

ARE YOUR APPLICATIONS SECURE?

BEA WebLogic

DEVELOPER'S JOURNAL

FEBRUARY 2003 - Volume:2 Issue:2

weblogicdevelopersjournal.com

FROM THE EDITOR
WebLogic Application
Security
by Jason Westra
page 5

SECURITY
JAAS
Fundamentals
by Bill Kemp & Rich Helton
page 6

FROM THE OFFICE
OF THE CTO
Importance of
Application
Architecture
by Gordon Simpson
page 36

NEWS &
DEVELOPMENTS
page 50

RETAILERS PLEASE DISPLAY
UNTIL MARCH 31, 2003

\$8.99US \$9.99CAN



SYS-CON
MEDIA

INTERNATIONAL JAVA DEVELOPER CONFERENCE & EXPO
REGISTER BY FEB. 28 AND SAVE UP TO \$350!

JDJ EDGE
conference & expo

Sun microsystems Java™
University Program
March 18-20, 2003
Boston, MA
Hynes Convention Center

See details on page 41

WEB SERVICES EDGE • JDJEDGE • XMLEDGE

SECURITY: **Writing A Custom JAAS LoginModule To Support Secure DataBase Authentication** When the third-party providers don't work, develop your own security



14

Tim Pijpops

SECURITY: **WebLogic Security Framework**
Working with your security ecosystem



18

Paul Patrick & Vadim Rosenberg

FEATURE: **Maximizing Performance, Availability, and Security of BEA WebLogic Clusters**
Using Cisco content switching technology to maximize end-user experience



28

Brian Walck & Vadim Rosenberg

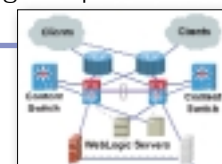
PRODUCT REVIEW: **Component-Level Performance Monitoring with Dirig PathFinder** Dirig simplifies Web application performance management



32

Ralph Decker

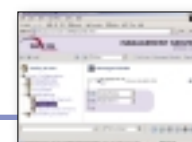
TRANSACTION MANAGEMENT: **Sending Messages to the Other Side** Enabling integration between WebLogic Server and MQ



34

Peter Holditch

WEB SERVICES: **Using SOAP Message Handlers with WebLogic 7** A standard, flexible system that can save you time



38

Michael Gilbode

Sitraka (now part of Quest Software)
www.sitraka.com/performance/wldj

BEA Systems
<http://dev2dev.bea.com/useworkshop>

Wily Technology
www.wilytech.com



EDITORIAL ADVISORY BOARD

LEWIS CIRNE, SHAUN CONNOLLY, STUART HALLOWAY,
KEVIN JONES, TYLER JEWELL, WAYNE LESLEY LUND,
SEAN RHODY, BLAKE STONE

FOUNDING EDITOR

PETER ZADROZNY

EDITOR-IN-CHIEF

JASON WESTRA

EDITORIAL DIRECTOR

JEREMY GEELAN

EXECUTIVE EDITOR

GAIL SCHULTZ

EDITOR

NANCY VALENTINE

ASSOCIATE EDITORS

JAMIE MATUSOW, JEAN CASSIDY

ASSISTANT EDITOR

JENNIFER STILLEY

WRITERS IN THIS ISSUE

RALPH DECKER, MICHAEL GILBODE, RICH HELTON,
PETER HOLDITCH, BILL KEMP, PAUL PATRICK,
TIM PIPOPS, VADIM ROSENBERG,
GORDON SIMPSON, BRIAN WALCK, JASON WESTRA

SUBSCRIPTIONS

For subscriptions and requests for bulk orders,
please send your letters to Subscription Department.

SUBSCRIPTION HOTLINE:

888-303-5282

Cover Price: \$8.99/Issue

Domestic: \$149/YR (12 Issues)

Canada/Mexico: \$169/YR

Overseas: \$179/YR

(U.S. Banks or Money Orders)

PRESIDENT AND CEO

FUAT 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

LOUIS F. CUFFARI • RICHARD SILVERBERG

ASSISTANT ART DIRECTOR

TAMI BEATTY

VP, SALES & MARKETING

MILES SILVERMAN

ADVERTISING SALES DIRECTOR

ROBYN FORMIA

ADVERTISING ACCOUNT MANAGER

MEGAN RING-MUSSA

ASSOCIATE SALES MANAGERS

CARRIE GEBERT • ALISA CATALANO

KRISTIN KUHNLE • LEAH HITTMAN

PRESIDENT, SYS-CON EVENTS

GRISHA DAVIDA

CONFERENCE MANAGER

MICHAEL LYNCH

REGIONAL SALES MANAGERS

MICHAEL PESICK • RICHARD ANDERSON

FINANCIAL ANALYST

JOAN LAROSE

ACCOUNTS RECEIVABLE

KERRI VON ACHEN

ACCOUNTS PAYABLE

BETTY WHITE

VP, INFORMATION SYSTEMS

ROBERT DIAMOND

WEB DESIGNERS

STEPHEN KILMURRAY • CHRISTOPHER CROCE

ONLINE EDITOR

LIN GOETZ

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



BY JASON WESTRA
EDITOR-IN-CHIEF

WebLogic Application Security

My house has bars on its windows. Yes, bars. I am sure at some point in the life of the 110-year-old house, they served a functional purpose. Surely, if I were a robber, I'd be more motivated to look elsewhere for my next DVD player to steal, but the bars are more decorative, just ornamental now. If I were truly concerned about security, I'd get myself a modern home security system with all the bells and whistles like motion detectors and night vision cameras. Technology has really bolstered the security for homes in the past decade.

Similarly, application security has come a long way in the past few years, pushed by a rise in Internet usage and an increase in use of and reliance on corporate systems. Thankfully it has not taken a century to progress like my home's security. Companies such as Entegriety, Netegriety, and Tivoli (now an IBM software division) have done well to push security beyond purely data security and into the Web application and Web services layers.

