and Sitraka's DeployDirector. The articles span the application infrastructure, financial services, and dental industries respectively. Also this month, we have Part 2 of Joe Krozak's series on advanced integration of PeopleSoft utilizing Web services, and the regular crew of Peter Holditch and Sam Pullara adding wisdom as always.

On a final note, next month will be the twelfth issue of ***WLDJ***. Looking back, it has been a fun year, and I have been committed to providing you with the best literature on WebLogic in the industry. In the future, I promise ***WLDJ*** will remain true to this commitment.

AUTHOR BIO...

Jason Westra is the editor-in-chief of ***WLDJ*** and the CTO of Evolution Hosting, a J2EE Web-hosting firm. Jason has vast experience with the BEA WebLogic Server Platform and was a columnist for ***Java Developer's Journal*** for two years, where he shared his WebLogic experiences with readers.

CONTACT: jason@sys-con.com

In Search of Operational Excellence

MAINTAINING A LEADERSHIP POSITION BY MANAGING CUSTOMER-FACING APPLICATIONS AND SERVICES CRITICAL TO BUSINESS OBJECTIVES

BY **FRANCO R. NEGRI**

AUTHOR BIO...

Franco Negri is the founder, CTO, and Chief Strategist of PANACYA. In a 23-year career with leading suppliers and consumers of advanced management technology, Franco developed a keen understanding of market needs and a strong vision for the next generation. He was most recently VP, Product Marketing and VP, Research & Development at Computer Associates, where he was responsible for Unicenter TNG - CA's flagship Enterprise Systems Management product line.

CONTACT...

franco.negri@panacya.com

The Industry Challenge: Realigning IT and Business

Companies that aspire to lead their industries continue to find ways to optimize themselves in an endless pursuit of excellence. The race is on for many leading companies to leverage technology and provide enriched online services to their customers in order to maintain their positions and, in some cases, distinguish themselves from their competitors. It sometimes seems as though every month brings new innovation that enables and enhances a company's ability to expose its unique value through online Web services.

With the advent and standardization of technology inspired by the Internet, companies have access to a common business medium that allows flexible, rich products and services to be offered directly to customers and business partners. The enablers of these business services are mostly technology dependent and based on powerful and complex architectures. As a result, leading companies are turning to their IT departments to develop, provision, and maintain high-quality standards for these customer-facing applications and services.

WHERE BUSINESS AND IT OPERATIONS DISCONNECT

One of the most profound problems that exists today in IT is the pronounced disconnect between IT operations and the business lines that rely on IT to proactively manage business services in order to thrive in a highly competitive environment. Due to the complexity involved, IT has traditionally managed applications and infrastructure components from the perspective of specific devices and components, including network and systems, applications, and databases. Today, modern Web services provide automated online methods for customers and business partners to access information and services directly in a much more efficient manner. On one side, business drives the requirements for this complex online world, expecting IT to create, deploy, and assure high-quality, consistent online services that operate without outages and perform optimally. On the other, operational, side, IT often manages these business services by monitoring the health of the individual components, not from the more holistic perspective of the business services or processes these technology components enable.

This is where the misalignment between business and IT is amplified, and where IT operations become a bottleneck. This is particularly true if IT engages legacy approaches and tools designed to manage applications and supporting infrastructure in technology silos, as though they aren't related to the very business they support.

With the advent of new accelerated business models, like that of e-business, IT has had the difficult, if not impossible, task of keeping up with business demands. The time has come for a new operational approach to application and infrastructure management that enables IT operations to perform in lockstep with the business lines it is supporting – running IT as a business. This article outlines a new business-centric approach to IT operations management, along with a new software solution designed from the ground up that enables this approach, and a use case of a technology leader, BEA Systems, Inc. BEA adopted PANACYA's eBusiness Application Performance Management Suite to achieve business-focused operational excellence and help maintain not only its technology leadership but also its continued leadership in serving customers and business partners.

The BEA Business Driver

BEA wanted to extend its leadership in the application infrastructure market, which enables enterprises to rapidly develop, deploy, manage, integrate, and secure enterprise applications. To do so, they set out to enhance the quality and reach of services to their users and business partners with enriched online services, including a developer community (dev2dev), a self-service and support portal (eSupport), a partner network (PartnerNET), and www.bea.com, to mention a few. Focusing on customer service, BEA established an "operational excellence" initiative to define and establish operational processes and management services to support these critical online services. The idea was to extend BEA's high-quality standards for excellence in its products to operations and its own online services. BEA's IT operations team subsequently set out to find enabling technologies to assist in enhancing services and operational efficiencies.

A HIGH STANDARD FOR APPLICATION AND INFRASTRUCTURE MANAGEMENT

As with many progressive IT organizations, BEA's IT operations team wanted more than just the ability to detect the availability status of systems and network components. They wanted proactive visibility across all applications and supporting infrastructure in order to detect performance bottlenecks and faults before they impacted their end users' online experience. With the bar set very high, and consistent with its own high-quality business standards, the BEA operations team set stringent guidelines and requirements for the management system that would empower IT operations to support its business-critical online services.

After evaluating solutions from 13 of the leading infrastructure, applications, and network management vendors, the team selected PANACYA as an integral part of the solution set. Following are some of the selection criteria and also the guiding principles for IT operational excellence.

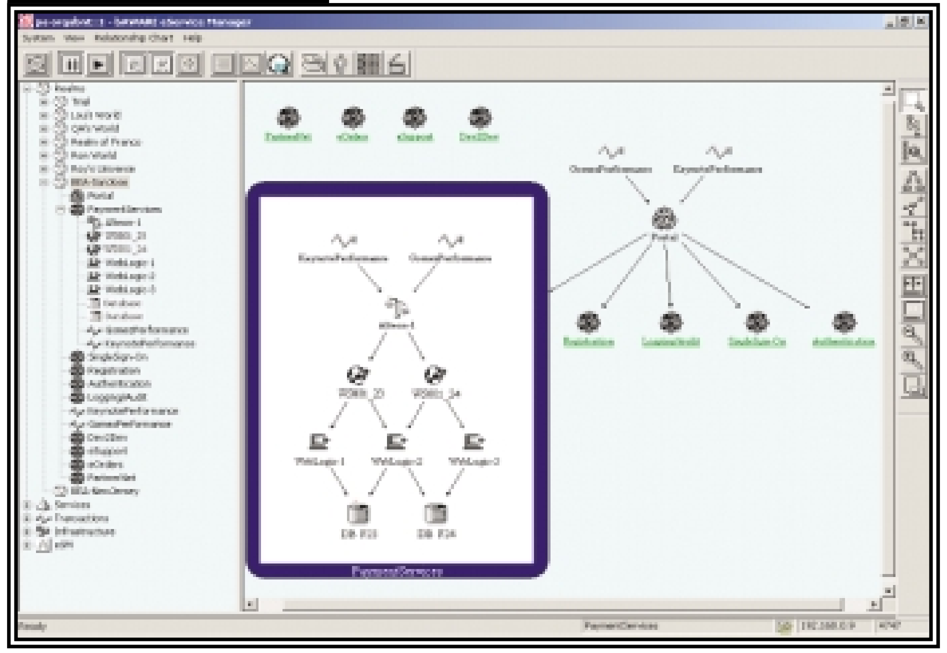
- ***Proactive problem identification and resolution***
  - *Granular and proactive application visibility and control:* The ability for a management solution to provide granular visibility into applications down to the management of J2EE components, applets, and methods.
  - *Intelligent root-cause analysis:* The ability to not only detect probable faults but also determine the detailed root cause of an emerging problem before it causes a fault and/or service outage.
  - *Correlating faults between applications and infrastructure:* The ability to easily correlate information among the components that deliver online services from the network to the provisioning applications themselves and ultimately with the customer experience in real time.
  - *End-to-end service-level monitoring:* The ability to view critical applications and associated infrastructure components holistically and from a service perspective. BEA wanted to understand the relationships and dependencies among components and how they interact in real time to provide online services. BEA also wanted the ability to enable this service-centric monitoring across IT functional areas of responsibility to provide a cohesive and collaborative business-focused operations capability.
  - *Managing the entire service delivery*

stack: The ability for the management system to support the entire infrastructure stack and any application. Also, the ability to support not only the current environment but also one that would adapt to future requirements and associated applications.

– *Alerting, notification, escalation, and reporting:* The ability to alert and notify operations personnel and business line managers with information pertinent to their area of responsibility. BEA also wanted an automated alerting and escalation capability in accordance with their best-practices operational policy. Additionally, BEA required a robust real-time and historical reporting capability.

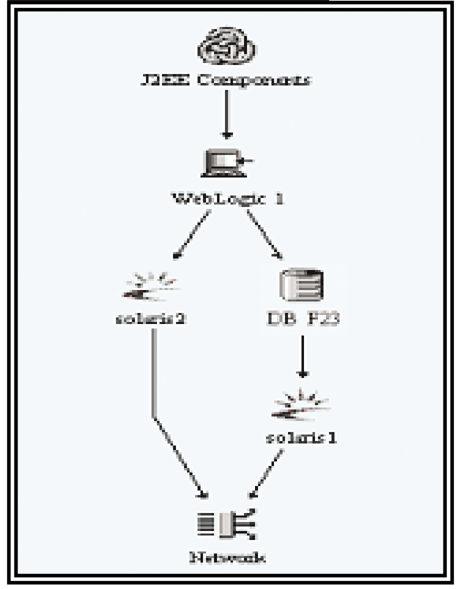
– *Quick time-to-value:* The solution needed to be implemented quickly and efficiently and enable high value with low maintenance.

FIGURE 1



Infrastructure to Business Process Modeling

FIGURE 2



Multitiered Infrastructure Model

- ***Managing applications and infrastructure to business objectives:*** In order to raise the bar and provide the highest possible service to customers and business partners, BEA was searching for an enabling management suite to coordinate the IT operations organization and focus it on the online customer services. The solution needed to provide proactive and detailed information about the key performance indicators of service quality and the customer experience. Leveraging this approach would enable BEA's operations staff to troubleshoot problems in real time, and optimize applications and infrastructure components in order to meet or exceed the high-quality standards they set.

- ***Operational collaboration across functional areas of responsibility:*** The management solution needed to provide a mechanism for providing accurate and timely information to a diverse group of users – database administrators, application support personnel, application developers, network and systems administrators, IT managers, and business managers. Each functional group was to use one integrated management solution in order to gain proactive insight into their own domain and across the entire infrastructure to enable operational collaboration and hence the highest possible service to partners and customers.

The PANACYA Solution

Note: The examples used in this document have been modified to protect the confidentiality and intellectual property of BEA and PANACYA.

PANACYA met BEA's requirements and operational business needs by implementing a solution that integrates Internet applications and associated infrastructure and business-centric service management into one cohesive management paradigm. The following details demonstrate three simple yet powerful steps the two teams used to deliver a solution enabled by PANACYA's bAWARE Application Performance Management Suite.

Step 1: Model IT infrastructure to business-centric service views

First, the PANACYA and BEA teams identified and organized application components and infrastructure into cohesive managed-service models. They used the PANACYA Management Suite's modeling capability to create visual models of all the objects that contribute to delivering online services. These models consisted of BEA's customer-facing applications and online services, including sales, customer support, developer center, and business partner services, as well as applications that support internal operational functions. The modeling enabled not only visual mapping of online services to applications and infrastructure but also detailed modeling of relationships and dependencies among application components and infrastructure components as they relate to business services and processes. Figure 1 depicts an example of these models.

PANACYA's Application Performance Management Suite enabled BEA administrators to define the relationships among application and infrastructure components with respect to online processes and services. Figure 1 shows a logic model of critical services as well as associated application and

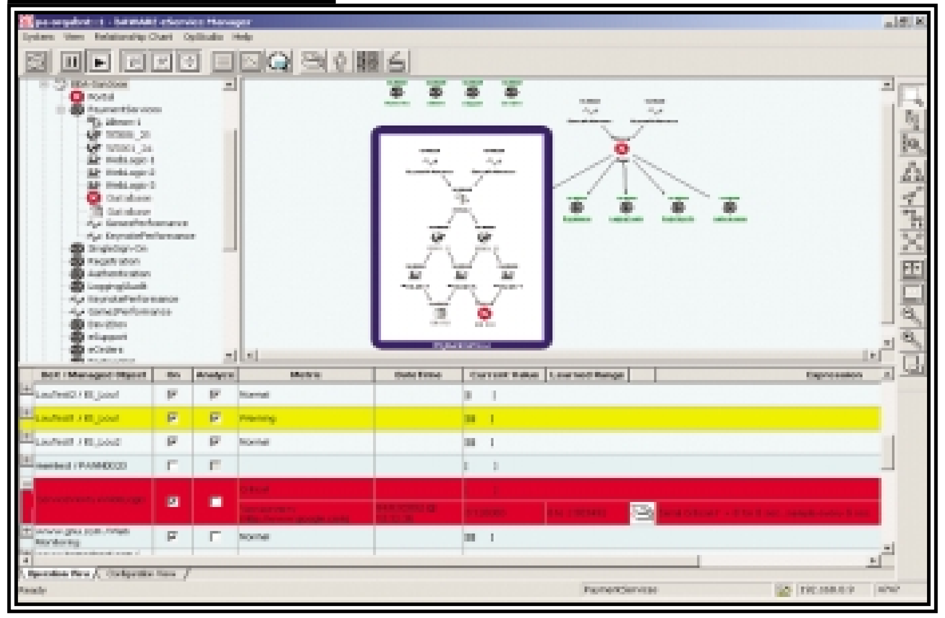
BEA

http://dev2dev.bea.com/useworkshop

infrastructure component relationships and dependencies. Models are created by simply dragging discovered objects from the object palette, located on the left side, onto the Modeling Canvas located on the right side (main view). By moving away from the more traditional physical topology type of visualization techniques used by other management solutions – which are brittle, require constant change, and do not associate infrastructure to business processes and services – the BEA team logically modeled its critical applications, processes, key performance indicators, and infrastructure as a starting point to create an operationally easy-to-understand and collaborative solution.

PANACYA is the first management vendor to incorporate semantic modeling as a core feature of its solution. There are many benefits of this model-based approach to application management, but BEA exemplifies perhaps the two most significant – proactive visibility into the complete service delivery infrastructure and operational collaboration. PANACYA's modeling capability enabled BEA's environment to be visually documented and published to users across many functional areas, including operations personnel responsible for managing databases, networks and systems, and application platforms, as well as application developers. The entire operational and development community benefited because they could easily visualize online services and all their component parts in one place and understand how components within the applications relate to one other, and more importantly, how they impact online services in real time.

FIGURE 4


Enterprise Manager with integrated service views and detailed root-cause drill-downs

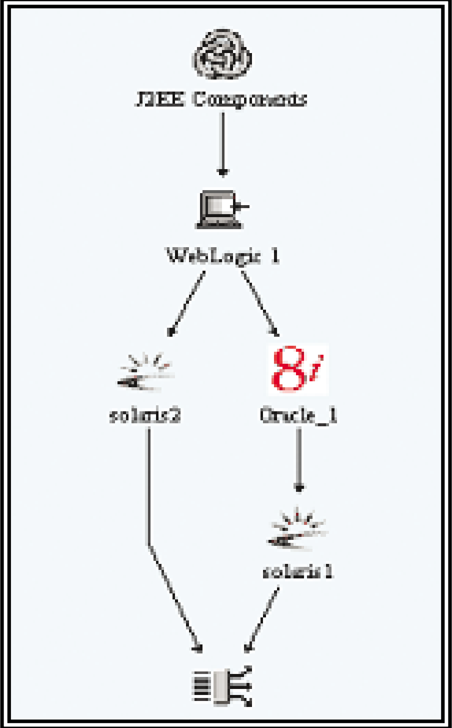
PANACYA's advanced modeling and visualization technology supports three types of models: logical infrastructure, business process, and workflow. These models can be easily associated with one another to provide true visualization and correlation among complex applications, infrastructure, and the business processes they support. A model can document a simple relationship, such as an application server's dependency on its underlying operating system or database server, or instead define a more complex relationship among supporting applications in sequential steps of a business process. Building relationships is a simple drag-and-drop operation via the suite's modeling interface.

Beyond the stated benefits, PANACYA's modeling and visualization technology is greatly enhanced by applying advanced peer-to-peer intelligence to it. PANACYA has pioneered distributed intelligence and advances the state of the art in management technology by applying prepackaged intelligence modules (Intelligent Behavior Experts – BeXs) to these customer-defined models. Instead of using fixed thresholds to detect faults and performance problems, an approach common to most management solutions, PANACYA leverages patent-pending BeX technology, which automatically adapts to the environment and the varying load conditions consistent with Web architectures. BeXs enable performance, availability, and resource utilization to be proactively monitored across the boundaries of infrastructure components – end-to-end and from the customer experience through the application components and platforms to the back-end databases.

Using PANACYA technology, intelligent models can be easily created that logically define infrastructure relationships and dependencies like the model depicted in Figure 2 or with a process orientation such as the model in Figure 3. These models can then be associated with each other to relate infrastructure relationships and performance behaviors to critical processes. In either scenario, PANACYA's modeling capability brings business context to application and infrastructure management, while its distributed peer-to-peer intelligence enables proactive performance and fault detection and root-cause analysis.

This advanced capability enables users, for the first time, to visualize critical services in a way that is most understandable to their job functions, while gaining insight into how business processes, applications, and infrastructure work together holistically to provide online services. PANACYA's model-driven visualization technology not only makes it possible to understand complex application environments but also helps to address and mitigate the finger-pointing that is pervasive in operations today. By receiving simple-to-understand, holistic views of services

FIGURE 3



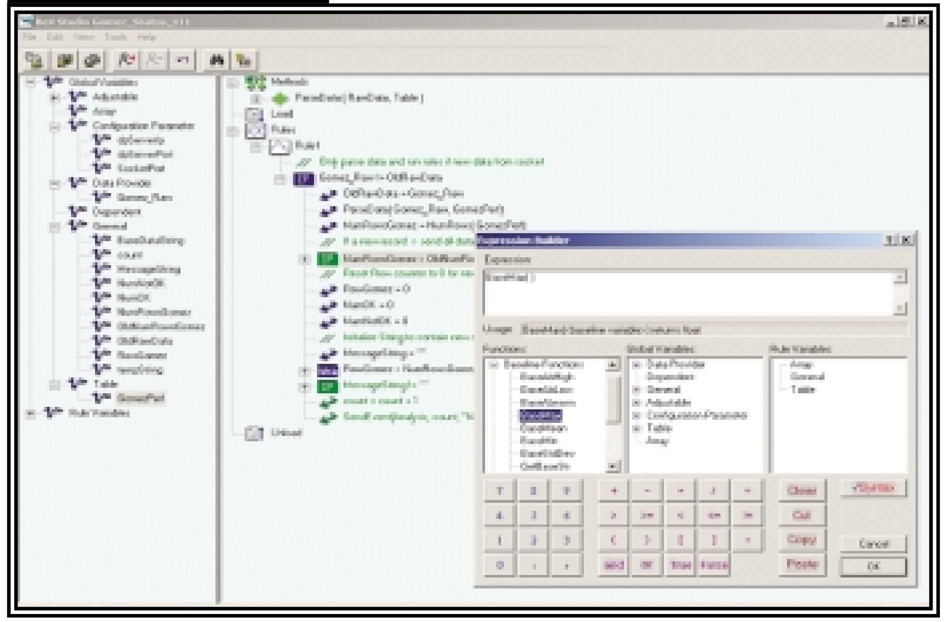
A simple process model.