Application security is something that can never stop progressing. As hacker techniques evolve, you need to constantly reevaluate your application security requirements and controls. How do you stay ahead of the game with regard to security? The build-versus-buy discussion rears its ugly head when you're determining what approach to take. Choosing a vendor for its security infrastructure will help you remain on top of the latest security enhancements with minimal effort from your development resources. However, in my opinion, taking a roll-your-own approach to security isn't bad, if you're only rolling a portion of the security and you roll it right. For instance, don't write your own SSL capabilities! But, if your application has unique requirements, then write a custom secu-

urity domain. Just be sure your solution will be flexible enough to integrate with your middleware correctly.

Security integration is a necessary evil in today's application space as departmental applications are pushed to greater visibility in the enterprise and different vendors' ERP systems are connected to each other or exposed through custom Web applications. I believe the BEA WebLogic Platform has a compelling security infrastructure in its 7.0 release. The platform provides pluggable security domains for NT, Unix, LDAP and RDBMS, as well as the ability to provide your own custom domain. Also, BEA's support for Java standards has enabled a best-of-breed approach toward security infrastructure. Entegriety's (www.entegriety.com) AssureAccess takes advantage of BEA's new implementation of the Java SSPI (security service provider interface) and provides an out-of-the-box custom security realm for BEA WebLogic Platform applications. It also provides a Java API, a JSP tag library for easy script security calls from your JSPs, and a Servlet 2.3-compliant servlet filter for even more complex security control.

This month, **WLDJ** focuses on the BEA WebLogic Platform and security. Our articles include an introduction to JAAS (Java Authentication and Authorization Service), new in JDK 1.4; and how to build your own custom login module for JAAS. Also, we're starting a new column from the "Office of the CTO," this month on the importance of application architecture.

Take a look at the new security model in 7.0 for yourself. It's a compelling solution for securing your enterprise content and services. For me, I think I'll stick to bars on my windows and maybe a "nice" dog - you know, like a pit bull or a German shepherd. 🐕

AUTHOR BIO...

Jason Westra is the editor-in-chief of **WLDJ** and an application architect for a global consulting 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

© COPYRIGHT, 2002 BY SYS-CON PUBLICATIONS, INC. ALL RIGHTS RESERVED. NO PART OF THIS PUBLICATION MAY BE REPRODUCED OR MECHANICAL, ELECTRONIC OR MECHANICAL, INCLUDING PHOTOCOPYING OR ANY INFORMATION STORAGE AND RETRIEVAL SYSTEM, WITHOUT WRITTEN PERMISSION. FOR PROMOTIONAL REPRINTS, CONTACT REPRINT COORDINATOR. SYS-CON PUBLICATIONS, INC. RESERVES THE RIGHT TO REPRODUCE, 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.



BY BILL KEMP & RICH HELTON

AUTHOR BIO...

Bill Kemp is a developer relations engineer with BEA in Colorado Springs, with 14 years of experience in software development and support. He is a Sun Certified Java Programmer and BEA WebLogic Certified Developer. Bill focuses on EJB, JMS, and JDBC.

Rich Helton has worked on computer systems since 1982 and entered the private sector in 1990 as a lead computer architect with OmniPoint Data Corporation. He is the author of *Java Security Solutions*, considered to be one of the most comprehensive security books, and coauthor of the *BEA WebLogic Server Bible*. He has lectured extensively on security and Java and has written articles on the subject. Rich currently works as the chief architect for the Colorado Department of Labor

CONTACT...

wkemp@bea.com
richware@earthlink.net



Security is any mechanism that can be used to protect and validate resources. There are many security models that can be used to protect the data. The security model may use encryption, access control, or several other security methods. Authorization, or access control, has different security services that may be used to protect the resources. One method may be the Java Authentication and Authorization Service (JAAS), and another is Windows 2000 Active Directory. This article focuses on the JAAS security service.

Two of the most basic security mechanisms are authentication and authorization. Authentication is simply the identification of an entity. Authorization is the decision process for granting the entity access rights to any data resources or

information. The JDK 1.4 comes standard with services that provide both authentication and authorization. These services are called the Java Authentication and Authorization Service (JAAS).

JAAS defines authentication mechanisms through a configuration file; code is not needed to define the mechanism. The authentication mechanism requires parameters in order to identify a user. The parameters used to authenticate a user, such as a name and password for each authentication mechanism, are known as a *Principal*. A *Subject* is who I am, but I may use a *Principal*, my driver's license, to authenticate myself to a cop, and use another *Principal*, my passport, to authenticate myself to the border patrol. Authorization will only occur after authentication because users must be identified before they are given access to protected resources. The JAAS framework will wrap authen-

NTALS

AND AUTHORIZATION

Handler, and the LoginModule requires the information, the LoginModule will use the CallbackHandler to call the LoginContext back to get the information required.

The LoginContext may also pass the Subject to the LoginModule for the Principals to use to log in.

After the initialization of the LoginModule when the LoginContext is created, the LoginContext may call the login() method (see Figure 2), which will authenticate the Subject. The login process may go through a daisy chain of LoginModules, each requiring a different type of Principal based on the configuration. The configuration can specify LoginModules that are optional but not required. The LoginModule will complete a two-phase login process. The first phase is when the login() method of the LoginModule will get called from the login process. The second phase is when the commit(), or abort(), method is called to finish the login process. The commit() process is called from the login process when all other LoginModules complete their login() method. Then the last LoginModule () will perform its commit, then the next, and so on. The abort method will perform any cleanup if the login() method did not complete successfully. If all of the required commit() methods completed successfully, then the login process completed successfully.

The authorization process requires the Principals established during the login process, so it is no surprise that the authorization is normally accomplished in the Subject class. The Subject class performs

authorization using a PrivilegedAction class. An extended PrivilegedAction class wraps the resources that are to have the access control. Whether or not the Principal has permission to access a resource is defined in a security policy file. The Java security manager will read the appropriate security policy file and parse the permissions and Principals. The security manager will then grant or deny the resources in the Privileged Action based on these permissions and principals. If the permission for the resource is not allowed for a specific Principal, then an access exception is thrown.