Sitraka

www.sitraka.com/performance/wldj

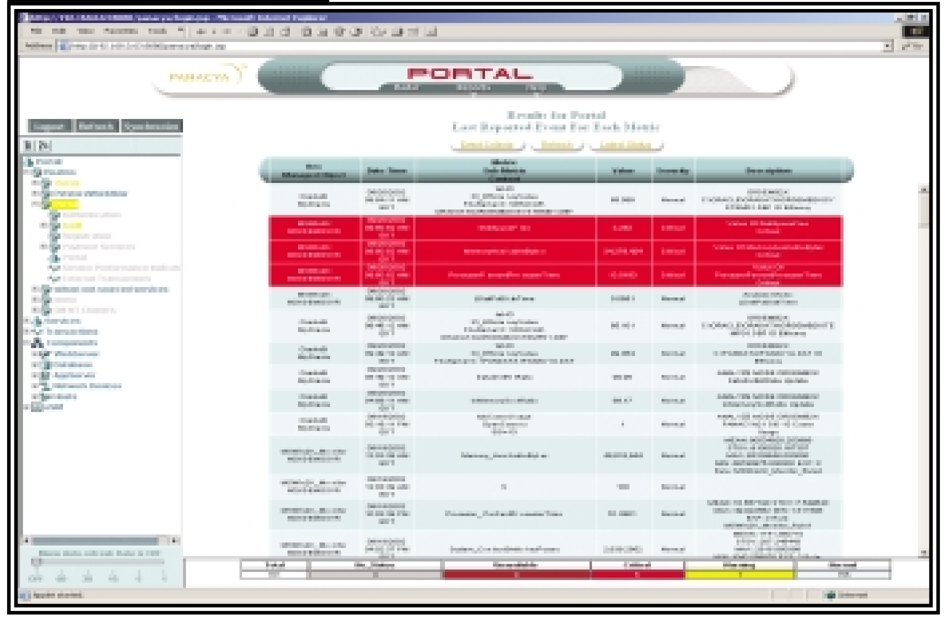
and their component parts, operations personnel can focus immediately on the pressing issues and accurately pinpoint and correct problems that may impact services, instead of the more common practice of managing systems and devices independently, of the state of business services. This service modeling and visualization approach enables BEA's system, network, database, and application support staff to collaborate in order to isolate and fix problems much faster than previously possible.

FIGURE 5



BeX Studio – Business rules and advanced correlation studio

FIGURE 6



PANACYA's Web-based Event Console for real-time and historical reporting

Step 2: Deploy distributed peer-to-peer out-of-the-box intelligence to proactively monitor applications and infrastructure:

After organizing the infrastructure into cohesive service-delivery models, the BEA team deployed PANACYA's out-of-the-box BeXs to the previously modeled services. PANACYA has created BeXs for the most popular Web servers, application servers, databases, network devices, and systems. Highly intelligent modules, BeXs automatically self-adapt to the environment and learn the desired behaviors of application platforms and infrastructure components without configuration. Then, instead of bombarding operators with thousands of events and reports to centralized consoles, PANACYA's BeXs focus on proactively alerting operators to only abnormal events, based on the learned normal behavior that could impact components and services. BeXs are highly deterministic and often detect emerging problems and trends long before they become faults and affect services.

BEA deployed out-of-the-box BeXs to manage BEA WebLogic Server, custom J2EE application components, enterprise databases, Cisco routers and switches, Solaris and Dell servers, and user transactions that traverse the infrastructure stack. By leveraging the information provided by the service models regarding component relationships and dependencies, and by leveraging the peer-to-peer distributed architecture, BeXs work together in an intelligence network to detect and pinpoint problems across the entire service-delivery stack.

When BeXs detect emerging problems, they display what they have learned through the service models so that operators always consider performance anomalies and faults in the context of business process models and services.

Figure 4 depicts BeXs reporting a potential service-threatening performance problem and its impact to the service model. We also see how BeXs provide detailed drill-downs into the root cause of the problem. Everything IT operations needs to proactively monitor applications and services is provided in one integrated and intelligent management console – allowing it to detect and fix problems across the infrastructure stack in real-time.

In addition to gaining proactive control of the application and infrastructure environment, PANACYA provides a unique solution to define and manage the environment according to business rules and policy via BeX Studio. Depicted in Figure 5, BeX Studio is an advanced visual analytical tool that enables both operations personnel and analysts to encapsulate and automate operational procedures, service level agreements, and advanced business policy rules that establish and define best practices in operational automation.

As an example, BEA is using BeX Studio to automate the detection of abnormal network behaviors; correlate abnormal contention between BEA WebLogic and many of its enterprise databases; and encapsulate, automate, and scale a multitude of operational and institutional best practices and knowledge. Instead of creating custom scripts that become difficult to maintain, BeX Studio uses "scriptless" intelligent objects that can be easily deployed, replicated, and managed. It provides a vehicle to scale operational expertise and automation and dramatically enhances operational efficiencies while negating the need for many subject matter experts. Additionally, BeX Studio is tightly integrated with PANACYA's bAWARE Suite and leverages the advanced analytical capabilities of PANACYA's intelligent distributed peer-to-peer architecture.

Step 3: Provide policy-driven service views of application and infrastructure performance to operations and business staff members

Empowering BEA's operations staff with integrated service views of the entire service-delivery stack enabled the business-focused proactive control and operational collaboration necessary to achieve "operational excellence." The final step in the project was to create roles, responsibilities, and custom views for operations staff and managers responsible for managing the online services. PANACYA's drag-and-drop policy modeling capability eased this process at BEA.

The most difficult part of the process was determining who had access to PANACYA's Management Suite and their associated roles and responsibilities in the context of the specified services and/or components of the managed services. Because of PANACYA's integrated and object-oriented design, defining and deploying operational policy was a simple matter. Roles were first created for IT managers, administrators, operators, and business managers. Attributes for these roles were assigned according to job function and responsibilities. Then, using a drag-and-drop metaphor, the policy objects were applied to the many service models. Instead of using laborious methods for defining and implementing alerts, notification, and usage rights policies, PANACYA's approach enabled BEA operational policy to be defined and deployed in less than a day.

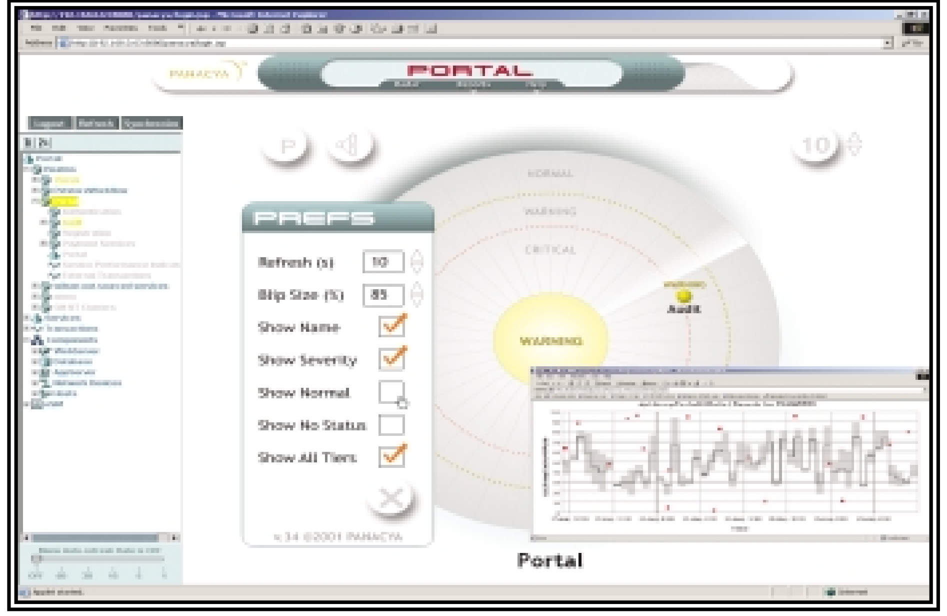
PANACYA's bAWARE Application Performance Management Suite is a complete and highly integrated solution. It not only enabled BEA to quickly deploy the solution across all Web services but also provided the BEA operations staff with integrated views of the entire service-delivery stack.

PANACYA provided BEA with many different views into their application infrastructure to accommodate the diverse needs of the user community. Pictured in Figure 6 is PANACYA's Web-based Event Console. This enables users to pick and choose services and components from an object palette and view performance information in either real-time or historically across the entire infrastructure stack. Events are categorized by severity and color-coded for easy viewing and to reflect state. Because PANACYA's underlying monitoring system is highly intelligent, the views are generally "quiet" and reflect only abnormal performance conditions unless the user requests all information (normal and abnormal behaviors) from the intelligent monitoring BeX technology. Additionally, PANACYA's BeXs store the learned normal behaviors of all metrics across all components by hour, day, week, and month (dynamic baselining). Operators use this information both to proactively detect abnormal performance behaviors in real time and to perform capacity-planning tasks.

PANACYA's eService Management Portal also supports Radar Service Views. Shown in Figure 7, the Radar View displays services and all their associated infrastructure components in this single view. If service levels are violated or if any component exhibits abnormal performance, it is automatically displayed in the inner bands of the Radar. The closer to the center an object appears, the more adverse an impact it has on the managed service. Additionally, operators can drill down into the details and the root cause by simply clicking on the object. The drill-down in Figure 7 shows normal behavior in gray and the actual component behaviors in red.

All of the performance management information in PANACYA's Management Suite is stored and displayed this way in real time and historically. PANACYA provides many different views and tools to aid in proactively monitoring and resolving problems. The accuracy of the information, the correlation of events across applications and infrastructure, and the root-cause analysis provided would not be possible without their unique, distributed peer-to-peer intelligent architecture.

FIGURE 7



PREFS

Refresh (s) 10
Blip Size (%) 85
Show Name ✓
Show Severity ✓
Show Normal
Show No Status
Show All Tiers ✓

PANACYA's eService Management Portal – with detailed drill-down

The Outcome: Operational Excellence

PANACYA has provided BEA with a significant capability and enabling technology to achieve operational excellence in order to provide the highest possible service levels to customers and business partners. This includes the ability to proactively view and control distributed and mission-critical applications, and correlate applications and infrastructure to business processes and services in real time. This may seem like an obvious outcome, but it's difficult to find this kind of integrated and proactive management solution deployed across an infrastructure and "operationalized," even in the most successful and progressive of companies. PANACYA endeavors to seek out other leaders in the industry who want to challenge current thinking and conventional wisdom and who, like BEA, aspire to achieve operational excellence.

Rich-Client Deployment in a ZAC-less World

LEADING DENTAL ADMINISTRATOR MOVES BEYOND ZAC WITH SITRAKA DEPLOYDIRECTOR

BY
DAVE TRUMAN

AUTHOR BIOS...

Dave Truman is a technology writer at Sitraka. He has more than 10 years of experience working with and writing about application development tools, and has worked with Java since JDK 1.0.

CONTACT...

dave.truman@sitraka.com

Organizations deploying rich client/server WebLogic applications need to fill the void created by the deprecation of BEA Zero Administration Client (ZAC). The affiliated Delta Dental Plans of Michigan, Ohio and Indiana, a leading dental plan administrator, chose Sitraka DeployDirector to deploy the client-side portion of their claims-processing application to thier users.

Some days it seems like you can't win. You've developed a WebLogic application that has a Swing-based "rich client" front end and were planning to deploy it using BEA's ZAC (Zero Administration Client) deployment utility. Then you learn that ZAC has been deprecated. What do you do? One company solved this problem with Sitraka DeployDirector, a third-party Java provisioning and management solution.

The affiliated Delta Dental Plans of Michigan, Ohio and Indiana (Delta Dental) run the largest dental benefits system in the Midwestern United States, providing dental health benefits to large and small employers. It is part of the Delta Dental Plans Association, a nationwide group of 37 independent affiliated dental service plan corporations. The tri-state operation provides coverage to more than 6 million people enrolled in over 4,200 groups.

Processing dental claims for an organization this large and geographically dispersed is a big job. The varying levels of network bandwidth and infrastructure between affiliated offices and Delta Dental's many claims processing agents was one challenge. The most unexpected obstacle, however, turned out to be around the *deployment* of the rich-client component of the application.

A WebLogic Application with a Twist – a Rich-Client Front End

Delta Dental designed a custom claims processing system to work well in this environment. A cluster of BEA WebLogic application servers running on Unix formed the core of the system, handling transaction processing and business logic. Not exactly an atypical J2EE application architecture – except for the front end.

While most WebLogic application developers initially opt for a browser-based presentation layer (or front end), Delta Dental determined that they needed a rich-client front end. Claims agents enter a lot of data and needed the speed and productivity a rich client would provide. A browser-based thin-client presentation layer can have usability and performance limitations compared to a Swing-based user interface. In this case, the team determined that it would require too many round-trips to the server to repaint screens as users changed fields in the complex user interface.

A hybrid Java Swing GUI application on the client that connects to WebLogic to process business logic and transactions is a bit of a twist on the standard J2EE application stack, but one that worked well to address the application's performance and usability needs.

The Deployment Challenge

Of course, one big reason most WebLogic applications employ browser-based user interfaces is the hassle of deploying Java-based rich clients. Rolling out a Java-based application

Yahoo

www.enterprise.yahoo.com/testdrive

across a large number of end-user machines in an organization is particularly challenging because a Java Runtime Environment (JRE) must also be present on each machine, and must be correctly configured. Using Java applets does not solve this problem entirely, and applets have major drawbacks of their own.

WebLogic applications that employ a browser-based presentation layer rely on the server to render pages to the user. Deployment is easy in this case, but applications with complex user interfaces require a high number of round, trips to the server, which impacts performance and usability.

Delta Dental had planned to use a feature in WebLogic called ZAC. But while ZAC was provided with earlier releases of WebLogic, BEA deprecated the feature in 6.0, and does not provide documentation for it in 7.0.

Therefore, deploying and managing the rich-client portion of the application to the many remote users became more of a challenge than originally anticipated. While the team could have chosen to use ZAC anyway, they felt it was not a good long-term decision, especially after learning that ZAC requires extra development work – work that would not migrate to a new deployment system down the road. Now was definitely the time to find a new solution.

Choosing a New Deployment Solution

Delta Dental turned to Amit Singh, a systems engineer with BEA's Professional Services Organization, for help. Mr. Singh had firsthand knowledge of the real challenges of deploying large Java client applications to a wide user base. Having used ZAC for past projects, Mr. Singh knew the effort required for that solution.

BEA's documentation for WebLogic 7.0 recommends migrating from ZAC to Sun's Java Web Start deployment system. However, Mr. Singh knew this solution wouldn't work for Delta Dental. While Java Web Start provides basic remote deployment and dynamic updating capabilities, it is not suitable for enterprise-wide application deployment and management. Specifically, Delta Dental required:

- A deployment solution designed for clustered application servers communicating with many clients
- Automated rollout and rollback of application updates
- Remote monitoring and administration capabilities to help IT staff troubleshoot problems with users

Delta Dental needed a deployment solution that would provide everything ZAC did and could meet the requirements. The team did some research and discovered Sitraka DeployDirector.

Sitraka DeployDirector is a Java application provisioning and management solution that helps IT organizations deploy, manage, and update rich clients. DeployDirector makes it easy to deploy, manage, and update Java rich clients across multiple client platforms from the WebLogic application server.

It provides WebLogic developers with control over application provisioning, updates, rollbacks, and version usage. In addition, DeployDirector provides powerful logging and reporting features, access control, JRE management, and immediate wireless and e-mail error alerts.

DeployDirector is essentially an automated deployment platform. It's server-side component stores the rich client application bundles and JREs, and manages the versions of the rich-client application. It also has a client-side component that wraps around the rich-client application, checking whether it needs to be updated, that it has the correct JRE, and so on. The team chose to evaluate DeployDirector to verify that it would meet their requirements and would integrate seamlessly into their WebLogic environment.

DeployDirector and WebLogic – Making It Work

The back end of Delta Dental's claims-processing system runs on a WebLogic cluster comprising two WebLogic servers and a load balancer. The key task was to integrate DeployDirector into the WebLogic servers because once integrated, a developer can access DeployDirector's tools to set up deployment of their own rich-client application.

The DeployDirector installer by default sets up and configures itself to use its bundled copy of the Apache Tomcat server. However, the distribution also provides a WAR file to integrate its server-side components with an enterprise-class application server such as WebLogic.

Mr. Singh and the Delta Dental team added the WAR file to each WebLogic server and configured the servers to run the DeployDirector server-side components. The team could also have chosen to run deployment centrally. Once configured, the team tested that DeployDirector was running in WebLogic. They then deployed the administration tools to a development workstation.

Setting up deployment of the client-side portion of the application was a point-and-click affair. The DeployDirector administration tool is a GUI-based application that creates and manages application deployment bundles. The tool enabled Delta Dental to create a deployment package for the client-side component of their application and specify which JRE it should run with, when to check for updates, and so on.

Delta Dental found that DeployDirector was not only able to replace ZAC, it was a better and more robust deployment solution than ZAC. "Everyone was very happy with how easily you can deploy and update the application on the fly with DeployDirector," says Mr. Singh. "Real-world use of ZAC requires a lot of custom coding, but DeployDirector didn't – Delta Dental was up and deploying in less than a week."

No ZAC? No Worries

After performing test deployments of the client-side application to pilot groups of users, Delta Dental was satisfied. Says Mr. Singh, "Delta Dental was very happy that the architecture they selected could be enabled with DeployDirector." Delta Dental subsequently realized that the DeployDirector management platform provides them with far greater control and flexibility than would have been possible with ZAC. Some key benefits include:

- Designed to deploy large applications over clustered environments to many users in a variety of locations.
- Applications can be set up for deployment easily within a GUI environment, without requiring application code changes or extensive scripting.
- Rapid updating and rollback capabilities, using class-level differencing to minimize bandwidth requirements.
- Easily monitors and reports on who is using what version of applications, including wireless and e-mail alerts when a rich client has runtime problems.

For More Information

For more information on the affiliated Delta Dental Plans of Michigan, Ohio and Indiana, please see www.deltadentalmi.com. For more information on DeployDirector see www.sitraka.com/deploydirector.

Hewlett Packard

www.hp.com/go/developers

Part 2

BUILDING BETTER BRIDGES

ADVANCED J2EE/PEOPLESOFT INTEGRATION USING WEB SERVICES

BY
JOSEPH K. KROZAK

AUTHOR BIO...

Joseph K. Krozak is vice president of Technology Development for Krozak Information Technologies, Inc. (www.krozak.com), a supplier of advanced software solutions to Fortune 500 and midmarket companies. He has over 15 years of experience in the information technology industry, including 10 years of extensive experience building large, distributed object- and component-oriented systems.

CONTACT...

jkkrozak@krozak.com

The most challenging integration efforts frequently involve integrating enterprise resource planning (ERP) or customer relationship management (CRM) systems with new and existing custom applications.

These enterprise software applications often have proprietary architectures and complex APIs, and use proprietary programming languages unfamiliar to the mainstream developer community. However, some enterprise software vendors are attempting to make the integration effort more approachable. Using XML and Web services, these vendors are attempting to bridge the gaps between their respective products and the world around them.

In the first installment of this two-part series (***WLDJ***, Vol. 1, issue 10), I looked at how the Java Messaging Service (JMS) and PeopleSoft 8.4's Integration Broker (IB) technology can be used to integrate PeopleSoft and J2EE applications. In this article, I will explore how Web service technology can be used to integrate PeopleSoft with J2EE applications. These concepts can be extrapolated to integrate PeopleSoft with any application capable of manipulating XML content and maintaining an HTTP connection. This opens the doors of integration to a wide range of environments such as J2EE and .NET.

Web Services – A Matter of Definition

For PeopleSoft Inc., the definition of a Web service is clear – it is an application component that can be accessed using XML messages over an HTTP connection. This definition means that Web services are inherently open and accessible over the Internet and across corporate intranets. In addition, they are accessible to a wide range of technologies and platforms.

XML and HTTP are the essential ingredients of any Web service. Additional standards, such as Simple Object Access Protocol (SOAP) and Web Services Description Language (WSDL), improve the consistency and accessibility of Web services by standardizing the description of Web services and the format of the messages they exchange. However, SOAP, WSDL, and Universal Description, Discovery, and Integration (UDDI) – a protocol for cataloging, publishing, and locating available Web services – are not requirements for Web services. Instead, these protocols mainly enhance the overall usability of Web services.

I will describe two scenarios for integrating PeopleSoft and J2EE applications. The first describes how a PeopleSoft application can invoke a WebLogic Server (WLS) 7.0–hosted Web service. The second describes how a J2EE application can invoke a PeopleSoft Web service. Both scenarios make ample use of XML and HTTP, while only the first uses SOAP, and neither explicitly uses WSDL. For the sake of simplicity, neither will use UDDI. Rather, it is assumed that both the PeopleSoft and J2EE clients know the specific names and locations of the Web services they desire.

Example 1: Invoking a J2EE Web Service from a PeopleSoft Client

In this scenario, I used a very simple Web service that is implemented as a session bean. WebLogic's servicegen Ant task automates the process of turning Java classes, session beans, and message-driven beans into Web services. Hence, the work involved in providing and publishing the Web service is considered trivial and thus, in the interest of time, will not be described. Instead, I'll first focus on the IB definitions that are required to support the integration. Next, I'll explore the actual client-side PeopleCode required to invoke the Web service.