Using JAAS with WebLogic

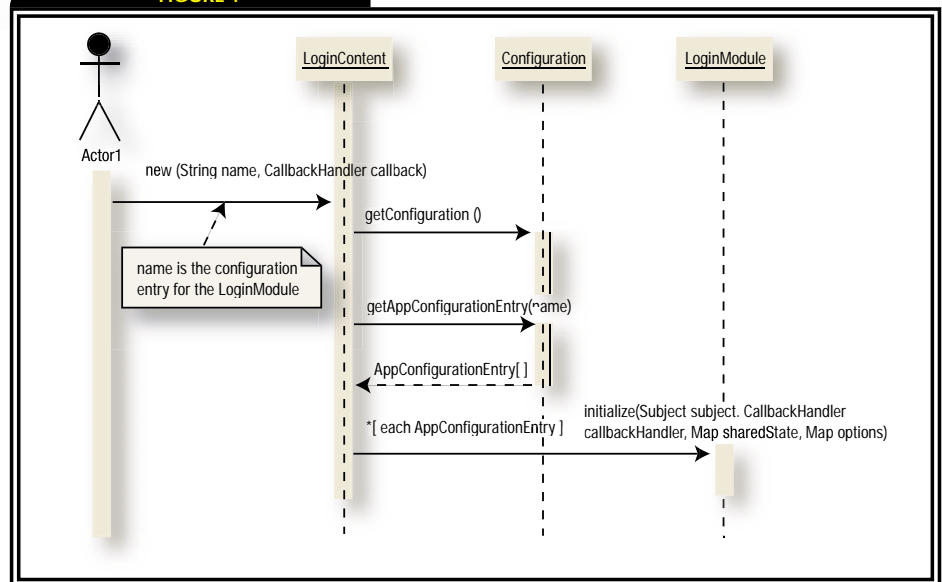
WebLogic Application Server (WLS) 7.0 does not redefine or change JAAS. JAAS remains the same regardless, but it is an extensible framework. WLS 7.0 includes its own authentication and authorization mechanisms that do not require a policy file. The configuration file will define the WLS LoginModule to be used to wrap the WLS authentication mechanism. The WLS LoginModule normally requires a username, password, and URL to point to the appropriate WLS server. WLS 7.0 also contains an authorization mechanism from mapping resources to roles. Because WLS 7.0 has an authorization mechanism implemented in the WebLogic security framework, policy files and the Java security manager do not have to be defined. A weblogic.security.Security class is required when using calling the Privileged Action.

WebLogic 7.0 comes with an example of

tication modules specified by the configuration file. A successful authentication process will return a Subject containing Principals. The Principals returned by the authentication mechanisms will be used for the authorization process.

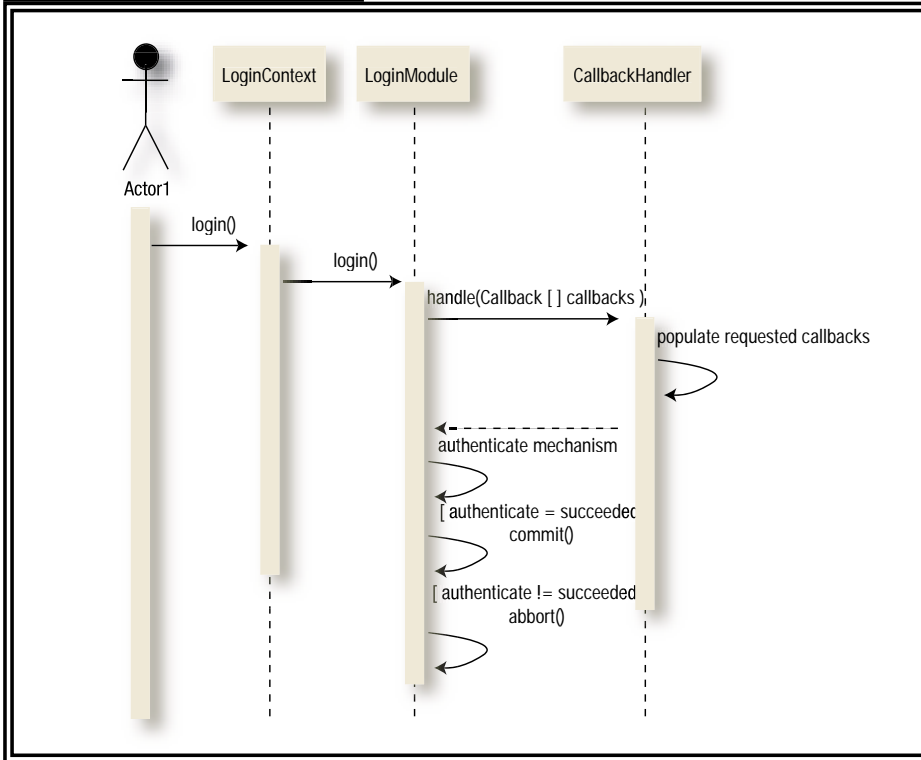
JAAS will start with a LoginContext class to find the entries in the configuration file that will initialize the appropriate LoginModules (see Figure 1). The configuration file will contain any initialization parameters that the LoginContext does not specify. The LoginContext will also pass the LoginModule a CallbackHandler. The CallbackHandler will call back to the application to request any additional authentication information. For example, if a LoginContext did not specify a username and password through the creation of a Callback-