As described in my first article, a number of key IB definitions must be created in order to support an integration scenario. In this scenario, PeopleSoft is synchronously invoking an HTTP-based Web service that resides on an external system. Since a synchronous invocation implies both a request and associated reply, I'll first define the request and reply messages, JKK_SBINFOFACADE_SOAP_REQ and JKK_SBINFOFACADE_SOAP_RESP, using the PeopleTools Application Designer environment. Next, an IB node must be defined to represent the external system, and its associated target connector must be PeopleSoft's HTTPTargetConnector. This node, JKK_WS_TARGETNODE, and its associated HTTPTargetConnector properties are shown in Figure 1.

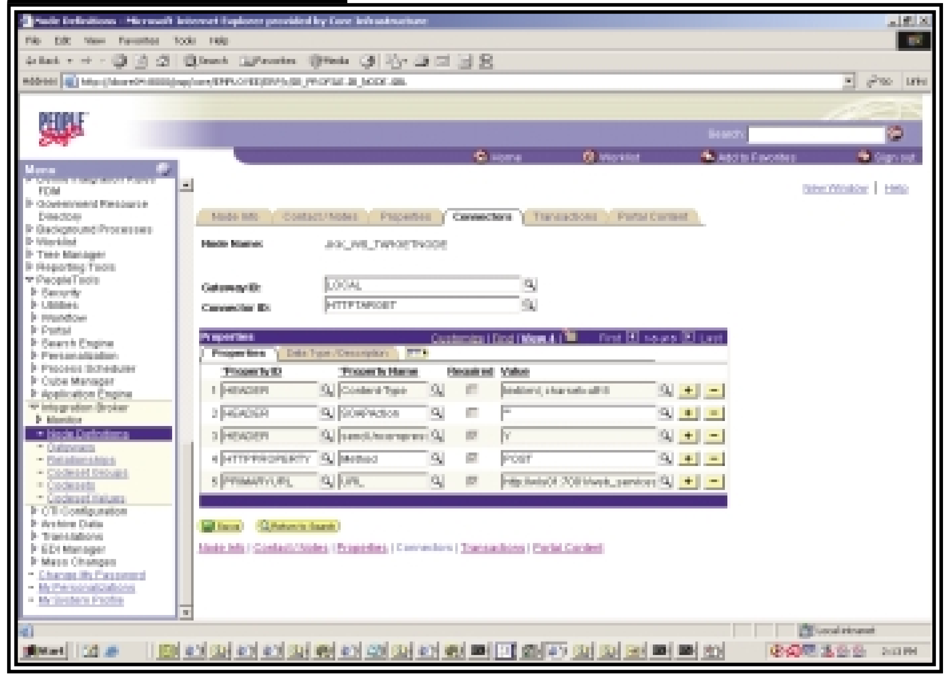
The HTTPTargetConnector states that an HTTP POST to the URL, http://wls-01:7001/web_services/SBInformationFacade, is needed to reach the JKK_WS_TARGETNODE IB node. This URL is the address of the WLS-hosted Web service. Next, an outbound synchronous (outSync) transaction must be defined on the JKK_WS_TARGETNODE node, specifying JKK_SBINFOFACADE_SOAP_REQ and JKK_SBINFOFACADE_SOAP_RESP as the request and reply messages, respectively. Collectively, these IB definitions will support the synchronous exchange of messages with the WLS-hosted Web service, using an HTTP connection.

Next, I'll address the specific tasks required to create the PeopleCode Web service client. We begin by using the deployed Web service's home page (located at the URL listed above) to examine the SOAP messages exchanged when a Web service operation is invoked. For this example, I'll focus on invoking the getID Web service method, which simply generates and returns a unique identifier to its caller. Figure 2 illustrates the results of invoking getID() using the Web service home page.

This information can be used to implement the PeopleSoft client. Since PeopleSoft has no JAX-RPC equivalent, the client cannot readily make use of a Web service's WSDL description. Hence, in order to invoke the getID() operation, the PeopleSoft client must explicitly create, send, receive, and interpret the SOAP messages shown in Figure 2.

PeopleSoft provides direct support for manipulating SOAP messages and XML documents, via the SOAPDoc and XMLDoc PeopleCode classes. To invoke the getID() operation, the client must create and then

FIGURE 1



Node Name: JKK_WS_TARGETNODE

Gateway ID: LOCAL
Connector ID: HTTPTARGET

Properties

	Property ID	Property Name	Required	Value
1	HEADER	Content-Type		text/xml; charset=utf-8
2	HEADER	SOAPAction		""
3	HEADER	sendUncompressed		Y
4	HTTPPROPERTY	Method		POST
5	PRIMARYURL	URL		http://wls-01:7001/web_service

Web service IB node with HTTPTargetConnector send an appropriate SOAP request document to the Web service. These steps are illustrated in the PeopleCode program in Listing 1.

The special PeopleCode functions, CreateSOAPDoc() and ValidateSOAPDoc(), are used to create and validate SOAP documents. Both methods operate on instances of SOAPDoc, a special PeopleCode class that encapsulates SOAP documents.

SOAPDoc allows clients to specify and retrieve the fundamental elements of a SOAP document, including its envelope, body, method, and associated parameters. Upon inspecting the SOAP request and response documents in Figure 2, we learn that the actual SOAP method for invoking getID() is, in fact, getID(). Hence, after adding the mandatory envelope and optional body to the SOAP request document using SOAPDoc's AddEnvelope and AddBody methods, I used AddMethod to specify "getID" as the target Web service operation. Since SOAP methods can represent a request or response, AddMethod's second calling parameter can be used to specify which role to use. Supplying "1" means the method is a request, which will ultimately generate a getID element in the SOAP document (ex: <getID></getID>). However, supplying a "0" means the method is a response, which will be rendered in the SOAP document using the generally accepted practice of appending "Response" to the name of the method, eg., <getIDResponse></getIDResponse>. If getID() took arguments, they could be specified using SOAPDoc's addParm(parmName, parmValue) method, in which parmName is the formal name of the parameter or "part" (as specified in the Web service's WSDL description), and parmValue is its value.

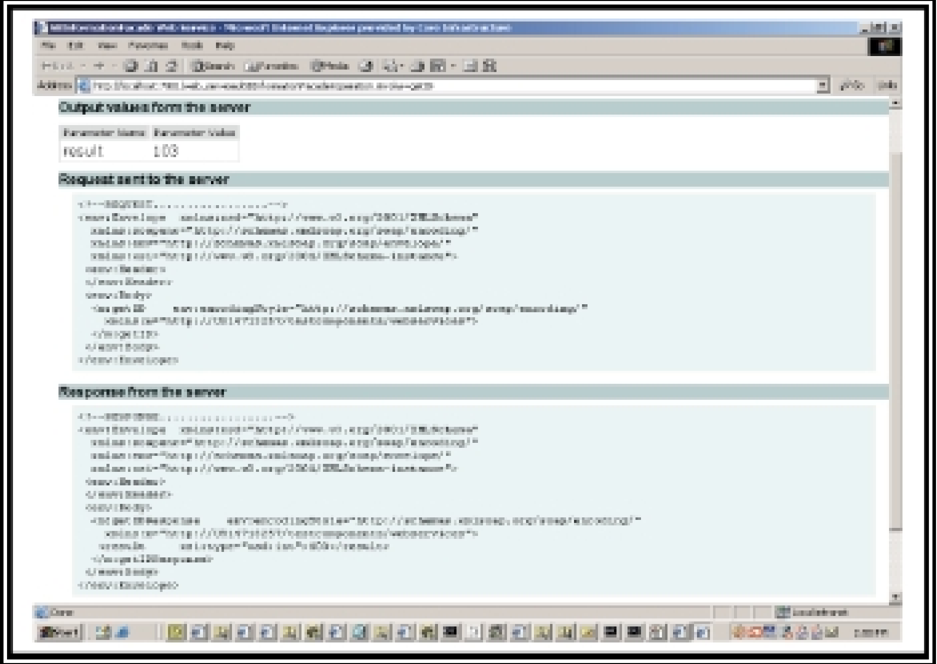
The next step is to send the SOAP request document to the Web service using either the synchronous SyncRequestXmlDoc IB function or its asynchronous counterpart, PublishXmlDoc. Both functions will send a given XML document, under a specified message name, to a specified IB node. In Listing 1, the SOAPDoc's XMLDoc component is first extracted before being sent as a JKK_SBINFO-FACADE_SOAP_REQ message to the JKK_WS_TARGETNODE IB node. In turn, IB will send the SOAP request document to the specified URL, effectively invoking the Web service that resides there. Once the Web service getID() operation completes, IB will convey the results to the PeopleCode client.

Finally, the PeopleCode client will use a SOAPDoc instance to capture and interpret the Web service's response. Because a response can contain multiple SOAP parameters (i.e., for the out and in-out parameters of a Web service operation), the SOAPDoc class provides GetParmName and GetParmValue methods to determine the name and value of specific parameters. In our scenario the getID() operation (and getIDResponse SOAP method) only returns one parameter, an integer named result (see Figure 2). So, for our purposes, we can assume that the first parameter name and value correspond to the result and associated value returned by the getID() operation.

Example 2: Invoking a PeopleSoft Web Service from a J2EE Client

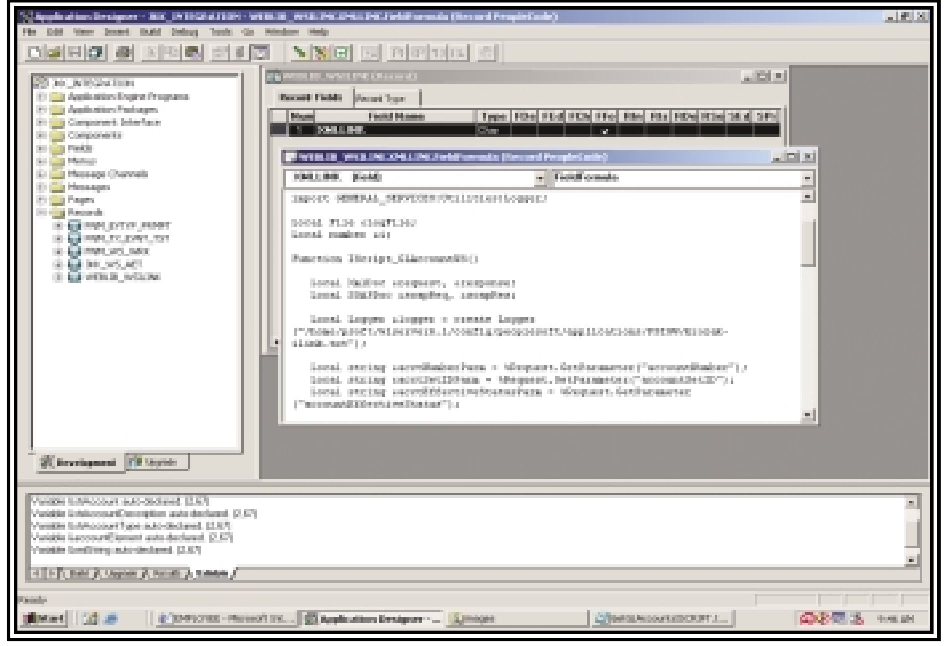
PeopleSoft functionality can be exposed as Web services using an Inbound Business

FIGURE 2



SOAP request and response messages for getID() operation

FIGURE 3



WEBLIB definition

Borland

www.borland.com

Interlink. Business Interlinks are integration mechanisms that enable external systems to perform component-based, real-time integration with PeopleSoft systems. Business Interlinks accomplish this by creating synchronous transactions that facilitate the real-time exchange of data between PeopleSoft applications and external systems.

Inbound Business Interlinks provide Internet Scripts (or IScripts), which are specialized PeopleCode programs that behave much like J2EE servlets. An IScript resides on the PeopleSoft server and is capable of exchanging XML data with external systems using HTTP. Incoming HTTP requests can originate from a Web browser or a client program communicating via an HTTP connection. An IScript definition spans both the PeopleSoft application and Web servers. The application server portion, which must be defined first, contains the actual implementation or business logic of the IScript, and is written in PeopleCode. The first step is to use the PeopleTools Application Designer to create a record, called a Web Library, or WEBLIB. WEBLIBs, which are roughly analogous to Java Archive (JAR) files, can store collections of IScript definitions within the Field Formula (FFO) fields of their columns. These records and columns, although similar to the definitions discussed in the first article, are not necessarily intended to store application data. Rather, they simply provide a convenient place to house and organize IScripts. Figure 3 illustrates the WEBLIB record, WEBLIB_WSILINK, which is used in our example.

The Web server portion, defined via the PIA screens, links the IScript to a service name, and stores this linkage in an XML registry. The service name will become part of the URL that any HTTP client (including a Web browser) must use in order to access the IScript. The definition and linkage of this service name is shown in Figure 4.

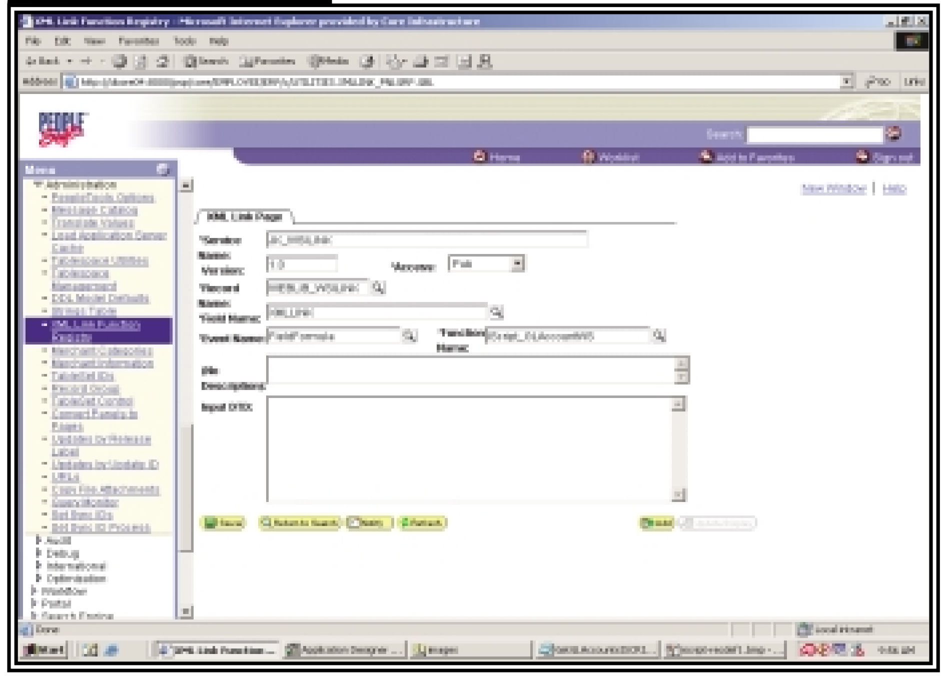
PeopleSoft's rich security model is at play here – ensuring that only properly authorized clients can access an IScript and the business components it uses. You can use the PeopleTools Security PIA screens to enable client access to an IScript and its WEBLIB, as shown in Figure 5.

Once the appropriate security settings are made, the URL can be used from a browser or client program to invoke the IScript. The IScript can be written to accept and return information in XML form, effectively making it a Web service.

An IScript, like a J2EE servlet, has programmatic access to the incoming HTTP request and outgoing HTTP response. As such, an IScript can read and use the contents of the incoming request object to guide its operation. For example, an IScript could take one or more arguments. These arguments could be passed through the IScript's URL, making them available via the request object. The IScript can respond, much like a J2EE servlet, by populating the supplied response object, which is ultimately conveyed back to the HTTP client. For our example, I'll create a simple IScript-based Web service that returns a list of known general ledger accounts, given some selection criteria. Listing 2 shows the PeopleCode implementation of the IScript, which ultimately creates and returns to the HTTP client an XML document containing the desired accounts.

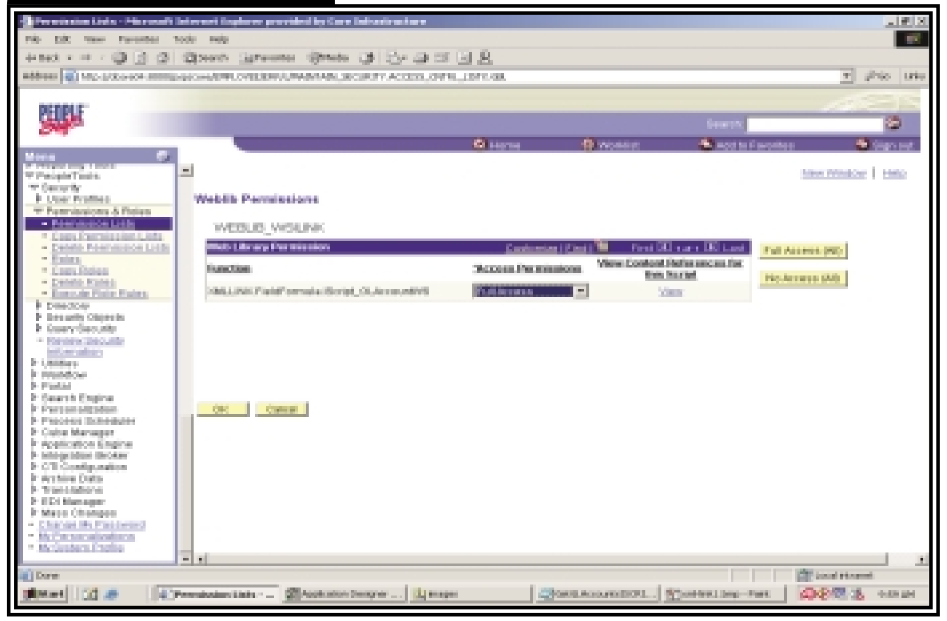
As shown in Listing 2, the special PeopleCode variables, %Request and %Response provide access to the HTTP request and response objects, respectively. Additional PeopleCode classes, like SQL, are used to query the account data, while XMLDoc and XMLNode classes are used to build the XML document containing the account data. Ultimately, this Web service can be tested from within a Web browser by simply jumping to its URL. Figure 6 shows the result of invoking the Web service from within a browser. This same data would be returned to a programmatic HTTP client.

FIGURE 4



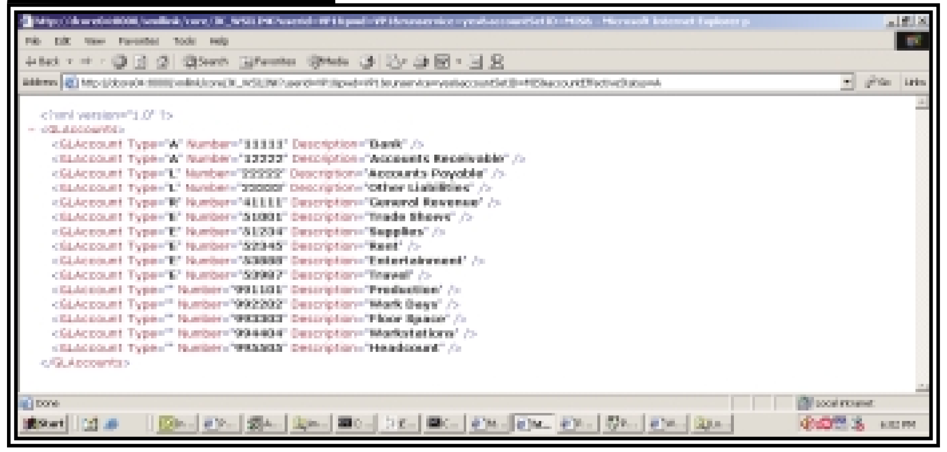
XML link registry entry for IScript

FIGURE 5



Security authorization for IScript use

FIGURE 6



```
<?xml version="1.0" ?>
- <GLAccounts>
    <GLAccount Type="A" Number="11111" Description="Bank" />
    <GLAccount Type="A" Number="12222" Description="Accounts Receivable" />
    <GLAccount Type="L" Number="22222" Description="Accounts Payable" />
    <GLAccount Type="L" Number="23000" Description="Other Liabilities" />
    <GLAccount Type="R" Number="41111" Description="General Revenue" />
    <GLAccount Type="E" Number="51001" Description="Trade Shows" />
    <GLAccount Type="E" Number="51234" Description="Supplies" />
    <GLAccount Type="E" Number="52345" Description="Rent" />
    <GLAccount Type="E" Number="53888" Description="Entertainment" />
    <GLAccount Type="E" Number="53987" Description="Travel" />
    <GLAccount Type="" Number="991101" Description="Production" />
    <GLAccount Type="" Number="992202" Description="Work Days" />
    <GLAccount Type="" Number="993303" Description="Floor Space" />
    <GLAccount Type="" Number="994404" Description="Workstations" />
    <GLAccount Type="" Number="995505" Description="Headcount" />
  </GLAccounts>
```

Testing the GL Account Web service from a browser

It is a fairly straightforward matter to automate the invocation of the Web service using an HTTP-enabled J2EE client. Listing 3 shows the Java implementation of a simple client, which first uses a URL to address the targeted Web service. Next, the client program performs a simple HTTP GET operation in order to receive the response. Finally, a mixture of JAXP and DOM invocations are used to parse and process the general ledger accounts returned from the Web service.

Conclusion

These articles only scratched the surface of integrating PeopleSoft with external systems. Issues such as security, scalability, and integration using various other protocols are complex and require far too much time to cover in this medium. However, these articles do shed light on the incredible advancements PeopleSoft has made to this critical aspect of systems integration.

Listing 1

```
...
/*Create and Send SOAP request document. */
Local SOAPDoc &soapReq = CreateSOAPDoc():

&soapReq.AddEnvelope(0);
&soapReq.AddBody();
&soapReq.AddMethod("getID", 1);

Local number &ok = &soapReq.ValidateSOAPDoc();
Local XmlDoc &request = &soapReq.XmlDoc;

Local XmlDoc &response = SyncRequestXmlDoc(&request, Message.JKK_SBINFOFA
CADE_SOAP_REQ, Node.JKK_WS_TARGETNODE);

/*Receive and Interpret SOAP response document. */

Local SOAPDoc &newSOAPDoc = CreateSOAPDoc();
&newSOAPDoc.XmlDoc = &response;
Local string &parmName = &newSOAPDoc.GetParmName(1);
Local string &parmValue = &newSOAPDoc.GetParmValue(1);

Return &parmValue;
...
```

Listing 2

```
Function |Script_GLAccountWS()
   Local string &acctNumberParm =
      %Request.GetParameter(“accountNumber”;
   Local string &sqlString = "SELECT ACCOUNT, DESCR, ACCOUNT_TYPE FROM
    PS_GL_ACCOUNT_TBL WHERE EFFDT <= SYSDATE";

   If &acctNumberParm <> "" Then
      &sqlString = &sqlString | " AND ACCOUNT = '" | &acctNumberParm |
       "'";
   End-If;

   If &acctSetIDParm <> "" Then
      &sqlString = &sqlString | " AND SETID = '" | &acctSetIDParm | "'";
   End-If;

   Local SQL &getGLAccountsSQL = CreateSQL(&sqlString);
   Local XMLDoc &response = CreateXmlDoc(“straight”<?xml
     version='1.0'?><GLAccounts></GLAccounts>”straight”);
   Local XMLNode &anXmlNode = &response.DocumentElement;

   While (&getGLAccountsSQL.Fetch(&strAccount, &strAccountDescription,
     &strAccountType))
      &accountElement = &anXmlNode.AddElement("GLAccount");
      &accountElement.AddAttribute("Number", &strAccount);
      &accountElement.AddAttribute("Type", &strAccountType);
      &accountElement.AddAttribute("Description",
&strAccountDescription);
   End-While;

   /* Convert document to stringified form, and return... */
   &xmlString = &MyDoc.GenXmlString();
   %Response.Write(&xmlString);
   End-Function;
```

Listing 3

```
...
private final static String GL_ACCOUNTWS_URL=
"http://dcore04:8000/xmllink/core/JK_WSILINK?userid=VP1&pwd=VP1&run-
   service=yes&accountSetID=MIS";
private static DocumentBuilder documentBuilder;

public static void main(String[] args)
{
   ...
   URL wsURL = new URL(GL_ACCOUNTWS_URL);
   URLConnection urlConnection = wsURL.openConnection();
   urlConnection.setDoInput(true);

   InputStream iStream = urlConnection.getInputStream();

   DocumentBuilderFactory factory =
      DocumentBuilderFactory.newInstance();
   documentBuilder = factory.newDocumentBuilder();
   Document textDocument = documentBuilder.parse(iStream);

   NodeList nodeList =
      textDocument.getElementsByTagName(GLAccount”);

   if (nodeList != null)
   {
            for (int nodes=0;nodes<nodeList.getLength();nodes++)
            {
                  Node aNode = nodeList.item(nodes);
                  handleAccount(aNode);
                  ...
```

Web Services Interoperability

ENSURING INTEROPERABILITY USING WEBLOGIC WORKSHOP

BY ANBARASU KRISHNASWAMY

Web services has been promising to solve several business problems, including those of integration and interoperability. While Web services standards like SOAP, WSDL, and UDDI help to facilitate it, they don't really ensure it.

In the past, application-to-application integration and business-to-business integration were not easy due to the various platforms involved or different software used. With the advent of Web services and related standards, it all looks very simple. Web services standards have been widely adopted by several vendors, who provide their own implementation for these standards and tools to develop and deploy Web services.

With the evolution of these standards, multiple versions and multiple implementations of Web services coexist among the applications. Loose definitions in standards may be interpreted in different ways. For example, SOAPAction, defined in the SOAP 1.1 specification, may be interpreted as the intended target for the message or the name of the target service. Vendors provide tools to generate services and clients from WSDL; WSDLs are generated by tools and applications to describe Web service. All these result in Web service interoperability issues.

In order to keep Web services interoperable, we need to ensure that various vendor implementations are interoperable, the tools that are used do not affect the integrity of the Web services, and standards are strictly followed and implemented.

Several industry efforts like SOAP builders and the Web Service Interoperability Organization (WS-I) were started to address these issues. BEA has been actively participating in these efforts to make sure that BEA's Web services are interoperable. This article provides an introduction to Web services interoperability efforts and shows how some of these tests can be easily implemented and performed using WebLogic server tools and WebLogic Workshop. I also list some useful resources.

AUTHOR BIO...

Anbarasu Krishnaswamy, a senior principal consultant with BEA Professional Services, has several years of experience with BEA products and is a Sun-certified Java programmer and BEA-certified WebLogic Server developer. Anbarasu holds a master's degree in computer science and engineering and bachelor's in electrical and electronics engineering

CONTACT...

anbarasu@bea.com

SOAPBuilders

SOAPBuilders was created to promote interoperability as the SOAP specification evolves. The test labs provide test suites that can be used by SOAP implementations to test interoperability. SOAPBuilders defines four rounds of interoperability tests. The first three rounds are already available; the fourth was expected to be finalized in October 2002.

ROUND 1

This round defines echo tests for String, StringArray, Integer, IntegerArray, Float, FloatArray, Struct, StructArray, Void, Base64, and Date. In the echo tests, the data types sent by the client are simply echoed back by the server. While this may sound very simple, it tests several aspects of implementation including serialization, deserialization, encoding, conformance to standards, support for data types, etc. The Round 1 SOAP interoperability tests specification can be found at www.xmethods.net/soapbuilders/proposal.html. The WSDL can be found at www.xmethods.net/tmodels/InteropTest.wsdl.

ROUND 2

SOAPBuilders Round 2 defines the specifications for echo tests using SOAP 1.1 and Section 5 encoding. Round 2 has been classified into three test suites: base, Group B, and echo Header. Table 1 summarizes the test suite, URLs for proposal, and WSDL.

ROUND 3

This round proposed overall testing strategy to make sure that SOAP/WSDL tools can generate WSDL docs for given scenarios correctly, consume WSDL docs generated by others, and consume and reuse given WSDL docs. The focus of this round was to demonstrate WSDL interoperability between SOAP-based Web service toolkits.

Round 3 is split into groups D, E, and F. Group D targets the classic RPC development model. Tests are defined by a given WSDL definition that is used to generate both servers and clients.

Group E targets ensuring coverage of mainstream WSDL features. These tests rely on a server-centric development model, ensuring that clients can consume WSDL generated by servers and matching certain criteria.

PANACYA

www.panacya.com

Group F tests are defined in terms of WSDL documents to test the ability of a toolkit to generate and execute client code based on given WSDL without error. More information on round 3 can be found at www.whitemesa.com/r3/interop3.html.

ROUND 4

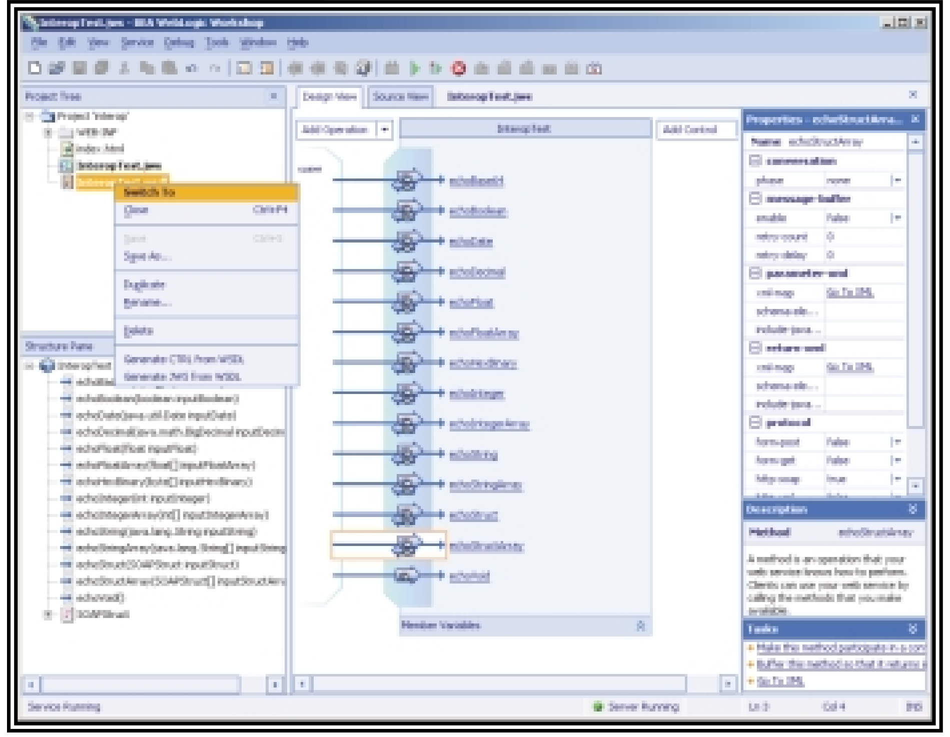
This round is divided into groups G, H and I. Group G focuses on testing the ability to send and receive attachments using SwA (SOAP messages with Attachments) and DIME (Direct Internet Message Encapsulation).

TABLE 1

	Base	Group B	Group C (Echo Header)
Tests	echoString	echoStructAsSimpleTypes	echoMeStringRequest header entry
	echoStringArray	echoSimpleTypesAsStruct echo2DStringArray	echoMeStructRequest header entry
	echoInteger		
	echoIntegerArray	echoNestedStruct echoNestedArray	Unknown header entry
	echoFloat		
	echoFloatArray		
	echoStruct		
	echoStructArray		
	echoVoid		
	echoBase64		
	echoHexBinary		
	echoDate		
	echoDecimal		
	echoBoolean		
Proposal URL	http://www.whitemesa.com/interop/proposal2.html	www.whitemesa.com/interop/proposalB.html	www.whitemesa.com/interop/proposalC.html
WSDL URL	www.whitemesa.com/interop/InteropTest.wsdl	www.whitemesa.net/interop/InteropTestB.wsdl	www.whitemesa.net/interop/InteropTestC.wsdl

Test suite, URLs for proposal and WSDL

FIGURE 1



Implementing round 2 base test

WSDL documents for RPC style and Document/literal style test cases can be found at www.whitemesa.net/r4/interop4.html.

Group H addresses fault message processing and Group I focuses on WSDL/XSD testing.

SOAPBUILDERS FORUM

A Yahoo group, http://groups.yahoo.com/group/soapbuilders, has been created to facilitate communication about interoperability issues. This group is a forum for builders of SOAP implementations to discuss cross-implementation interoperability issues, SOAP spec interpretations, etc.

WS-I

WS-I is an open industry organization chartered to promote Web services interoperability across platforms, operating systems, and programming languages. The organization works across the industry and standards organizations to respond to customer needs by providing guidance, best practices, and resources for developing Web services solutions.

While SOAPBuilders focuses on testing tool interoperability, WS-I focuses on providing the Web service implementer or developer with a set of tools and implementation guidance to ensure that the services they build can interoperate and conform to the current Web service standards. Whereas testing materials from other organizations tend to focus on a particular specification, WS-I–created test tools will address interoperability at a level above that of specification-by-specification and help determine the overall conformance of a given Web service to a profile or set of specifications that may come from various sources.

WS-I.org has 3 primary goals:

- ***Provide*** implementation guidance and education to help customers with Web services adoption.
- ***Promote*** consistent and reliable interoperability among Web services across platforms, applications, and programming languages.
- ***Articulate*** and promote a common industry vision for Web services interoperability to ease customer decision making, grow industry adoption of Web services and ensure the continued evolution of Web services technologies.

WebLogic Web Services Interoperability

You can easily implement and test the interoperability tests using the tools provided by WebLogic Server 7.0. WebLogic Server provides plenty of powerful tools and utilities to develop and test Web services and Web services clients. This section shows you how to use WebLogic tools like WebLogic Workshop and clientgen ant task to effectively implement and perform the tests. While this article does not attempt to show how to implement all the test suites, it explains some nice features of WebLogic utilities that can be used to easily develop and test these services.

CREATING ECHO WEB SERVICE

To implement a test from the given WSDL, WebLogic workshop offers a feature to generate JWS file from WSDL. Once the JWS file is generated, it can be edited to complete the implementation. For example, to implement the round 2 base test using Workshop, follow the steps listed below:

- Copy the WSDL file from SOAP Builders Web site
- Create a new text file in workshop with .wsdl extension, paste the WSDL file into it.
- Edit the service end point information in the WSDL file. If the <service> section is not already present, add it. Enter appropriate service name, port name, and location. For example:

```
<service name="InteropTest">
  <port name="InteropTestPort"
binding="tns:InteropTestSoapBinding">
    <soap:address location="http://local-
host:7001/interop/InteropTest.jws"/>
  </port>
</service>
```

- Save the file, right click on it in the project tree and choose “Generate JWS from WSDL”.
- Workshop generates the Web service as shown in Figure 1.
- Go to the source view tab and complete the methods. In our case, to implement the echo test, just type “return <input argument>;” for all the methods except echoVoid().
- When the service is run from Workshop, it displays the test form as shown in Figure 2. It may show that the operation is not supported over HTTP-GET. If you want to test the service using the test form, you will have to disassociate the WSDL in the JWS file and set form-get and form-post to “true” for each method in the property editor.
- The Web service can be tested using clients generated externally but still the logs can be viewed using the test form. Writing client programs is explained later in this article.

CREATING CLIENTS

As explained earlier, SOAPBuilders’ round 3 defines tests that ensure interoperability between generated clients, prebuilt server and generated servers. Client proxies can be generated using clientgen ant task or WebLogic workshop. This section discusses how to generate and code clients using each of these methods.

Generating client proxy using clientgen

Clientgen ant task can be used to generate the client proxy to test the InteropTest Web service we just created. Following is an excerpt from the build.xml file (source code for this article may be found at www.sys-con.com/weblogic/sourcec.cfm):

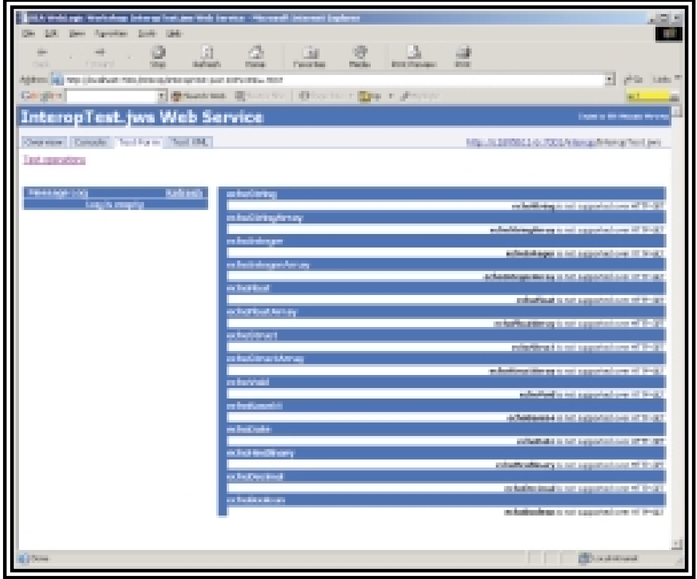
```
<target name="build" depends="check">
  <mkdir dir="${build}"/>
  <clientgen
wsdl="http://localhost:7001/interop/InteropTest.j
ws?WSDL"
   serviceName="InteropTest"
   autotype="True"
   overwrite="False"
   useServerTypes="False"
   packageName="com.bea.interoptest"
   clientJar="${build}"/>
</target>
```

When the ant task runs, it creates the client proxies for the Web service specified by the WSDL URL in the build directory under the package specified by packageName. This could point to a local Web service or a remote service on the Internet. It should be noted that the client proxy can be generated for Web services implemented on other vendor platforms too. The serviceName parameter is the name of the service defined in the WSDL.

Proxy Wrapper

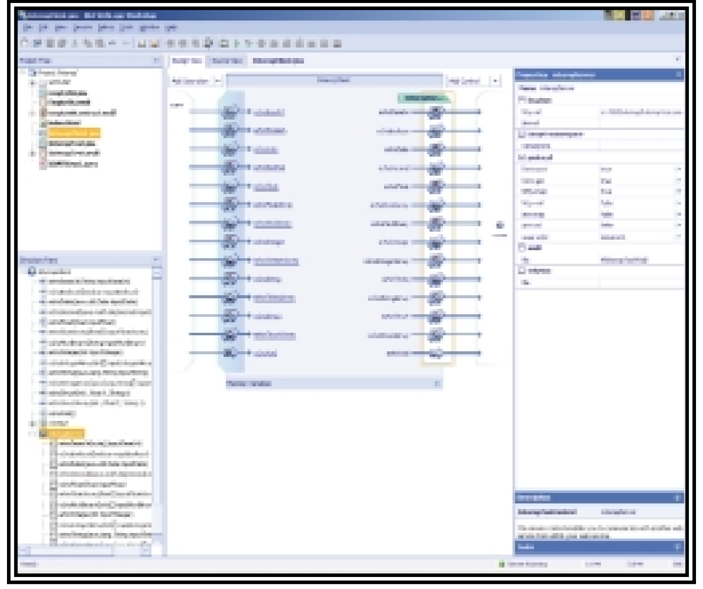
The following listing shows the proxy wrapper that invokes the client proxy to call the Web service.

FIGURE 2



Test form for initial echo test service

FIGURE 3



Developing client for a remote Web Service

```
private InteropTestPortType service;
// Constructor
public ProxyWrapper() {
try {
            service = new
InteropTest_Impl().getInteropTestPortType();
} catch(Exception e) {   /*Process Exception
here*/    }
 }
// echoString
  public String echoString(String input) {
  try {
                   return
service.echoString(input);
  } catch (Exception e) { /* Process Exception
here*/    }
  }
```

Clientgen generates several classes, two of which are important for us. One class is derived from the service name; the other is derived from the port name specified in the WSDL. You can also verify the class names by checking the package directory specified in the build.xml file. This proxy wrapper nicely encapsulates the client proxy access details and may be used by a test driver like jUnit test case to invoke the service. The following listing shows a sample jUnit test case using the proxy wrapper:

```
import junit.framework.TestCase;
……
public class InteropTest extends TestCase {
…….
    public void testEchoString() throws
Exception {
         ProxyWrapper soapInterop = new
ProxyWrapper();
        String input = "Foo";
        assertEquals("Failed on echo String",
input,soapInterop.echoString(input));
    }
……..
}
```

CREATING CLIENT USING WORKSHOP

One of the nice features of WebLogic Workshop is the service control. To create a client for a Web service, follow these steps:

- Create a Web service, InteropClient.jws from the WSDL as explained in the "Create Echo Web Service" section
- Add a service control and input the WSDL URL and variable name for the Web service. The design view looks like Figure 3.
- In the method definitions, proxy the requests to the service control. For example:

```
  /**
     * @jws:operation
     * @jws:protocol soap-style="rpc"
     */
    public java.lang.String echoString
(java.lang.String inputString)
    {
        return
interopServer.echoString(inputString);
    }
```

- If you want to test using the test form, accept String for echoByte and echoByteArray and pass String.getBytes() to the service control
- Accept elements of the structure as input and populate the structure in the code for echoStruct and echoStructArray as structures can not be entered directly.
- Figure 4 shows the Test form for the InteropClient service.

INTEROPERABILITY WITH OTHER VENDORS

www.whitemesa.net lists several live end points for several vendors to test various rounds of the interoperability test. Clients can be generated using one of the methods described in the previous section and interoperability test may be performed.