FIGURE 1



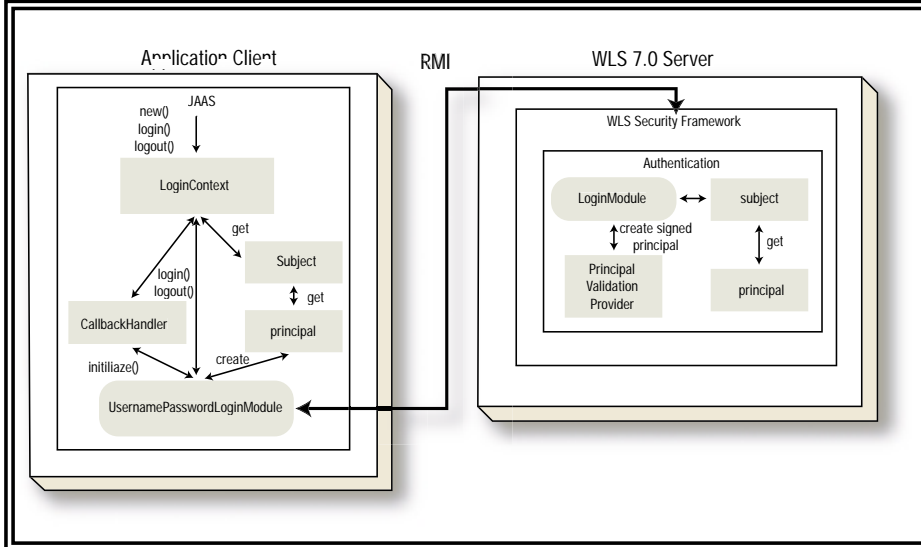
JAAS initialization sequence

FIGURE 2



JAAS login sequence

FIGURE 3



WLS 7.0 Authentication

how to perform JAAS-based authentication and authorization from a Java client application to make calls to an EJB that has access restrictions enabled on its methods. The example is a slight modification of the Sun example found at <http://java.sun.com/j2se/1.4/docs/guide/security/jaas/JAASRefGuide.html#Sample>.

This example includes a JAAS configuration file, which specifies the class name of

the LoginModule used to perform authentication of the EJB client; a CallbackHandler used to gather the client credentials and the URL of the authentication server; a PrivilegedAction, which contains the code that performs the EJB access; and a client application that creates a LoginContext, calls the login method, and invokes the PrivilegedAction through the weblogic.security.Security.runAs method. The flow of

this example application is diagrammed in Figure 3. You can refer to the full example by looking in \$WL_HOME/weblogic700/samples/server/src/examples/security/jaas. The remainder of this article will refer to this example to discuss JAAS-based authentication and authorization in WLS from remote clients and to compare that with server-side components such as servlets.

Java Client Authentication

Authentication of a stand-alone Java client is used by client applications that need to directly access EJBs or JMS Destinations located on a WebLogic Server. The client-side example, supplied by BEA with the WLS 7.0 release, uses a JAAS policy file, sample_jaas-config, to specify a single LoginModule, weblogic.security.auth.login.UsernamePasswordLoginModule, to the application. Here are the contents of this file:

```

Sample {
    weblogic.security.auth.login.UsernamePasswordLoginModule required debug=false;
};
    
```

The code for the UsernamePasswordLoginModule is not included in the example package, but the class file is located in weblogic.jar. Details of the code can be found at http://edocs.bea.com/wls/docs70/security/cli_apps.html#1096287.

It uses the weblogic.security.auth.Authenticate class, a WebLogic-specific class, to perform the actual authentication to a WebLogic server instance that is specified to the LoginModule through a URLCallback. The application's CallbackHandler supplies the URL of the server that will perform the authentication through the server's configured Authentication Provider.

The LoginModule passes a NameCallback and a PasswordCallback to the CallbackHandler to acquire the username and password of the client initiating the application. Once the user is authenticated, the LoginModule populates the Subject passed to it by the LoginContext. The application then retrieves the authenticated Subject using the LoginContext.getSubject method. The Subject holds the WebLogic Principals – which can be a WLSUser or WLSGroup – which will be used for authorization when the Subject is passed to the Security.runAs method. The entire JAAS authentication sequence is initiated by the application when it instantiates a LoginContext and calls the LoginContext.login() method. The LoginContext locates a

Programmer's Paradise

www.programmersparadise.com/BEA

Configuration object that loads the configuration information found in the JAAS policy file. It uses the Configuration to create an instance of the LoginModule that will be used for the authentication sequence.

```
// Create LoginContext; specify username/password login module
LoginContext = new LoginContext("Sample", new
SampleCallbackHandler(username, password, url));
```

This instantiation of the LoginContext will locate and instantiate a LoginModule found in the Configuration, using the name "Sample" to locate the LoginModule in the Configuration, and pass the SampleCallbackHandler to its initialize method. From the entry in the sample_jaas.config file, you can see that the LoginContext will instantiate an instance of UsernamePasswordLoginModule to perform the authentication.

The WebLogic documentation states that the use of the weblogic.jndi.Environment class is deprecated in the WLS 7.0 release. However, the LoginModule supplied with the example uses the weblogic.jndi.Environment object to perform the authentication by passing the Environment, which contains the username, password, and URL of the server, to the Authenticate.authenticate method. This apparent contradiction is a good opportunity to plug the advantages of the pluggable nature of the LoginModule. The mechanism of the example LoginModule, the use of the Authenticate class, is not visible to the client application attempting to authenticate itself to the WebLogic server. Since the LoginModule is an instance of a Pluggable Authentication Module (PAM), it can be replaced or rewritten

without affecting the client application. This is a good example of how the pluggable nature of a LoginModule prevents the development of brittle application code.

The Authenticate.authenticate method makes a connection to the WebLogic server specified in the URL and passes the Subject and the Environment, which holds the credentials, to the server to be authenticated by the Authentication provider configured for the WebLogic Security Realm of the server. The Authentication provider configured in the server's security realm also implements a LoginModule. The implementation of the Authentication provider can perform the authentication using any number of technologies, such as LDAP or a relational database. If the authentication succeeds, the Authentication provider adds the authenticated Principal(s) to the Subject. Before it adds the Principal to the Subject it makes a request to the Principal Validator to digitally sign the Principal. This prevents potentially malicious clients from tampering with the Principal embedded in the returned Subject in an attempt to circumvent authorization checks when the Subject is returned via a Security.runAs call. The Principal Validator is consulted during authorization to ensure that the returned Subject is the same one that was digitally signed during authentication.