BEA INTEROPERABILITY SERVER

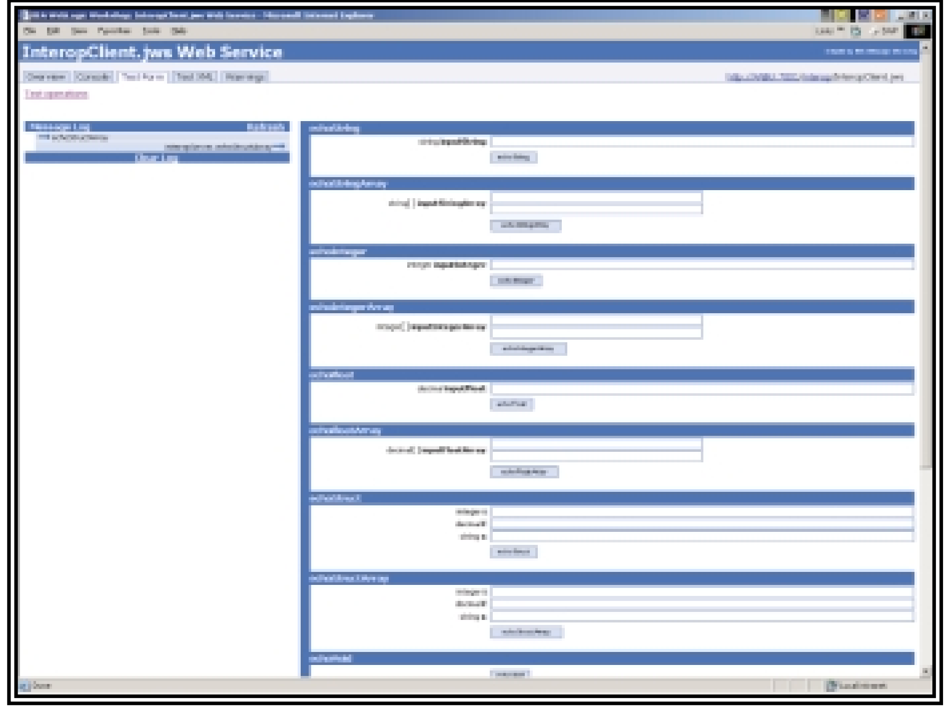
BEA maintains an interoperability server and test implementations in an effort to support Web services interoperability. This server is available at http://Webservice.bea.com:7001/index.html for SOAP 1.1.

SOAP 1.2 versions of these tests are available at http://webservice.bea.com:9001. WSDLs for various interoperability tests may be obtained from this server. This Web site also has browser clients to perform the tests.

Resources

- *SOAP Builders interop Web site:* www.whitemesa.com/interop.htm
- *SOAP Builders Web site:* *www.whitemesa.net*
- *SOAP Builders interoperability lab:* www.xmethods.net/ilab
- *BEA dev2dev Web services page:* http://dev2dev.bea.com/managed_content/direct/webservice/index.html
- BEA interoperability test server for SOAP 1.1: http://webservice.bea.com:7001/index.html
- *BEA interoperability test server for SOAP 1.2:* http://webservice.bea.com:9001/index.html
- Chappell, David A. and Jewell, Tyler (2002). Java Web Services. O'Reilly & Associates.
- *SOAP Builders forum* http://groups.yahoo.com/group/soapbuilders

FIGURE 4

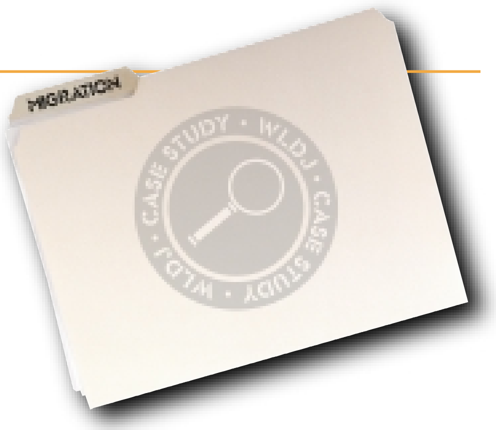


InteropClient.jws Web Service

Running the client using Test form

Covasoft

www.covasoft.com/today's_tech/



Taking the Migraine Out of Migration

APPLICATION MIGRATION FROM ATG DYNAMO TO BEA WEBLOGIC

BY
SHYAM NAGARAJAN

AUTHOR BIO...
Shyam Nagarajan is a senior consultant at Cacheon, Inc., providing customers with migration solutions. Previously, he was involved in designing and developing software solutions based on ATG and BEA technologies. Before joining Cacheon, Shyam worked on developing large financial systems at Logica.

CONTACT...
shyam.nagarajan@cacheon.com

When J2EE was still in its nascent stage, ATG took a bold step, becoming one of the first vendors to provide a similar framework for enterprises to build their applications on. Though largely based on Java, their technology did not adhere to any of the earlier J2EE specifications. It was quite cumbersome for ATG customers to keep up with the J2EE technology. As a result, many ATG customers were left with huge code bases for applications that are difficult and expensive to maintain.

A Developing Headache

One of our customers, a financial services firm running their e-commerce sites on ATG Dynamo 4.5, merged with a firm that ran their applications on BEA WebLogic 6.1. The newly formed management mandated that all business applications and e-business components be standardized to run on one platform so both businesses could share intellectual property – applications, components, and data libraries. They chose WebLogic to be the standard platform because of its rich set of features, including security, scalability, and standards-based architecture.

Their IT organization was tasked with finding a cost-effective solution for moving all business applications to WebLogic while minimizing disruption to customers. However, the business applications were updated frequently with new components and services, so the solution required moving applications quickly to ensure that all updates were captured.

Two options were considered for this task: elimination or migration.

- ***Elimination:*** Rewrite existing ATG applications for the WebLogic platform. Since the ATG Dynamo Server is not J2EE compliant, applications written for it aren't easily portable to WebLogic 6.1. Rewriting is a reasonable solution if the application is small (e.g., less than 5,000 lines of code). However, rewriting an application can cost nearly as much as developing the original application. For sophisticated e-commerce sites that cost over $800,000 and 15–20 months to build, rewriting becomes a less reasonable choice.
- ***Migration:*** Until recently, application migration was completed manually either by in-house developers or outside consultants. Considering that vendors often implement the J2EE specification in nonuniform ways, the process of identifying and updating proprietary code to be compatible with a new platform can be time intensive. Consequently, the advantage of migration over rewriting was only marginal at best.

However, an automated migration tool vastly improves the process of moving applications, making migration a viable solution for enterprises seeking to standardize on a single platform. With the Cacheon Migrator (see Figure 1), automated application migration from one application server to another is possible, making the process of converting application source code faster and easier. The Cacheon Migrator automatically converts much of an application's source code. For the portion not automatically converted, it highlights problem areas and provides detailed solutions and workarounds.

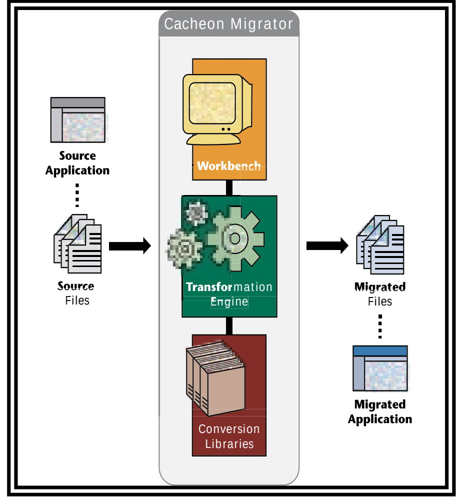
Thus migration using the Cacheon Migrator is a practical solution for extending the life of applications that need to run on a new or upgraded application server platform. As for our customer, their IT organization decided to use one application to evaluate migration as a solution for moving their applications from ATG Dynamo Server to WebLogic Server.

A Prescription for Migration

A four-step plan was devised to scope the size of the project, measure success, perform the conversion, and verify results.

- ***Planning:*** We began by assessing application candidates and code changes required to maintain the runtime environment. Once the application was selected, it needed to be prepared for migration. This involved removing unused code, consolidating existing code, and identifying missing properties and segments that required rearchitecture in WebLogic 6.1. Next, WebLogic Server, the Cacheon Migrator, and other development and deployment tools required for migration were installed. Following this, it was verified that third-party libraries could run on WebLogic.
- ***Analysis:*** Once the planning was complete, the IT organization performed compliance analysis on all libraries. Project schedules were created and resources were assigned.
- ***Execution:*** Once resources were identified, code development was frozen so that migration of core classes and libraries could be initiated. The customer ran the ATG application source code through the Cacheon Migrator. The Cacheon Migrator converted JHTML files to JSP files, repackaged droplets and components as JavaBeans and tag libraries, and identified the usage of initial services and scheduling services within the code base as potential startup class candidates. Once the automated conversion was complete, the IT organization reviewed the migration report, which was available in both HTML and CSV formats, and manually fixed identified problem areas.
- ***Validation:*** After the application source code was completely converted, testing commenced. This included migrating existing system test and quality assurance environments to WebLogic. Once the environments were set up, QA environment tests, functional tests, regression tests, and user acceptance tests were run. When the application passed all tests, it was moved to the production environment in stages as "branches" off the load balancer.

FIGURE 1



The Cacheon Migrator at work

Migration Pain Points

The key issues discovered while migrating from ATG Dynamo to WebLogic can be categorized into the following buckets:

- JHTML pages
- ATG Nucleus
- Custom Droplets (Dynamo Servlet Beans)
- Form handlers
- Relational views
- Initial services and scheduler services
- Servlet pipeline

JHTML PAGES

JHTML pages, analogous to JSP in the J2EE world, are considered one of the dogged issues in ATG migration. The JHTML format predates J2EE and thus does not follow any of the JSP specifications. As a result, the syntax and workings of the tags are different. It's quite common to find JHTML pages cluttered with business logic, as there were few ways to distinguish between the presentation layer and the business layer.

In order to migrate applications, all JHTML pages need to be converted to JSP pages on a one-to-one mapping basis. The developers responsible for the migration need to be proficient in understanding JHTML intricacies, as well as J2EE equivalents. The components and droplet instantiation code within the JHTML pages needs special attention, as they could be interdependent. The only reasonable way to identify these dependencies is by poring through the property files of the components.

ATG NUCLEUS

Nucleus is Dynamo's framework for running server applications built from JavaBean components. Nucleus is responsible for creating and initializing components within the Dynamo application server and is similar to the JNDI services within the J2EE world. A basic operation of the Nucleus is resolving component names. The framework and the design of the Nucleus, however, cannot be directly migrated onto any J2EE application server. This presents a challenge for the architects and developers trying to find the appropriate Nucleus dependencies and identify a workaround that offers the same functionality.

CUSTOM DROPLETS

Custom Droplets, also referred to as Dynamo Servlet Beans, are commonly used as droplet tags within JHTML pages. To migrate Custom Droplets requires understanding the functionality of the droplet as well as the scope in which it is used. Custom Droplets are common within ATG Web applications.

FORM HANDLERS

Form handlers within JHTML pages handle user inputs from forms. It's possible in JHTML pages to embed Java/JHTML constructs within the HTML form tags. This makes it difficult to separate the HTML code from the control logic.

RELATIONAL VIEWS

Relational views (RVs) are data store retrieval APIs that facilitate communication between the data stores and the presentation layers. RVs, which are proprietary in their implementation, are commonly used within an application. RVs offer a means to do object-to-data mapping within ATG – roughly equivalent to CMP in J2EE.

INITIAL SERVICES AND SCHEDULER SERVICES

ATG offers configurable bootstrap services used to run services at the startup of an application server. This is achieved by implementing InitialServices and scheduler APIs, and by registering the appropriate class within the application server. J2EE does not necessarily have an equivalent, thus a developer must create an equivalent within the appropriate platform.

SERVLET PIPELINE

Servlet pipeline components exert control on the request handling of pages within an application server. Servlet pipelines are often used to implement custom request handling, especially security authentication mechanisms. The J2EE specification did not provide an equivalent until the Servlet 2.3 Specification was released. Although there isn't an explicit one-to-one mapping, the functional equivalents of security realm authentication mechanisms allow a developer to achieve the same functionality as the servlet pipeline components.

The Cacheon Migrator Remedy

In this particular situation, the Cacheon Migrator performed the migration from ATG Dynamo 4.5.1 to BEA Weblogic 6.1. The Cacheon Migrator (see Figure 2) migrated the application source code and configuration files to the equivalent artifacts for the WebLogic platform. Given that the source application server was ATG Dynamo (pre-J2EE and many proprietary implementations), it was believed that some of the application code base might be unsalvageable.

The application had a number of JHTML pages that were tightly integrated with content that was made available through Documentum (a content management software solution), contained customer security authentication and image server redirection, and included URL manipulation performed through servlet pipeline components. The content also incorporated droplets and relational views for functional integration to database and legacy systems, and included initial services and scheduler services for e-mail campaigns and Web log management.

Cacheon Migrator addressed the migration of the aforementioned features in the following ways:

JHTML PAGES

The Cacheon Migrator converted the JHTML pages to JSP pages, keeping intact all of the text formatting as well as the positioning of the current file. All importbean and droplet tags were converted into J2EE-equivalent instantiation code and tag libraries, respectively. Furthermore, the Cacheon Migrator identified dependent components within properties files and instantiated them through useBean tags. All standard ATG droplets were converted into inline Java or tag library–based tags depending on requirements.

Additionally, to migrate custom droplets, the Cacheon Migrator was extended by writing additional rules that were used to convert droplet code into tag library–based tags or inline Java code. The Cacheon Migrator also altered the form handlers so they would function effectively within the JSP page without modification.

JAVA SOURCE CODE MIGRATION

Identifying all areas where proprietary APIs had been used was key for successful migration. The Cacheon Migrator intelligently parsed Java files into Java object models and validated nodes for possible matches against the ATG to BEA migration rules.

One case worth highlighting looks at the atg.servlet.DynamoServlet class, which is a custom extension of HttpServlet class. During migration, code that uses DynamoServlet classes must either be converted to the HttpServlet equivalent where applicable, or the custom methods and properties of the DynamoServlet class must be replaced with their functional equivalents. The Cacheon Migrator performed this programmatic refactoring automatically using existing rule sets. Similarly, all ATG Nucleus resolve name calls were translated into HttpServletRequest.getsession().getAttribute(beanName) calls depending on the context of usage.

Additionally, Servlet pipeline components were converted into Servlet filter classes with appropriate APIs.

RELATIONAL VIEWS MIGRATION

The Cacheon Migrator mapped all relational view code to equivalent JDBC calls within the J2EE specifications. Additionally, it flagged issues that required manual intervention where there was no direct mapping.

INITIAL SERVICE AND SCHEDULER SERVICE CLASSES

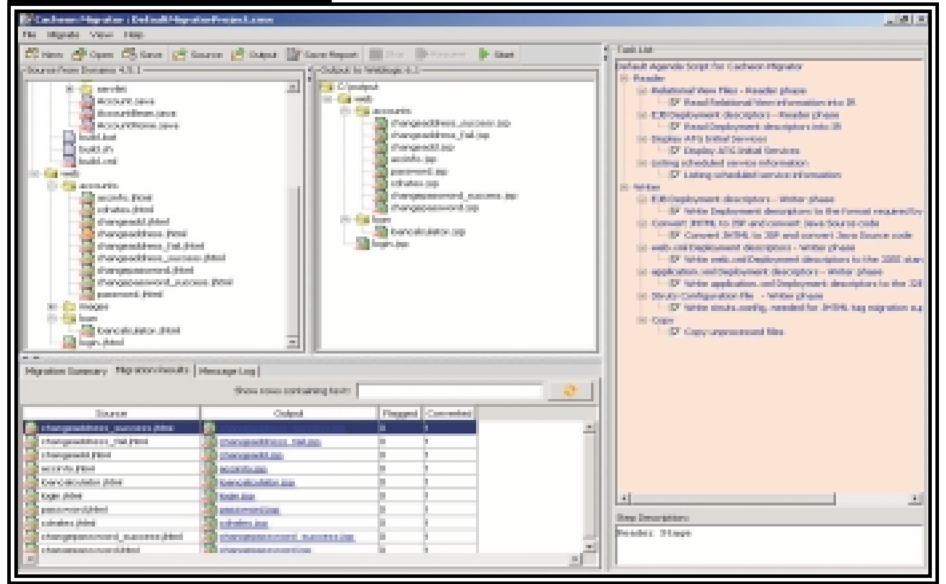
A warning was provided for all occurrences of initial service classes and scheduler service classes that were identified in the configuration files and the Java code base.

Results

The ATG application was analyzed and converted to a WebLogic deployable application. The software automatically converted 85% of the JHTML files and nearly half of the core libraries and files. The code that needed manual attention was flagged with detailed guidelines for manual intervention. Manual steps ranged from locating reference files and renaming file directories to converting logging classes and replacing queue classes. Altogether, the application was deployed on the WebLogic platform in three weeks – one week for conversion and two weeks for testing and deployment.

Manually migrating applications can be as time-consuming and resource intensive as rewriting the applications for a new platform. However, with proper planning and the right software, migration becomes a viable solution for moving applications from legacy application servers to newer application servers.

FIGURE 2



The Cacheon Migrator

THE WORLD'S LEADING INDEPENDENT WEBLOGIC DEVELOPER RESOURCE

Helping you enable intercompany collaboration on a global scale

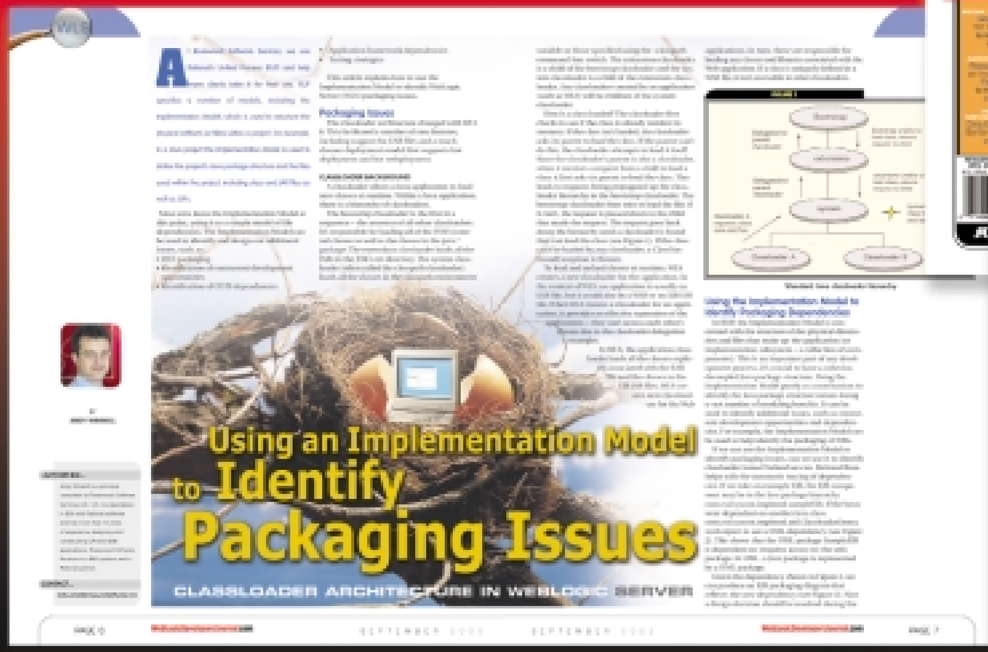
• Product Reviews
• Case Studies
• Tips, Tricks and More!

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

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

Go Online & Subscribe Today!

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



SYS-CON MEDIA

WebLogicDevelopersJournal.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.

Security and JMS Coverage Are Highlights of "Bible"

BY JASON WESTRA

As good as product documentation gets, there is always room for more code samples, deployment descriptor samples, and tips on how to take advantage of undocumented tools. While integrating WebLogic Server 6.1 as a product offering for my company's hosting platform, I needed examples for configuring WebLogic JDBC and JMS that the standard documentation (or lack thereof) could not provide. I got the information I needed from **BEA WebLogic Server Bible**.

This book covers version 6.1 of the server, and includes support for SOAP, Web services, and JCA (Java Connector Architecture). These are cool technologies to learn about; however, keep in mind that WebLogic Server 7.0 has now been released with more advanced support for them.

Each of the book's eight sections contains multiple chapters. Most begin with an overview of a technology or standard and then describe how WebLogic supports the technology. The chapters walk the reader through how to use the new WebLogic Console, while explaining the configuration of services through samples, text, and UI screen shots. For those of you who only modify your WebLogic configuration file by hand, the bible offers no reciprocal for you to understand the config.xml elements and attributes. Using the console and then looking at the generated XML in config.xml is the best way to learn the config.xml DTD quickly.

Sections Reviewed

To help you understand what *BEA WebLogic Server Bible* offers, I have briefly reviewed each section below.

Section 1, "Preparing Your Enterprise for WebLogic," covers many J2EE project basics, including how to select a server based on application requirements, and what your team's skill sets should be to accomplish a full J2EE development life cycle. If you are a WebLogic project veteran, you'll move on to the sections containing samples and technical jargon. Otherwise, these first chapters are nice primers for WebLogic project management and development newbies.

Section 2, "WebLogic J2EE APIs," is the largest section in the book. Chapters 5–10 deal with the J2EE APIs supported by WebLogic Server, such as JDBC, JTA, JNDI, RMI, JMS, and JavaMail, but don't cover JSP, servlets, or EJBs. Since these are the most commonly used building blocks for J2EE applications, they are covered more extensively in their own sections.

I found the section on JDBC to be helpful in describing WebLogic's support for multipools (pools of JDBC pools), but it only had two sentences on clustered JDBC and not many more than that in Chapter 24, "Working with WebLogic Clusters." I enjoyed the JNDI chapter's coverage of serious topics like binding your own objects into WebLogic JNDI in a clustered environment, and performing directory operations within Microsoft's Active Directory. The RMI chapter covered many RMI basics but also clarified the differences between Java RMI and WebLogic RMI, BEA's optimized implementation. This chapter builds upon the JNDI chapter because remote objects can be registered and looked up via JNDI.

Last, the JMS chapter was very extensive. I found it helpful in configuring JMS servers, topics, and queues because the properties were described more eloquently than by the config.xml documentation on the BEA Web site. Support for persistent messages is covered, including a file-backing store and JDBC tables. JMS and transactions, a topic often misunderstood, is likewise covered. How do you do an asynchronous, distributed, XA-compliant transaction anyway?

Section 3, "Developing Web Components," covers Web application development using JSP and servlets in the WebLogic Web container. This section highlights the WebLogic-specific deployment descriptor for the Web container, weblogic.xml. For instance, the book shows how to set JSP parameters in weblogic.xml that were once configured in weblogic.properties. Unfortunately (or fortunately, depending on which way you look at it), JSPs are pretty standard, and BEA hasn't written many enhancements to the Web container. The chapter on JSP does have a few pages on custom WebLogic tag libraries, which is nice.

Section 4, "Developing EJB Components," is devoted to WebLogic Enterprise JavaBeans. Chapters 14–17 cover each type of EJB supported by WebLogic, including entity beans, session beans, and message-driven beans. These chapters contain numerous detailed samples and descriptions of the configurable properties in the WebLogic EJB container. Some clustering is mentioned, but more detailed clustering information can be found in a later chapter.

Section 5, "Deploying and Testing Enterprise Applications," walks you through the process of testing and tuning your WebLogic application. Tuning options discussed include JVM optimizations, JDBC tuning, caching strategies, and more obvious options like excessive logging and synchronization in your applications. I was disappointed that none of the chapters in this section were devoted to WebLogic development and deployment using Jakarta Ant, which has become the universal build tool for Java applications. Perhaps Ant will be added to this section in the book's next release!

Section 6, "Implementing Security," has three chapters. The first provides an overview of security principles, Java security concepts, and types of attacks. The second chapter covers WebLogic's security architecture to help you understand the types of protection WebLogic provides out of the box, and what needs to be done programmatically. The last chapter in this section walks you through how to secure an application using JAAS and a security realm based on JDBC.

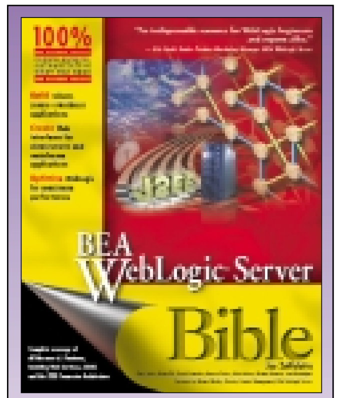
Section 7, "WebLogic Server Administration," provides an in-depth examination of the properties you can modify and monitor within the WebLogic Console. It's a must-read for WebLogic administrators! This section also contains more details on clustering and security administration, with a chapter devoted to each. Last, each type of supported realm (file, LDAP, RDBMS, and Unix) is described.

Section 8, "Enterprise Application Integration," covers two hot areas in enterprise development, Web services and the JCA. Within this section, the book describes how WebLogic has approached Web services. It shows how to use a custom Ant task, wsgen, to automate the construction of Web services and provides numerous examples. It also elaborates on WebLogic-specific deployment settings that can be used for JCA Connectors deployed on WebLogic.

IMHO

Even though WebLogic Server 7.0 is shipping, many organizations continue to deploy on version 6.1. WebLogic Server security is really the only area that has changed enough between 6.1 and 7.0 to outdate the book's coverage of this topic. The chapter devoted to performance tuning was weaker than I'd hoped, so I recommend looking elsewhere for this information. In particular, check out TheServerSide.com (www.theserverside.com) for ECPerf results on the BEA WebLogic Server Platform. Finally, I'd like to see more examples of Ant used as the build tool of choice in this and other WebLogic books. There was only a smattering of Ant in the last section on Web services.

In general, the book's detailed code and deployment descriptor samples are outstanding. I think its security and JMS coverage is excellent, as well as its JSP, servlet, and EJB coverage. I highly recommend *BEA WebLogic Server Bible* for beginners, yet I think even advanced users can glean knowledge from a number of its sections.



Title:
BEA WebLogic Server Bible

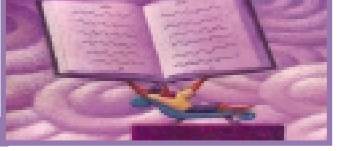
Author:
By Joe Zuffoletto, Gary Wells, Brian Gill, Geoff Schneider, Barrett Tucker, Rich Helton, Michael Madrid, and Sunil Makhijani

Publisher:
John Wiley & Sons

ISBN:
0764548549

Pages:
973

List price:
$49.99



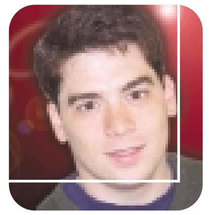
Performant Software Solutions

www.performant.com/weblogic1

High-Performance CMP Features

THE MORE ADVANCED OPTIMIZATION TECHNIQUES

BY SAM PULLARA



This month I've decided to explore some of the more advanced performance enhancements that you can use if you are using EJB 2.0 on WebLogic. Our container-managed persistence (CMP) engine exposes several strategies for you to configure to get the most efficient – meaning least – use of your database. Field-groups allow you to specify which fields are loaded from the database together. Relationship caching tells the CMP engine to load the related bean when the parent bean is loaded. Cache-between-transactions allows you to cache the contents of entity beans between transactions against that bean. This is combined with Optimistic concurrency to get very good guarantees. Finally, there is ReadOnly concurrency, which gives you great performance with the ability to flush programmatically or through timeout. Using these optimization strategies, you will easily surpass the performance of naively written bean-managed persistence (BMP) entities and even exceed the performance of perfectly written J2EE-compliant BMP entities without the added maintenance and complexity that writing your own persistence layer can entail.

When you access a bean through a finder the CMP engine will by default load all the fields of the entity from the datastore. In many circumstances there are fields that you know you're not going to use in a certain code path; these fields need not be queried nor read from the database. Using field-groups you can specify which fields are to be loaded and the other fields will be loaded on demand. Field-groups do not stop you from eventually accessing the field and loading the data; it's just put off until the field is accessed by the application, or never loaded at all if you never access it. As a simple example, imagine a search application. The entities that you are searching for might contain the following fields: URL, Summary, Date, Keywords, and Cached Content. When you do your initial query of the database and return results for the user to look at, you probably don't want to load the cached content. It is a large field and if the user doesn't look at all of the results you're wasting a lot of work reading from the database.

In this example you would put the URL, Summary, and Date in a field-group and assign that group to the findByKeywords finder in the entity bean. When the finder was called, you'd get a collection of results with their designated fields prepopulated, and if the user happened to ask for the cached content the CMP engine would automatically go back out to the database and populate it. Some experimentation and benchmarking may be required to get the field-groups exactly right. Sometimes, if you misidentify which fields are really needed, you can reduce performance by not loading a field that is often used after the query. In order to activate this optimization you simply declare what groups each CMP field belongs to using the group-names attribute on the cmp-field entry and then associate a group with the finder using the group-name attribute.

Relationship Caching

Relationship caching is very important when the related data is usually used to access the parent bean. It reduces the number of SELECTs against the database by including the related bean fields in a SQL join. For one-to-one relationships this can offer a huge increase in performance because no extra work is done while you are reducing the SELECT statements. In the one-to-many case it will often depend on the fields present in your parent bean because under the join you will read those fields once for each related bean. I suggest that in this case you analyze the typical number of related beans and determine if it makes sense to take the extra per row performance hit in order to reduce the number of SELECT statements and the number of round-trips to the database.

As an example, if you have an Employee bean that also has a one-to-many relationship with Address beans, and when you access the Employee bean you often read the address data, you would probably enable this option because most Employee beans would have one or two related Address beans. In the case of the same Employee bean being related to PayrollStatement beans, you probably wouldn't want to enable relationship caching because the number of statements could be quite high and you would not be referencing them all every time you viewed the Employee bean.

Optimistic Concurrency

Perhaps the biggest performance increase you can get is by enabling cache-between-transactions and choosing Optimistic concurrency in uncontentious applications. Caching between transactions allows the CMP container to avoid returning to the database between every different use of the bean. Additionally, the application server will send out flushes on updates to the cached beans (even in a cluster) so that they will not be overly stale. With Optimistic concurrency enabled, any updates that are done have an included WHERE clause that checks to make sure that the row that is being updated hasn't been changed since the data was read from the database.

–continued on page 39

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

WebServices .NET J2EE XML JOURNAL

The Best .NET Coverage Guaranteed!

LEARN WEB SERVICES. GET A NEW JOB !

Subscribe today to the world's leading Web Services resource

Get Up to Speed with the Fourth Wave in Software Development

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

Only $69.99 for 1 year (12 issues)*
* Newsstand price $83.88 for 1 year
Subscribe online at www.wsj2.com or call 888 303-5252
*Offer subject to change without notice

SYS-CON MEDIA

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

Administrative Tasks Using the weblogic. A d m i n C o m m a n d- Line Utility

A STEP-BY-STEP GUIDE

BY KUMAR ALLAMRAJU

AUTHOR BIO

Kumar Allamraju is a senior developer relations engineer at BEA Systems in the WebLogic server support division. Kumar has five years of experience in object-oriented programming and in J2EE-related technologies.

CONTACT...

kumara@bea.com

WebLogic Server (WLS) provides several ways to configure servers, clusters, machines, JDBC connection pools, JMS servers, and so on, using the following:

- ***Domain Configuration wizard:*** GUI tool
- ***Admin console:*** Browser-based GUI interface
- ***Programmatic JMX API interface***
- ***weblogic.Admin command-line utility***

The command-line interface comes in handy if you want to integrate this tool into Perl or Ant scripts for administration and management efficiency, if you can't access the Admin console through a browser, or if you prefer using command-line tools over a GUI interface.

In this section we'll focus on doing admin tasks using the weblogic.Admin command-line utility. I've assumed that you're using WLS 7.0, although the same syntax should work fine in the 6.x release.

Before invoking the weblogic.Admin utility, set the WLS development environment. In Windows it's setWLSEnv.cmd; in Unix platforms it's setWLSEnv.sh. These files are located under $WL_HOME/user_projects/your-domain folder.Let's try some basic MBean commands using WebLogic. Admin utility.

1. To create a server named "WLDJServer", use the following example:

```
<<Note: for the blue text, set apart and use different
font but not code font>>

java weblogic. Admin -url {admin-serverl url} -username
{admin-user} -password
{admin-password} CREATE -mbean {object_name}
```

where object_name is in the form of

```
"domain-name:Name={server name}:Name,Type={type of
Mbean}"
```

Example:

```
java weblogic. Admin -url {admin-serverl url} -username
{admin-user} -password
{admin-password} CREATE -mbean "WLDJDomain:
Name=WLDJServer, Type=Server"
```

where "WLDJDomain" is your domain name. After executing the above command, write the entries shown in Listing 1 into config.xml.

2. To create a cluster named "WLDJCluster", use the following example:

```
java weblogic. Admin -url {admin-serverl url} -username
{admin-user} -password
{admin-password} CREATE -mbean {object_name}
```

where object_name is in the form of

```
"domain-name:Name={cluster name}:Name,Type={type of
Mbean}"
```

Example:

```
java weblogic.Admin -url {admin-serverl url} -username
{admin-user} -password
{admin-password} CREATE -mbean
WLDJDomain:Name=WLDJluster,Type=Cluster"
```

After executing the above command, write the following into config.xml:

```
   <Cluster Name="WLDJCluster"/>
```

3. To set/change the configuration attributes of this Mbean, use the weblogic.Admin SET option:

```
java weblogic. Admin -url {admin-serverl url} -user-
name {admin user} -password
{admin password} SET -mbean "{object_name}" -property
{property_name} {property_value}
```

where property_name can be one of the Mbean attributes, such as MulticastAddress, Cluster-Address, and InterfaceAddress, and so on.

Example:

```
java weblogic. Admin -url {admin-server url} -username
{admin user} -password
{admin password} SET -mbean
"mydomain:Name=WLDJCluster,Type=Cluster"
-property "MulticastAddress" "224.0.0.1"
```

After executing the above command, write the following entries into config.xml:

```
<Cluster MulticastAddress="224.0.0.1"
Name="WLDJCluster"/>
```

4. To assign the managed server created in step 1 to a cluster, use weblogic.Admin SET:

```
java weblogic. Admin -url {admin-serverl url} -user-
name {admin user} -password
{admin password} SET -mbean "{object_name}" -property
{property_name} {property_value}
```

Example:

```
java weblogic. Admin -url {admin-server url} -username
{admin-user} -password
{admin-password} SET -mbean
"mydomain:Name=WLDJServer,Type=Server"
-property Cluster
"WLDJDomain:Name=WLDJCluster,Type=Cluster"
```

The above command will update the config.xml with the following entries:

```
<Server Cluster="WLDJCluster" Name="WLDJServer">
 </Server>
```

5. To view configuration attributes of a particular Mbean, use weblogic.Admin GET:

```
java weblogic.Admin -url {admin-server url} -username
{admin-user} -password
{admin-password} -pretty GET -type {Config Mbean}
```

where ConfigMBean could be ServerConfig, ExecuteQueueConfig, DomainConfig, and so on.

Example:

```
java weblogic.Admin -url {admin-server url} -username
{admin-user} -password
{admin-password} -pretty GET -type ServerConfig.
```

6. To view runtime statistics of a particular Mbean, use weblogic.Admin GET:

```
java weblogic. Admin -url {admin-server url} -username
{admin-user} -password
{admin-password} -pretty GET -type {ConfigMbean}
```

where ConfigMBean could be ServerConfig, ExecuteQueueConfig, DomainConfig, and so on.

Example:

```
java weblogic. Admin -url {admin-server url} -username
{admin-user} -password
{admin-password} -pretty GET -type ServerRuntime.
```

7. To create a connection pool, use weblogic.Admin CREATE_POOL:

```
java weblogic. Admin -url {admin-server url} -username
{admin-user} -password
{admin-password} -pretty CREATE_POOL {pool_string}
```

Example:

```
java weblogic. Admin -url {admin-server url} -username
{admin-user} -password
{admin-password} -pretty CREATE_POOL WLDJ817ThinPool
"url=jdbc:oracle: thin:
@baybridge:1521:bay817,driver=oracle.jdbc.driver.Oracle
Driver,
initialCapacity=1,maxCapacity=1,props=user=SCOTT;pass-
word=tiger;"
```

Summary

The weblogic.Admin utility provides a powerful command-line interface to accomplish administrative tasks that can be done using the WebLogic console. This tool comes in handy for administrators and infrastructure teams who can use it to integrate with Ant or Perl scripts, thereby automating the build process.

References

- java weblogic.admin.help
- http://e.docs.bea.com/wls/docs70/admin guide/clihtml 53594

Listing 1

```
    <Server Name="WLDJServer">
        <COM Name="WLDJServer"/>
        <IIOP Name="WLDJServer"/>
        <JTAMigratableTarget Cluster=""
Name="WLDJServer" UserPreferredServer="WLDJServer"/>
        <KernelDebug Name="WLDJServer"/>
        <Log Name="WLDJServer"/>
        <SSL Name="WLDJServer"/>
        <ServerDebug Name="WLDJServer"/>
        <ServerStart Name="WLDJServer"/>
        <WebServer Name="WLDJServer"/>
    </Server>
```

sam's SOAPBOX

–continued from page 36

There are a number of options you can use to make this verification, including verifying that the read columns, the written columns, the version number, and the timestamp are the same. Each case has its own advantages but I would suggest that Version is probably the most universally applicable and may already be a column in your database. Column TYPE verification is the most expensive but the easiest to implement. To enable these optimizations you need to set the two flags on your bean and then change your update code to make sure that you handle the case when an OptimisticConcurrency-Exception might be thrown from a method that does an update or from an explicit commit statement for your bean-managed transaction.

ReadOnly Concurrency

Finally, there is ReadOnly concurrency, which implies caching between transactions. In this case, the data is only loaded from the database on the following conditions: the first read of the bean, the timeout has expired on the bean, or a programmatic flush of the bean was received. If you want to use ReadOnly beans and still occasionally change them, but don't want the overhead of Optimistic concurrency, you should have two beans that are backed by the same data – one ReadOnly and one normal EJB that can be updated. To programmatically flush your ReadOnly beans, simply cast the EJB home to weblogic.ejb.CachingHome and use the invalidate methods on that interface. For more extensive information on how to use these optimization strategies, please refer to http://edocs.bea.com/wls/docs70/ejb/index.html

Programmatic Clients, Symmetry, and the Humble Ant

A PIECE OF OBSCURE KNOWLEDGE

BY PETER HOLDITCH

Picnicking during my summer holidays with my family, I was a little peeved to find that we had set up camp near an ant hill and some of them had decided to help themselves to elements of our lunch. Just as well, really, that I prefer sausage rolls and pork pies to chocolate buns, I guess. Every time I see an ant, I'm reminded of countless documentaries from my youth, waxing lyrical about how an ant can carry many times its own body weight.

Right now, the evidence is being played out in front of me, so at least you can believe some of what you see on TV. Isn't nature a miraculous thing! So what does this have to do with transactions? Well, very little actually.

Bear with My Ramblings, Won't You? I'm on Holiday!

I thought this month I would come up for air with a review of a piece of obsure knowledge about the WebLogic Server transaction manager implementation. As you probably realize by now, if you want to demarcate a transaction from a piece of Java code, you simply need to obtain a reference to the User Transaction object and start invoking its begin, commit, etc., methods at will. What could be simpler?

What may have escaped your notice (since the specs don't make a very big deal about it) is that you can do this wherever you want – in EJBs configured to use bean-managed transactions, in RMI objects, in startup classes, you name it. These are all server-side places; however, the list goes on…. You can also demarcate transactions from Swing clients or even Java applets. In fact, in an alarming throwback to flower power and Woodstock, you can do it wherever you please, on the server or off it. Think about that for a minute…coordinating a transaction (as you may remember from previous columns) requires transaction logs, which are the linchpin of your business data's integrity. You need to be very paranoid about these – put them on RAID disks, make them available to multiple machines for failover, prevent the intern system administrator from deleting them to save disk space, and so on. Now let's think about clients – potentially applets – not exactly where you'd keep your proverbial family silver, or your actual transaction log. So how can transactions be demarcated there? Visions of MS Internet Explorer (with the Java plug-in) powering over a picnic rug carrying a RAID disk array are spinning through my mind… I need to lie down, I'm overcome. Let's face it, this is technology, not the miracle of nature. There is no clever evolutionary mechanism to fall back on here.

What's Going on Behind This Choking Smoke and These Dizzying Mirrors?