Browser-Based Authentication

Many, if not most, WebLogic-based applications are accessed through browser-based clients. These applications are generally composed of servlets, JSPs, and EJBs. Authentication to these applications is specified as BASIC or FORM in the <login-config> element of the web.xml deploy-

ment descriptor of the Web application that will present the initial access page of the application. For example:

```
<login-config>
  <auth-method>BASIC</auth-method>
</login-config>
```

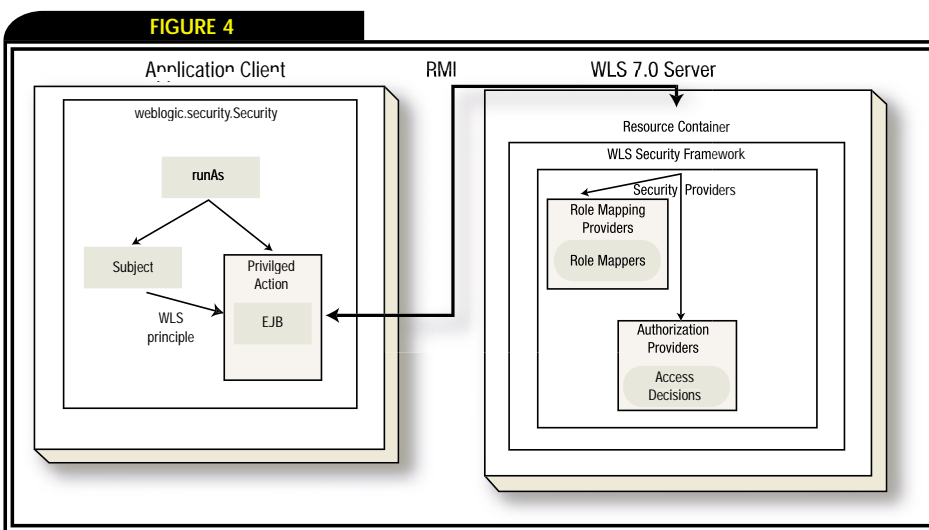
When one of these methods is used to authenticate a Web client, the Web container calls to the WebLogic Security Framework, on behalf of the client, to access the Authentication provider. This mechanism results in the creation of a JAAS Subject, which contains an authenticated Principal, that is stored within an internal session object. Subsequent requests from the client are authorized by locating the Subject in the internal session using the session ID passed in a cookie in the HttpRequest. The point here is that resource containers on WebLogic, such as the Web container, use JAAS-based authentication as well by calling the Authentication provider on behalf of the Web client to get a Subject populated with an authenticated Principal. The LoginModule used by the Authentication provider is not configured in a JAAS configuration file as it is in the Java client example. The Authentication provider is configured by adding it to the active Security Realm through the administration console. The default Authentication provider and LoginModule configured for WLS 7.0, out-of-the-box, use an embedded LDAP server. The server-side LoginModule that authenticates the Web-based client is the same LoginModule that is used to authenticate a Java client application when the client-side UsernamePasswordLoginModule calls the Authenticate.authenticate method passing the Subject and Environment objects. The Subject is not passed back to the Web client, but rather is kept by the Web container in an internal session object and is later referenced by the session ID.

JAAS-Based Authorization

While authorization in WLS does not use the JAAS Subject.doAs method, it is JAAS Subject based. Applications that require access to protected WebLogic resources request access to them through the weblogic.security.Security.runAs method. A Subject and PrivilegedAction are passed to this WLS Security Framework method to perform a task involving a WebLogic resource.

WebLogic Resources

Unlike Java system resources that are protected through security policies found



WLS 7.0 Authorization

Sitraka (now part of Quest Software)
www.sitraka.com/jclass/wldj

in the Java security policy file, WebLogic resources are protected by WebLogic security policies that are specified by the association of a WebLogic role with a WebLogic resource. A WebLogic resource is defined as a structured object that represents a server-side entity that can be protected from unauthorized access. Examples of WebLogic resources are EJB methods, servlets, and JMS Destinations. See the WebLogic 7.0 documentation at <http://edocs.bea.com/wls/docs70/dvspisec/atz.html#1134702> for examples of the types of WebLogic resources that can be protected using WebLogic roles.

WebLogic Roles

WebLogic roles, found in WLS 7.0, are used to replace ACL-based authorization found in previous releases of WebLogic. A role is defined by the WLS documentation as an abstract, logical collection of users similar to a group. Roles are different from groups because they are dynamically updated based on username, group membership, and time of day. Roles are used with resources to create WebLogic security policies. JAAS-based security policies are defined in the Java security policy file by granting Permissions to codebases, signers, and Principals. Associating a WebLogic role with a WebLogic resource creates a WebLogic security policy. WebLogic does not consult a Java policy file to apply security policies. A user who is in the role defined by a security policy for a resource at the time an access decision for the resource is made is allowed to access the resource.

WebLogic roles can be global, which associates them with all WebLogic resources, or scoped, which associates them with a specific WebLogic resource.

Global roles are declared using the administrative console. Security policies are created dynamically, using scoped roles, for Web applications and EJBs through the use of deployment descriptors. These roles are declared and associated with the resource in the web.xml file for Web application components and in the ejb-jar.xml file for EJBs using the <security-role> element. The declared roles are granted to Principals through the vendor-specific deployment descriptors, weblogic.xml, and weblogic-ejb-jar.xml respectively, using the <security-role-assignment> element. Dynamic roles may be configured in the console too, but the deployment descriptors of the components are not updated with the console modifications. The Role Mapper does the work of associating the roles granted to the principals at deployment time.