As you may have guessed, the BEA engineers who wrote the transaction manager in WebLogic Server were well aware of the asymmetry between the client and server, and they engineered the infrastructure (they'd get cross if I carried the smoke and mirrors reference forward) to take care of it. If you get a reference to the User Transaction object on a client, the implementation is well aware of the lowly status of the container it's running in. It does all the usual stuff with respect to associating a transaction object to the relevant threads on the client, and propagates the transaction context over the wire when the client makes RMI calls to remote objects in the scope of a transaction. However, at commit time it knows its place. It packages all the information about the transaction and sends it to a server, where the commit processing can proceed in an environment more conducive to reliability and where the RAID disks and suchlike are kept. The server processes the commitment of the transaction, and when that's done it sends the result back to the client (if the client is still around to receive it – remember, it's none too reliable out there!). From the programmers' point of view, all this was hidden; they just gaily used the

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

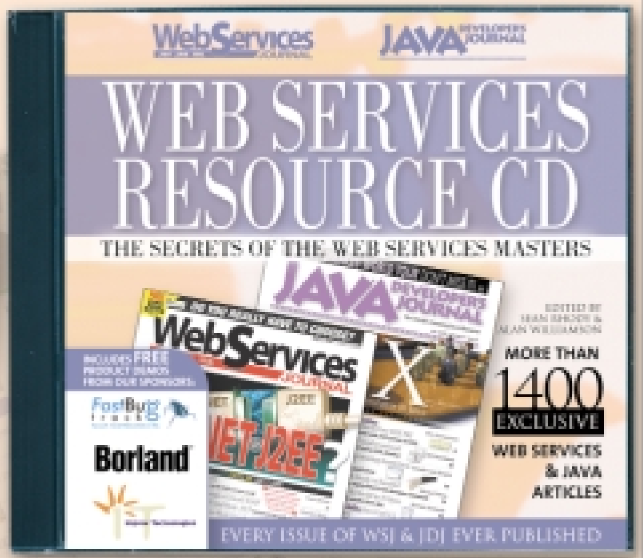
WebServices JOURNAL .NET J2EE XML JAVA DEVELOPER'S JOURNAL

WEB SERVICES RESOURCE CD

THE SECRETS OF THE WEB SERVICES MASTERS

$99 + S&H

Special Limited-time Price



Now Shipping

$119 CD VALUE FROM JDJ

EVERY ISSUE OF WSJ & JDJ EVER PUBLISHED

MORE THAN 1400 EXCLUSIVE WEB SERVICES & JAVA ARTICLES

7 YEARS 83 ISSUES 1400+ ARTICLES ONE CD

THE MOST COMPLETE LIBRARY OF EXCLUSIVE WSJ & JDJ ARTICLES ON ONE CD!

"The Secrets of the Web Services Masters"
CD is edited by well-known editors-in-chief Sean Rhody and Alan Williamson and organized into more than 50 chapters containing more than 1,400 exclusive *WSJ* & *JDJ* articles.

www.JDJSTORE.com

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

ORDER ONLINE ATJDJSTORE.COM SAVE $20

objects they knew and loved and the stuff they coded came to pass.

That's fine and dandy, but there must be some side effects, right? Well yes, from an administrative point of view there are. Relax, they're not too serious, but I'll tell you about them just in case you ever meet them in a dark alley somewhere.

Let's look at an example of a client invoking methods on three objects, A, B, and C (in that order), in a single transaction. Assume that these objects all run in different instances of the application server (clearly a poor design on all counts, but this is a Discovery Channel–style contrived example). At commit time, it's entirely possible that the three application servers have never directly communicated with one another. Certainly all the servers know of the existence of the transaction since it has touched them, but server A thinks it's the only participant – after all, it was the only participant the last time it was called. Only the client knows it has the full picture of the transaction. So the client chooses a server to handle the commit.

The current WebLogic Server implementation chooses the first server that was touched, A in this case, but as with all implementation details, that could change. Don't write code that relies on this behavior. Server A is sent the transaction object so now it, too, has the full picture of what the transaction is comprised of, and it can process the transaction's commitment. At this point, the commit processing is the same as for a transaction started within the application server; server A needs to tell servers B and C about the impending commitment. As you may already know from a previous article, the servers will need to trust one another (if you're not a regular reader, it serves you right – get the back issues!). There is, however, one subtle difference. Remember, servers A, B, and C may never have connected to each other before this moment, in which case they'll have to connect now. It's at this point that you'll get weird commitment failures (you guessed it, your old favorite HeuristicMixedException) if the servers can't contact each other. Whoever configured the TCP/IP network you're using better have allowed for this. Note that this is a static configuration error that can be caught only at runtime – a very good place for a timely reminder that you

"As you may have guessed, the BEA engineers who wrote the transaction manager in WebLogic Server were well aware of the asymmetry between the client and server"

should do preproduction testing of your applications in as close a facsimile of the production system as you can get, even down to the network.

• • •

So, that's more WebLogic Server arcana unearthed for you. I'm back off to the beach now, just as soon as I catch up with that pork pie that's receding over the hilltop!

THE INSIDER INTELLIGENCE YOU NEED...

JAVA Industry Newsletter
WebServices Industry Newsletter
XML JOURNAL Industry Newsletter
wireless Industry Newsletter
WebLogic Industry Newsletter
WebSphere Industry Newsletter
COLDFUSION Industry Newsletter
.NET Journal Industry Newsletter

SELECT THE INDUSTRY NEWSLETTERS THAT MATCH YOUR NEEDS! **CHOOSE ONE OR TRY THEM ALL!**

FREE E-Newsletters SIGN UP TODAY!

TO KEEP AHEAD OF THE CURVE

Go to www.SYS-CON.com

The most innovative products, new releases, interviews, industry developments, and plenty of solid *i*-technology news can be found in SYS-CON Media's Industry Newsletters. Targeted to meet your professional needs, each e-mail is informative, insightful, and to the point. They're free, and your subscription is just a mouse-click away at www.sys-con.com.

Exclusively from the World's Leading *i*-Technology Publisher SYS-CON MEDIA

TO REGISTER: www.sys-con.com or Call 201 802-3069

$195 FOR SYS-CON SUBSCRIBERS

BEST EDUCATIONAL VALUE OF THE YEAR!

web services EDGE world tour 2002

Take Your Career to the Next Level!

EACH CITY WILL BE SPONSORED BY A LEADING WEB SERVICES COMPANY

SHARPEN YOUR PROFESSIONAL SKILLS.

KEEP UP WITH THE TECHNOLOGY EVOLUTION!

"Presented an excellent overview of Web services. Great inside knowledge of both the new and old protocols. Great insight into the code piece."
– Rodrigo Frontecilla

"Very articulate on the Web services SOAP topic and well-prepared for many questions. I've learned a lot from this seminar and I appreciate this seminar for my job. Thank you!"
– Kenneth Unpingco, Southern Wine & Spirits of America

"I liked the overview of Web services and the use of specific tools to display ways to distribute Web services. Good for getting up to speed on the concepts."
– B. Ashton, Stopjetlag.com

Echoed over and over by Web Services Edge World Tour Attendees:

"Good balance of theory and demonstration."

"Excellent scope and depth for my background at this time. Use of examples was good."

"It was tailored toward my needs as a novice to SOAP Web services – and they explained everything."

Learn How to Create, Test and Deploy Enterprise-Class Web Services Applications

TAUGHT BY THE INNOVATORS AND THOUGHT LEADERS IN WEB SERVICES

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

IF YOU MISSED THESE...

BOSTON, MA (Boston Marriott Newton) SOLD OUT!
WASHINGTON, DC (Tysons Corner Marriott) SOLD OUT!
NEW YORK, NY (Doubletree Guest Suites) SOLD OUT!
SAN FRANCISCO, CA (Marriott San Francisco) SOLD OUT! CLASSES ADDED

BE SURE NOT TO MISS THESE...

...COMING TO A CITY NEAR YOU

2002	
LOS ANGELES	NOVEMBER 5
NEW YORK	NOVEMBER 18
SAN FRANCISCO	DECEMBER 3
BOSTON	DECEMBER 12
2003	
CHARLOTTE	JANUARY 7
MIAMI	JANUARY 14
DALLAS	FEBRUARY 4
BALTIMORE	FEBRUARY 20
BOSTON	MARCH 11
CHICAGO	APRIL 16
ATLANTA	MAY 13
MINNEAPOLIS	JUNE 10

REGISTER WITH A COLLEAGUE AND SAVE 15% OFF THE LOWEST REGISTRATION FEE.

WHO SHOULD ATTEND:

- Architects
- Developers
- Programmers
- IS/IT Managers
- C-Level Executives
- *i*-Technology Professionals

SPONSOR A CITY!

Position your company as a leader in Web services

Call 201 802.3066 to discuss how

PRESENT YOUR COMPANY'S EXPERTS TO AN EAGER AUDIENCE READY TO LEARN FROM YOU! ACT TODAY!

TOPICS HAVE INCLUDED: Developing SOAP Web Services
Architecting J2EE Web Services

The San Francisco tutorial drew a record 601 registrations.

SYS-CON *i*-TECHNOLOGY EDUCATION

REGISTRATION FOR EACH CITY CLOSES THREE BUSINESS DAYS BEFORE EACH TUTORIAL DATE. DON'T DELAY. SEATING IS LIMITED. **NON-SUBSCRIBERS:** REGISTER FOR $245 AND RECEIVE THREE FREE ONE-YEAR SUBSCRIPTIONS TO *WEB SERVICES JOURNAL*, *JAVA DEVELOPER'S JOURNAL*, AND *XML-JOURNAL*, PLUS YOUR CHOICE OF *BEA WEBLOGIC DEVELOPER'S JOURNAL* OR *WEBSPHERE DEVELOPER'S JOURNAL*, A $345 VALUE!

TO REGISTER: www.sys-con.com or Call 201 802-3069

In last month's article, I provided an in-depth description of the Advisor Framework that is embedded in WebLogic Portal. This month, I'll focus on describing the components, capabilities, and extensions of the Event Framework. Next month I'll look at the Portal Framework

Understanding the Event Framework

DEMYSTIFYING THE BEA WEBLOGIC PORTAL FRAMEWORK

BY DWIGHT MAMANTEO

AUTHOR BIO...

Dwight Mamanteo is a technical manager with the Global Alliances Technical Services organization at BEA Systems. He has been with BEA since 1999; his current responsibilities include providing advisory and technical enablement support to BEA's strategic SI and ISV partners. He has been involved with object-oriented programming, design, and architecture since 1993.

CONTACT...

dwight@bea.com

The Event Framework is the run-time engine and framework in WebLogic Portal that provides the capability to enable Web sites to react intelligently to user interaction, to capture business relevant interaction information, and to add customized business event agents. Incorporated into the framework are the EventHandlers, EventListeners, EventService, Events, and custom JSP tags.

Business Scenarios

The main purpose in using the Event Framework is to provide business value through the intelligent use of captured user interaction information. Some of the benefits of user interaction information include initiating business-defined campaigns as the user traverses the Web site, and analyzing user interaction behavior. Each is meant to help the business understand its users and to increase the potential of online revenue.

Event Framework

The Event Framework is a configurable framework that implements J2EE design patterns, and allows the easy incorporation of custom components via XML configuration files. The main components in the Event Framework architecture are the EventService EJB, EventHandler types, EventListener types, asynchronous delivery mechanism, Event types, and JSP tag library (see Figure 1).

At a high level, the event JSP tags or a servlet life cycle event would invoke the EventService with an Event notification; the EventService would then dispatch the Event to the EventHandler; the EventHandler would request the registered EventListeners to handle the Event; the EventListeners would perform some business processing and may request additional help from the EventProcessors; and finally, the EventProcessors may perform additional processing, including calling services and frameworks that are part of WebLogic Portal (see Figure 2).

EVENTSERVICE

The EventService is a Stateless Session EJB that implements the "Session Facade" J2EE design pattern. The main responsibility for the EventService is to encapsulate the business processing required to implement the behavior requested by an event JSP tag call or a servlet lifecycle event. As shown in Figure 2, the EventService receives the Event from the Event JSP Tag, and requests the synchronous and asynchronous EventHandlers to dispatch the Event to the registered synchronous and asynchronous Event Listeners. No return value is returned during and after the dispatching of each event type, so exceptions will need to be thrown and caught in order to handle any business processing errors.

STANDARD EVENTS

Events are Java classes that implement the "State" J2EE design pattern. The main responsibility for the Event object is to encapsulate the event information that is specific to each event type. The different types of Events that are part of WebLogic Portal include Session, User Registration, Product, Content, Cart, Buy, Rules, and Campaign Events (see Figure 3). At a minimum, each of the provided events contain information that describe the name of the application that generated the event, the time of the event, the type of the event, the Session ID, the User ID, and any other Event type-specific information.

STANDARD EVENTHANDLERS

EventHandlers are Java classes that implement the "Subject" participant of the "Observer" J2EE design pattern. The main responsibilities for the EventHandlers (synchronous and asynchronous) are to manage the registration of all the EventListeners and to dispatch Events to each registered EventListener. The EventHandler class manages events that are to be delivered synchronously to a registered EventListener, while the AsynchronousEventHandler manages events that are to be delivered asynchronously to a registered asynchronous EventListener.

STANDARD EVENTLISTENERS

EventListeners are Java classes that implement the "Observer" participant of the "Observer" J2EE design pattern. The main responsibility for the EventListeners is to execute the appropriate business processing for any events that they are responsible for handling. EventListeners can either be synchronous or asynchronous. Synchronous EventListeners are registered with synchronous EventHandlers, while asynchronous EventListeners are registered with asynchronous EventHandlers. The different types of event listeners that come with the product include the BehaviorTrackingListener, DebugEventListener, CampaignEventListener, AsynchronousCampaignEventListener, and SessionEventListener (see Figure 4).

CUSTOM EVENTS AND EVENTLISTENERS

Because the Event Framework was designed to be extendable, hooks are provided for customers who wish to add their own custom Event and EventListener classes. For example, a custom event can be created to determine how often a quote tab is selected on a trading portal. Capturing this information would provide value, as the business can decide to improve the "stickiness" factor of their Web site by combining frequently used functionality onto one page or grouping commonly used functionality together.

To write a custom Event, a developer would have to extend the Event class (see Figure 3); add an event type attribute; add event attributes to be passed along to the event listener; and provide a constructor that takes in the event attributes in the argument, passes the event type to the Event class, and adds the Event attributes to the attribute list. The Event class that is being extended provides helper methods to retrieve the event's timestamp and type, and provides setter and getter methods for the event's custom attributes.

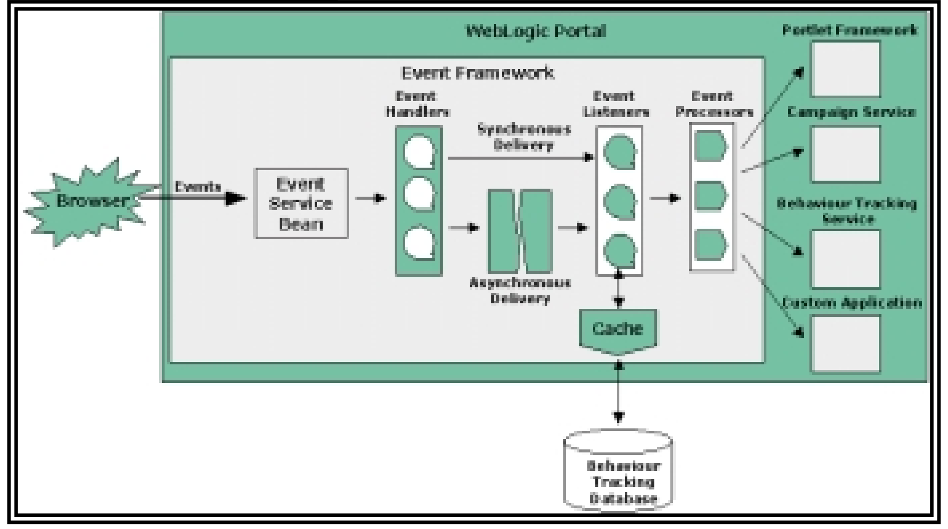
To write a custom EventListener, a developer would have to implement the EventListener interface (see Figure 4), provide a default constructor, implement the getTypes and handleEvent methods, and provide a list of Events that the EventListener is expected to react to. The Event Framework also provides the ability to dynamically determine, during the deployment and runtime stages, the events an event listener should respond to. This dynamic linkage between the many different types of Events and EventListeners provides the ability to adapt the application, via configuration changes, to meet the changing needs of the business.

After compiling the custom Event and EventListener classes, the developer should make sure to place the resulting files in the enterprise application classpath. This will insure that the custom Events and EventListeners are made available to all the Web applications in the enterprise application. More information about creating and registering custom Events and EventListeners can be found on the BEA Systems online documentation site for BEA WebLogic Portal (http://edocs.bea.com/wlp/docs70/dev/evnttrak.htm#998994).

JSP Tags

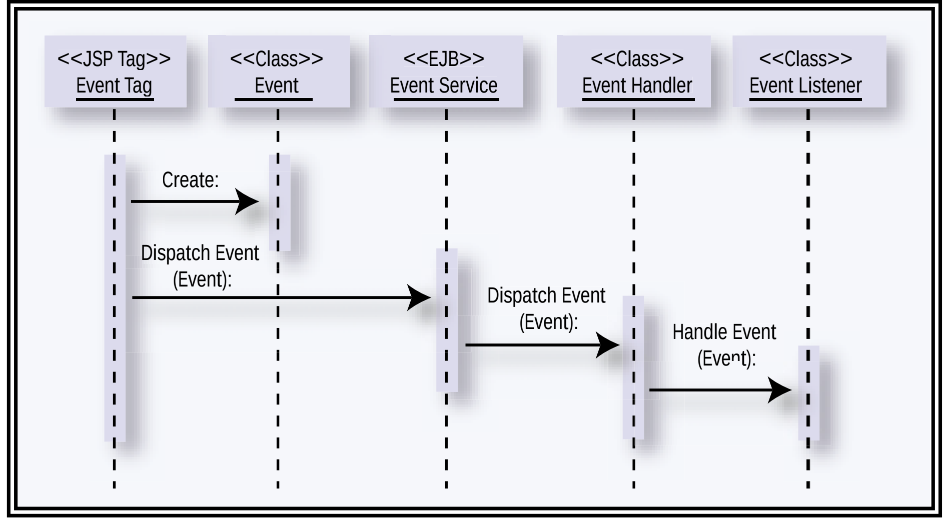
The Event Framework works in conjunction with events that are initiated during the manipulation of the Web site by the end-user. To help with the development of JSP pages that make use of the Event Framework, BEA has included custom JSP tags that instantiate Events and invoke the Event Framework. These JSP tags can be used to track user behavior, and to start promotions and campaigns.

FIGURE 1



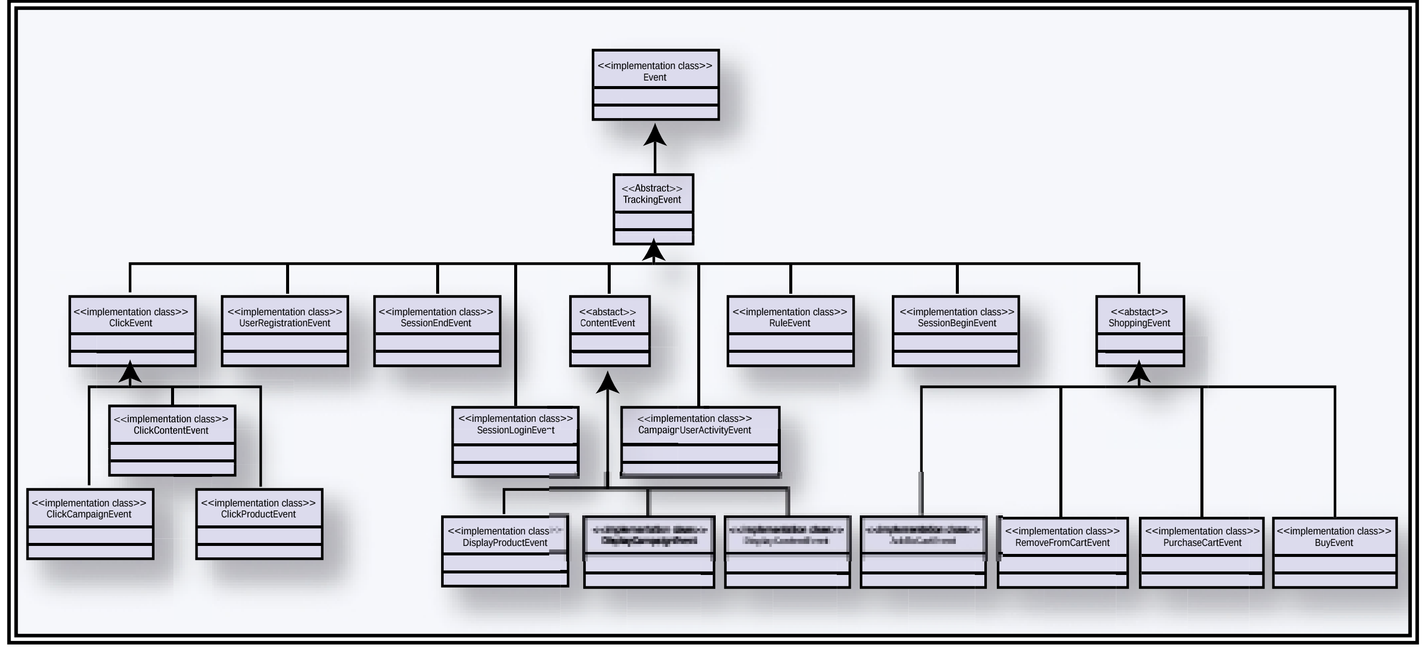
High-level Event Framework architecture

FIGURE 2



High-level Event Framework sequence diagram

FIGURE 3



<<implementation class>> Event
<<Abstract>> TrackingEvent
<<implementation class>> ClickEvent
<<implementation class>> UserRegistrationEvent
<<implementation class>> SessionEndEvent
<<abstact>> ContentEvent
<<implementation class>> RuleEvent
<<implementation class>> SessionBeginEvent
<<abstact>> ShoppingEvent
<<implementation class>> ClickContentEvent
<<implementation class>> ClickCampaignEvent
<<implementation class>> ClickProductEvent
<<implementation class>> SessionLoginEvent
<<implementation class>> CampaignUserActivityEvent
<<implementation class>> DisplayProductEvent
<<implementation class>> RemoveFromCartEvent
<<implementation class>> PurchaseCartEvent
<<implementation class>> BuyEvent

Event class diagram

<tr:clickCONTENTEVENT>

The <tr:clickContentEvent> JSP tag will generate a ClickContentEvent when the user clicks on an ad impression. This tag will return a URL query string that contains event parameters, which can be used to form a complete URL. This would mean that there is a two-step process when adding an ad click event JSP tag to a JSP page.