WebLogic Authorization

Authorization of the Java client application is initiated when the application attempts to execute a PrivilegedAction through the Security.runAs method (see Figure 4). The application passes the authenticated Subject and the PrivilegedAction to the WebLogic security framework when it calls this method. In the WLS example, an EJB method is invoked from a PrivilegedAction, SampleAction.java, using the authenticated Subject as the identity of the caller. The EJB container on the server receives the request from the client-side EJB stub in the context of the PrivilegedAction, which propagates the authenticated Subject to the EJB container with the method invocation. The EJB container calls to the Security Framework on the server to determine if the Subject is allowed to access the EJB method. This

activates the Authorization provider and the Role Mapper to make a decision regarding the Subject's access to the EJB method being called. The Role Mapper determines what Roles are held by the Principals in the Subject. The Authorization provider determines if the Roles held by the Principals allow access to the method. If access is allowed, the container proceeds with the method invocation. It should be noted that before the authorization process begins, the Principal Validator validates that the Principals in the passed Subject have not been tampered with since authentication.

Browser-based clients are authorized by the same mechanism on the server. The difference is that the Subject is kept on the server by the Web container rather than being passed from the client application. The browser sends the session ID in a cookie and the Web container locates the subject in an internal session before calling to the security framework to perform authorization.

Conclusion

The JAAS mechanisms of authentication and authorization, defined by the JAAS specification and implemented in JDK 1.4, use the Java SecurityManager, the AccessController, LoginModules, and Subjects to perform authentication and authorization checks. These mechanisms, used to protect system resources and properties, are configured using the Java security and policy files. WebLogic uses JAAS LoginModules and Subjects to perform authentication and authorization; however, the Configuration is not specified with, or retrieved from, the Java security and policy files.

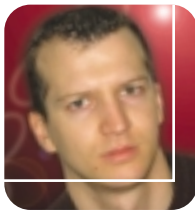
Security providers that are configured through the WebLogic console implement the roles of the Java SecurityManager and AccessController in a WebLogic server. These security providers control access to WebLogic resources using security policies that are defined by the association WebLogic roles and WebLogic resources. Roles may be configured through the console or through deployment descriptors. A WebLogic authentication provider uses LoginModules and Subjects to establish the identity of a user. An authenticated Subject is used by a WebLogic Authorization provider to grant or deny access to WebLogic resources based on the security policies defined for those resources and the roles held by the Principals in the authenticated Subject. 🍌

“Security providers that are configured through the WebLogic console implement the roles of the Java SecurityManager and AccessController”

Neon Systems

www.neonsys.com/BEApaper

WRITING A CUSTOM JAAS LOGINMODULE TO SECURE DATABASE AUTHEN



BY TIM PIJPOPS

AUTHOR BIO...

Tim Pijpops is a software architect at Egova-Cronos, a Belgian company that focuses on e-government solutions with J2EE technology. Besides Java and J2EE, his professional interests include software development methodologies, patterns, and quality assurance. Tim holds a master's degree in computer science from the University of Leuven (KUL).

CONTACT...

Tim.Pijpops@egova.com

With the arrival of BEA WebLogic Server 7.0, a new security architecture with improved and expanded security features was introduced.

The primary goal of that new architecture is to let you plug in security providers from different security vendors and to let them cooperate seamlessly with your J2EE applications

A security provider is a module that handles some specific security tasks, like authentication and authorization. However, instead of buying a security provider you can develop your own. When there are no third-party providers that meet your security needs, this can be your only effective solution. I ran into this situation when I needed to authenticate users based on information in a relational database. Since there was no standard WebLogic provider that did the job, and no suitable third-party alternative, I decided to develop my own database authentication-provider. In fact, it wasn't that difficult, as I will show you in this article.

Authentication in WebLogic

Authentication is probably the most important part of security. It is the process that proves the identity of a user based on some credentials, in most cases a password. The default WebLogic 7 authentication provider uses an embedded LDAP server to store users and groups as a replacement of the well-known file realm in WebLogic 6. In addition, WebLogic 7 provides a set of LDAP

authentication providers for accessing external LDAP stores like OpenLDAP or iPlanet. And what happened with the RDBMS security realm that was present in WebLogic 6? Well, it is still possible to use the RDBMS security realm, but you will have to boot WebLogic Server in so called "compatibility mode," meaning that the server provides backward compatibility with the WebLogic 6.x security realms by using a special kind of authentication provider: a realm adapter authentication provider. However, the use of these realm adapter providers is deprecated; hence, to make your projects future-proof it's better to use an authentication provider that fits into WebLogic's new security architecture.

Secure Passwords

Prior to guiding you through the different steps in developing your own authentication provider, it's important to discuss how you can use a relational database as an authentication information store with the help of encryption. After all, the most important part is the storage of passwords. Storing these passwords in plain text in the database isn't the right way to go. It's better to obscure the passwords with a messagedigest algorithm. A messagedigest is a secure, one-way hash function: a function that calculates a fixed-length hash value of an arbitrary-sized original value and can be computed only one way. Once the algorithm has been applied to the password, it's impossible to go backwards and find the original password – viewing the messagedigest alone will not reveal the

WHEN THE THIRD-PARTY PROVIDERS



SUPPORT TICATION

contents of the original message and changing even a single bit on the message will result in a totally different output value. The only way is to pass in random input until you get the original password – a brute-force attack. Your authentication provider doesn't need to know the original passwords at all – it only needs to apply the hash function on any password given by a user and compare it with the stored hashed password of that user. If both are the same, the user supplied the correct password and can be authenticated.

Putting the MessageDigest to Work

In Java there's an easy way to calculate message digests by using the `java.security.MessageDigest` class. The following code snippet shows you how to apply an MD5 message digest algorithm (a 128-bit digest) on a password string:

```
MessageDigest md =
    MessageDigest.getInstance("MD5");
md.update(originalPwd.getBytes());
byte[] digestedPwdBytes = md.digest();
```

Message digests are also used to create a checksum, a unique ID for a piece of text (also called a digital fingerprint). This is what happens if you sign a JAR file: a checksum is calculated from the contents of the JAR file, encrypted, and stored in the `manifest.mf` file

in base64 encrypted format. Base64 is a way to encode arbitrary binary data so that the result contains only printable characters (note that base64-encoded data takes one-third more space than the data before conversion). Since the message digest algorithm outputs its response as a byte array, we can use base64 encoding to transform the hash bytes into a string so that we can store it in a varchar field in the database. There are many base64 encoders available, but the easiest way is to use the one that's included in the `weblogic.jar` library: `weblogic.apache.xerces.utils.Base64`. The use of this class is trivial, as shown in the next code sample:

```
String digestedPwdString =
    new String(Base64.encode(digestedPwdBytes));
```

Security Provider Architecture

Now let's go back to the Authentication provider itself. As I mentioned before, an Authentication provider is one kind of security provider in WebLogic 7.0. To understand the structure of an Authentication provider, it's important to discuss the general architecture of all security providers. Each provider is a module that can be plugged into a security realm. It is developed by implementing the appropriate security service provider interfaces (SSPIs) from the `weblogic.security.spi` package and by creating an MDF file to generate an MBean type used to configure and manage the provider (see Figure 1).

MBeans, or Managed Beans, are a construct of JMX, the Java Management eXtensions package. JMX provides support for application and network management in Java. An MBean is a specific type of JavaBean that can be managed in a JMX-compliant application. In WebLogic, MBean types are created by the WebLogic MBeanMaker utility on the basis of an MBean Definition File (MDF), an XML file that describes the MBean type. This XML file consists of properties used to configure the security provider, e.g., name, description, and implementation classes. These properties can be exposed in the server's administration console, where you can view and edit them.

JAAS for Authentication

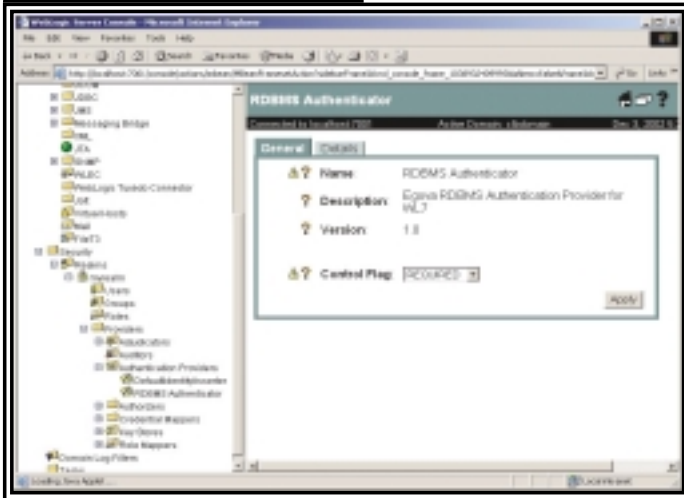
In developing my own authentication provider, the sample security providers on BEA's dev2dev site turned out to be very useful. I used them as a starting point to work out my database authentication

provider and I recommend that you take a close look at them if you are planning to do some security provider development yourself. If you look at the sample authentication provider, you will see the use of the Java Authentication and Authorization Service (JAAS) in the provider's implementation classes. WebLogic uses JAAS internally for authentication as well as for remote fat-client authentication. The authentication provider authenticates users in a JAAS login module (`javax.security.auth.spi.LoginModule`). Each authentication provider needs to have exactly one `LoginModule`, but you can configure multiple Authentication providers in one security realm. This way you have multiple `LoginModules` that can perform different kinds of authentication for a user. The authentication mechanism itself is based on users and groups: a user represents a person, a group represents a category of persons. After authentication, WebLogic assigns a principal to both users and groups. The principals are stored in a JAAS subject during the `commit()` method of the login module. Any principal that is going to represent a WebLogic Server user or group needs to implement the `WLSUser` or `WLSGroup` interface. The sample Authentication provider on dev2dev uses the `weblogic.security.principal.WLSUserImpl` to represent the user into a subject, and the `weblogic.security.principal.WLSGroupImpl` to represent the groups that contain the user. That's sufficient for the database provider too, so you can reuse the same mechanism.

Closer Look at the LoginModule

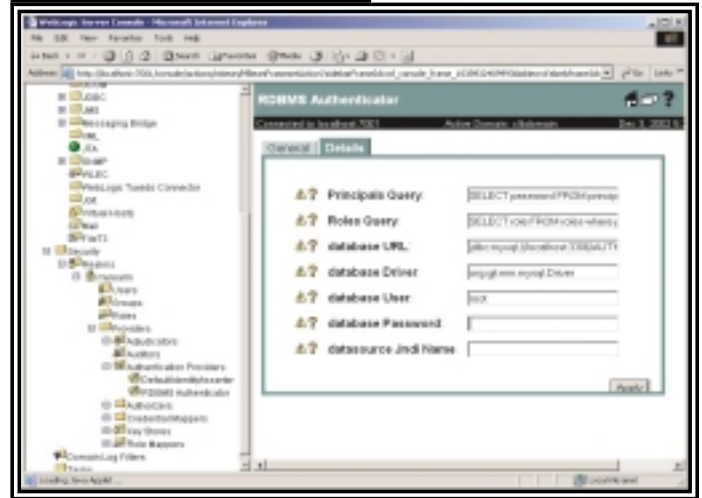
The only things you need to change in the sample login module are the details in checking the correctness of username and password in the `login()` method. Obviously the security framework calls this method when a user has to be authenticated. Authentication has to be proven here by checking the user's password. The sample Authentication provider delegates to the `SampleAuthenticatorDatabase` class to retrieve user and group information from a persistent store – in this case, a plain text file. You have to do two things. The first one is changing the password check in the `login()` method. Remember that you should store a hash value of the password, so instead of checking if a given password matches the password in the database, you should apply the message digest algorithm on the given password first, and then com-