As shown below, the first step to adding an ad click event would be to execute the JSP tag in order to create the URL query string.

```
<%@ taglibs URI=" tracking.tld" prefix="tr" %>
.
.
.

<tr:clickContentEvent
id="urlQuery"
documentId="<%=documentId %>"
documentType="<%=documentType %>"
userId="<%=request.getRemoteUser() %>"
/>
```

The second step would be to add the URL query string to the hyperlink that would execute the event. The example below adds the URL query string to the hyperlink.

```
<% finalURL = "www.bea.com/specials" + "&" +
urlQuery; %>
```

```
<A HREF="<%= finalURL %>">
```

The "id" tag attribute contains the URL query string that is returned by the custom JSP tag and is used during the construction of the final URL string.

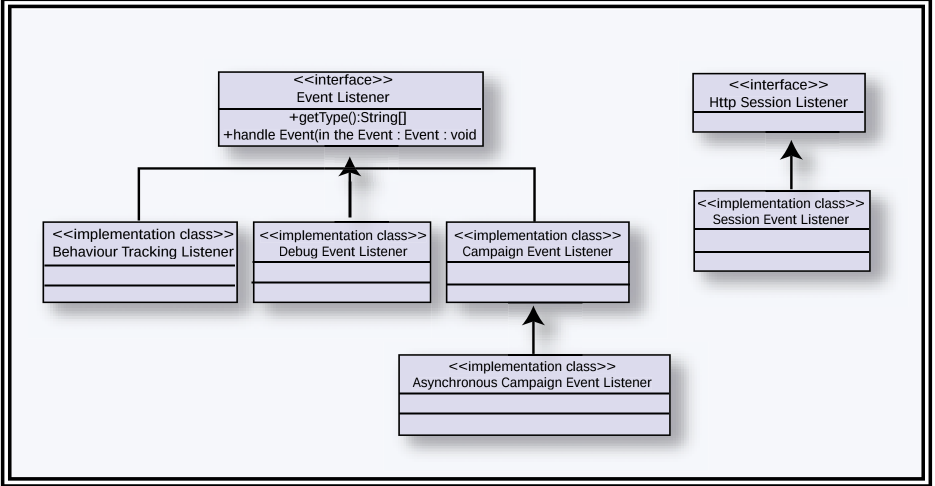
<tr:displayCONTENTEVENT>

The <tr:displayContentEvent> JSP tag will generate a DisplayContentEvent event when an ad impression is displayed to the end-user. As shown below, the developer would simply include the custom JSP tag when displaying the content to the end-user.

```
<%@ taglibs URI=" tracking.tld" prefix="tr" %>
<%@ taglibs URI=" es.tld" prefix="es" %>

.
.
.
<es:forEachInArray
id="nextRow"
   array="<%=ads %>"
   type="com.bea.p13n.content.Content">
.
.
.
         <tr:displayContentEvent
documentId="<%=documentId %>"
```

FIGURE 4

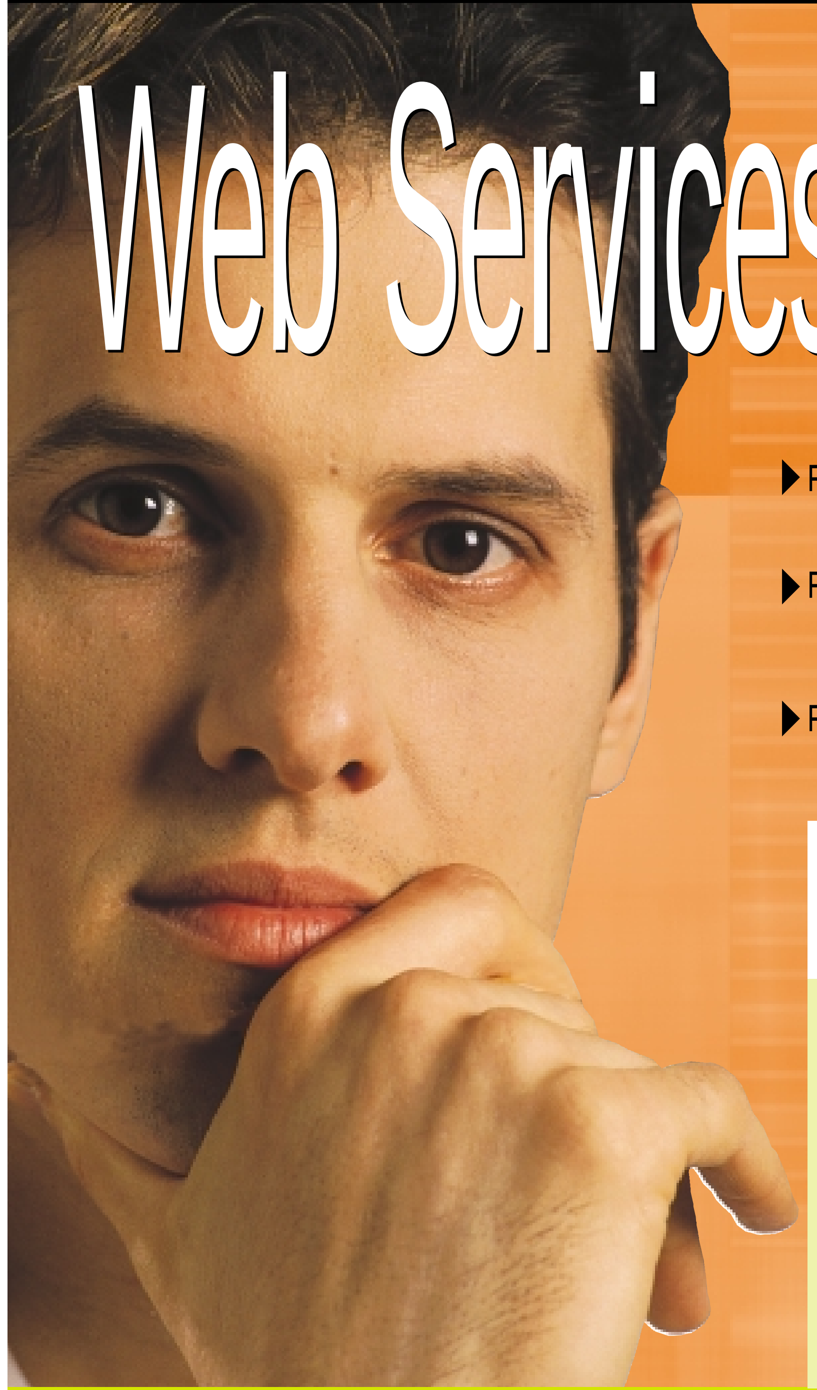


<<interface>> Event Listener
+getType():String[]
+handle Event(in the Event : Event : void
<<interface>> Http Session Listener
<<implementation class>> Behaviour Tracking Listener
<<implementation class>> Debug Event Listener
<<implementation class>> Campaign Event Listener
<<implementation class>> Session Event Listener
<<implementation class>> Asynchronous Campaign Event Listener

EVENT LISTENER CLASS DIAGRAM

Visit www.sys-con.com for more information

Web Services Edge 2003



Focus on Web Services
Companies that get an early jump on Web Services will be winners in today's challenging landscape. Get ready to take your early pilot projects to the next level!

Focus on Java, XML, and .NET
Hear from the leading minds in Java how this essential technology offers robust solutions to i-technology. Make the right choices. Explore the evolving world of standards, interoperability, application integration, security, and more as you build a collaborative enterprise.

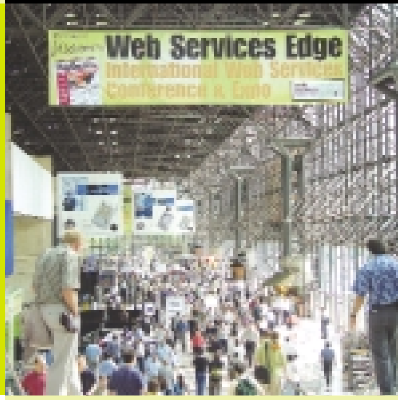
Focus on the Knowledge You Need
Immerse yourself in information-packed conference sessions. Meet today's i-technology leaders, and gather resources in an action-packed Expo!

web services EDGE conference&expo

The Largest Web Services, Java, XML, and .NET Conference and Expo!

Boston
New York
London
Berlin
Hong Kong
03

THE NEXT FRONTIER: BEYOND THE FIREWALL



OWNED BY SYS-CON MEDIA
PRODUCED BY SYS-CON EVENTS
JAVA DEVELOPERS JOURNAL · WebServices JOURNAL · XML-JOURNAL · WebLogic DEVELOPERS JOURNAL · COLDFUSION Developer's Journal · PowerBuilder Developer's Journal · wireless BUSINESS&TECHNOLOGY · .NET JOURNAL · WebSphere DEVELOPER'S JOURNAL · LINUX BUSINESS WEEK · CF Advisor

Call for Papers Opens October 15, 2002
201 802-3069

SUBSCRIBE TO THE FINEST TECHNICAL JOURNALS IN THE WORLD...

IT'S JUST A CLICK AWAY!

SYS-CON MEDIA

www.SYS-CON.com

```
documentType="<%=documentType %>"
/>
.
.
.
</es:forEachInArray>
```

<tr:clickPRODUCTEVENT>

The <tr:clickProductEvent> JSP tag will generate a ClickProductEvent event when the user clicks on a product impression. This tag will return a URL query string that contains event parameters, which can be used to form a complete URL. As with the <tr:clickContentEvent> custom JSP tag, the <tr:clickProductEvent> tag also employs a two-step process when adding a product click event JSP tag to a JSP page.

As shown below, the first step to adding a product click event would be to execute the JSP tag in order to create the URL query string.

```
<%@ taglibs URI=" productTracking.tld"
prefix="trp" %>
.
.
.

<trp:clickProductEvent
id="urlQuery"
documentId="<%=productId %>"
sku="<%=productSKU %>"
userId="<%=request.getRemoteUser()%>"
/>
```

The second step would be to add the URL query string to the hyperlink that would execute the event. The example below does just that:

```
<% finalURL = "www.bea.com/productDetails/" + "&"
+ urlQuery; %>

<A HREF="<%= finalURL %>">
```

The "id" tag attribute contains the URL query string that is returned by the custom JSP tag and is used during the construction of the final URL string.

<tr:displayPRODUCTEVENT>

The <tr:displayProductEvent> JSP tag will generate a DisplayProductEvent event when a product impression is displayed to the end-user. As shown below, the developer would simply include the custom JSP tag when displaying the content to the end-user.

```
<%@ taglibs URI=" productTracking.tld"
prefix="trp" %>
<%@ taglibs URI=" es.tld" prefix="es" %>

.
.
.
<es:forEachInArray
id="nextRow"
   array="<%=ads %>"
   type="com.bea.p13n.content.Content">
.
.
.
         <trp:displayProductEvent
documentId="<%=productId %>"
documentType="<%=documentType %>"
sku="<%=productSKU %>"
/>
.
.
.
</es:forEachInArray>
```

Conclusion

The Event Framework enables the creation of applications that track customer behavior and dynamically execute business campaigns. The out-of-the-box Event Framework components include event JSP tags, a runtime engine, and an extendible interface that allows the easy inclusion of custom Event and EventListener components.

The Event Framework enables the creation of applications that track customer behavior and dynamically execute business campaigns

The business value gained from capturing user interaction information ranges from enabling the system to intelligently react to customer interaction in order to increase the potential of a sale to providing valuable information that can be used to identify how to increase a portal site's usability and user satisfaction. The Event Framework contained in BEA WebLogic Portal provides the capabilities that enable businesses to achieve greater value from their portal implementations.

Once you're in it...

reprint it...

Wireless Business & Technology
Java Developer's Journal
XML Journal
ColdFusion Developer's Journal
PowerBuilder Developer's Journal

Contact Carrie Gebert
201 802-3026
carrieg@sys-con.com
SYS-CON MEDIA
RePrints

WLDJ ADVERTISER INDEX

ADVERTISER	URL	PHONE	PAGE
Altaworks	www.altaworks.com	603-598-2582	52
BEA Systems	http://dev2dev.bea.com/useworkshop	800-817-4BEA	41
BEA dev2dev Days	www.bea.com/events/dev2devdays	404-240-5506	11
Borland	www.borland.com	831-431-1000	2
Covasoft	www.covasoft.com/today's_tech/	888-963-2682	13
Dirig Software	www.dirig.com/illusion/weblogic	603-889-2777	9
Intel	www.intel.com/ad/bea	408-765-8080	29
PANACYA	www.panacya.com	877-726-2292 x3344	3
Performant	www.performant.com/weblogic1	866-773-6268	37
Precise Software Solutions	www.precise.com/wldj	800-310-4777	15
Rational Software	www.rational.com/offer/bea	800-728-1212	19
Sitraka	www.sitraka.com/jclass/wldj	800-663-4723	17
Sitraka	www.sitraka.com/performance/dev2dev	800-663-4723	25
Sitraka	www.sitraka.com/performance/wldj	800-663-4723	51
SYS-CON Publications	www.sys-con.com	201-802-3020	31
SYS-CON Reprints	www.sys-con.com	201-802-3026	47
Web Services Edge World Tour	www.sys-con.com	201-802-3069	45
Web Services Journal	www.wsj2.com	888-303-5282	49
Web Services Resource CD	www.jdjstore.com	201-802-3012	27
WebLogic Developer's Journal	www.weblogicdevelopersjournal.com	888-303-5282	33
Wily Technology	www.wilytech.com	888-GET-WILY	4

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

LOOK WHAT'S COMING NEXT MONTH

The Role of Web Services in Enterprise Application Architecture
by Frank Sinton
A look at the vision for Web ervices within complex corporations, how it compares and fits versus EAI, and how to begin implementing Web services your business.

Are Your Handhelds Safe in J2EE Hands ?
by Nick Maiorano
Software engineers are realizing that out-of-the-box components cannot be used to solve problems for which they were not designed. We look at why traditional techniques don't work and how you can create your own scalable, fault-tolerant SSM solution.

Understanding the Portal Framework
by Dwight Mamanteo
In the third article in this series Dwight looks at the components, capabilities, and extensions of the Portal Framework.

Transactions: the Lingua Franca of Computers...
by Peter Holditch
In this month's column, Peter looks at the connection between transactions and Web services.

Plus,
BEA's dev2dev Days
A look back at a successful series of events: who went, what was covered, and what's coming on dev2dev.com

Editor-in-chief Jason Westra looks at grid computing -- and its relationship to football.

BEA WebLogic DEVELOPER'S JOURNAL

News & Developments

BEA Systems and HP Extend Relationship

(Los Angeles) – BEA Systems, Inc., the world's leading application infrastructure software company, and HP have announced two new offerings that will assist customers seeking to develop and manage their business in real time.

A joint initiative will make a six-month trial version of BEA WebLogic Server 7.0 Advantage Edition available to HP-UX 11i customers. At the same time, HP has announced the expansion of its portfolio of HP OpenView software solutions for managing and optimizing complex business services, providing customers with simplified and ongoing management of real-time transactions.

The new HP OpenView Transaction Analyzer uses transaction management application program interfaces, codeveloped with BEA, to monitor WebLogic Server 6.X and 7.0. It is the industry's first solution to provide tag-and-trace capability that automatically troubleshoots performance bottlenecks in both J2EE and Microsoft DNA application infrastructures by analyzing the actual path of all transactions and quickly locating network, server or application troubles. www.bea.com, www.hp.com

TogetherSoft and BEA Expand Development Choices

(Raleigh, NC and San Jose, CA) – TogetherSoft Corporation and BEA Systems, Inc., have announced the availability of Together ControlCenter Accelerator Plug-in for BEA WebLogic Workshop. The new offering integrates TogetherSoft's best-of-breed visual Java modeling tool into BEA's visual Web services development framework, giving developers a uniquely productive environment for designing, developing, deploying, and testing J2EE applications. Through TogetherSoft's integration, the two companies will offer more choices to developers and enterprise customers. www.togethersoft.com

BEA Signs Worldwide Partnership With Schlumberger

(San Jose, CA) – BEA Systems, Inc. has announced a strategic partnership with Schlumberger, a global technology services company, to provide innovative and business-critical systems built on the BEA WebLogic Enterprise Platform. Under the agreement, SchlumbergerSema, the IT business arm of Schlumberger, will standardize the development of its solutions on the BEA platform, and the two companies will jointly develop and promote a worldwide systems integration offering.

The BEA platform will help customers simplify the flow of information; decrease the cost of managing applications; and become more agile, productive, and connected. www.slb.com

Communix Launches Rapid Provisioning Solution

(London) – Communix Limited, a value-added reseller of e-business infrastructure products and services, has launched AppSynergy. The new solution reduces the cost and time it takes to procure, install, and configure best-of-breed and leading-vendor J2EE application server software on branded workstation and server class computers for development, test, and production environments. www.communix.com

Appfluent Technology Provides New Functionality to BEA WebLogic

(Arlington, VA) – Appfluent Technology, provider of software to improve enterprise application performance, has announced that its Appfluent SQL Monitor for BEA WebLogic is available for download, free of charge.

Designed specifically for WebLogic, the plug-in integrates with the console and allows users to view and examine every SQL query executed against an Oracle database. It provides insight into database activity that affects application performance, and generates recommendations to eliminate inefficiencies.

For a free download of Appfluent SQL Monitor for BEA WebLogic go to www.appfluent.com or http://dev2dev.bea.com/resourcelibrary/utilitiestools/monitoring.jsp?highlight =utilitiestools.

BEA and Sun to Provide Joint Support Center

(San Francisco) – BEA Systems, Inc., and Sun Microsystems, Inc., have announced a joint support center to support BEA and Sun customers. This relationship will be designed to speed time to resolution and increase application availability for scalable, Internet-ready solutions.

This seamless support relationship can also provide lab environments worldwide for testing interoperability between BEA WebLogic and Sun's Java Virtual Machine software and the Solaris Operating Environment. With this new support model, BEA and Sun can provide enhancements to the established Sun Vendor Integration (SunVIP) Program and help provide faster response times to customer related issues, minimizing system downtime and achieving full interoperability.

The Joint Support Center will be available to customers with Sun's SunSpectrum Platinum and SunSpectrum Gold contracts and with BEA's Mission Critical and Production Support contracts. http://sunnetwork.sun.com.

Precise Automates Application Performance Management

(Westwood, MA) – Precise Software Solutions has announced an industry breakthrough that enhances customer productivity by reducing application performance problem resolution time to a fraction of the norm. Precise i3 version 6.0 combines Precise's unique, in-depth data collection technology with its TotalCorrelation and SmarTune technologies to resolve application performance slowdowns before they damage business performance. This solution reduces problem resolution time to just minutes, compared to the 26-hour average. www.precise.com.

Sitraka

www.sitraka.com/jclass/wldj

Altaworks

www.altaworks.com