FIGURE 1



Installation of the RDBMS Authenticator

FIGURE 2



Configuring the authenticator's properties

pare it with the stored hash value. Second, you have to adapt the `SampleAuthenticatorDatabase` (or better, use your own class like `RDBMSAuthenticatorDatabase`) to retrieve user and group info from a relational database instead of a text file. If you don't need hierarchical roles, it's sufficient to create two simple tables in the database:

```
CREATE TABLE Principals
(principalId varchar(50), password var
char(25));
CREATE TABLE Roles
(principalId varchar(50), role varchar(50));
```

Next you need to add some simple database access code in the authenticator's database class. However, there's a snake in the grass here: since WebLogic uses the Authentication provider to also authenticate the user that boots the server, you can't use a `javax.sql.DataSource` as the factory for database connection objects because the naming service isn't started yet. You have to use a `DriverManager` to obtain a connection to the database. Once the server is started, it's no problem to use a `DataSource`, and you should probably do it at that time to benefit from connection pooling.

Implement the SSPI

The last Java class you have to write is the implementation of the `weblogic.security.spi.AuthenticationProvider` SSPI. This class represents the runtime implementation of the authentication provider. Here you need to specify the authenticator's database and `LoginModule` implementation classes. For the `SampleAuthenticationProviderImpl`, these are `SampleLoginModule` and `SampleAuthenti-`

`atorDatabase`. Obviously you need to change these into your own classnames. After successful authentication, a principal validation provider is needed to sign the principals and ensure their authenticity between programmatic server invocations. The `SampleAuthenticationProviderImpl` reuses the standard WebLogic principal validator for this purpose, and that's sufficient for the database authentication provider too.

Configuration and Deployment

That's all for the Java classes. The last thing you need is the MDF file. Again, you can easily adapt the `SampleAuthenticator.xml` file and supply some additional properties as `<MBeanAttribute>` tags, e.g., database URL, drivename, username, password, query to retrieve principal info, query to retrieve role info. The MBean that is generated from this MDF has to be supplied as an argument in the constructor of the authenticator's database class, so you can use the properties of the MBean to configure the class. The next step is to generate the MBean using WebLogic's `MBeanMaker`. The sample security provider's archive contains example ANT scripts showing you how to do this with the `weblogic.management.commo.WebLogicMBeanMaker` class. The same class is used to build the authentication provider's MJF file (MBean Jar File). Finally, you need to copy that Java archive into the `/lib/mbeantypes` directory relative to the home dir of your WebLogic Server installation. After a reboot of the server, you can install the authentication provider by navigating in the admin console to `Security - Realms - <yourRealm> - Providers - Authentication Providers`. The name of the newly created provider should

be visible now, e.g. "configure new RDBMS Authenticator" (see Figure 2). Set the right properties and reboot the server. This will activate the authentication provider. Remember that WebLogic uses it immediately upon booting up the server, so make sure there's a user in the Administrator's role present in your database or else you won't be able to start the server.

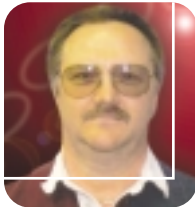
Conclusion

Although primarily intended as a framework to plug in third-party security providers, WebLogic's new security architecture can be used to develop your own custom security provider too. It's not that hard to create, deploy, and configure your own authentication provider, as I've demonstrated in this article. Basic knowledge of JAAS and JMX, some good samples, and a bit of encryption magic is all you need. The last thing is perhaps the most important, because even the best authentication provider turns out to be worthless when combined with an insecure authentication store.

Resources

- *BEA WebLogic Server 7.0 Security Web site:* <http://edocs.bea.com/wls/docs70/security.html>
- *BEA dev2dev code direct: WebLogic 7.0.1 Sample Security Providers download:* http://dev2dev.bea.com/code/code_direct.jsp
- *Java Management Extensions home page:* <http://java.sun.com/products/JavaManagement>
- *Java Authentication and Authorization Service home page:* <http://java.sun.com/products/jaas>

BEA eWorld
www.bea-eworld.com



BY PAUL PATRICK
& VADIM ROSENBERG

AUTHOR BIO...

Paul Patrick is the chief security architect for BEA. He was the architect of BEA's (and earlier Digital Equipment Corporation's) Object Broker CORBA ORB, coarchitect of WebLogic Enterprise (Tuxedo), and has been the security architect for WebLogic Server since version 6.0.

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

CONTACT...

paul.patrick@bea.com
vadimr@bea.com

This article appears courtesy of
BEA dev2dev Online.



WEBLOGIC SECURITY FRAMEWORK

WORKING WITH YOUR SECURITY ECOSYSTEM

WebLogic Server 7.0 offers a new, integrated approach to solving the overall security problem for enterprise applications. With this framework, application security becomes a function of the application infrastructure and is separate from the application itself. Any application deployed on WebLogic Server (WLS) can be secured either through the security features included with the server out of the box, by extending the open Security Service Provider Interface to a custom security solution, or by plugging in other specialized security solutions from major security vendors that the customer's enterprise standardizes on.

This article defines the major requirements for an integrated application security solution, and explains how WebLogic Server 7.0 Security Framework delivers them to your application.

Requirements

The goals of application security are simple: (1) enforce business policies concerning which people should have access to which resources, and (2) don't let attackers access any information. Goal (1) causes a problem because it seems acceptable to enforce business policies in

business logic. This belief is misplaced because it's much harder to change policies when enforcement occurs in business logic. Consider the analogy to a secure physical filing system. You don't take a document and rewrite it when a security policy changes. You put it in a different filing cabinet. Different filing cabinets have different keys and a security officer controls their distribution. Similarly, application developers should not have to change business logic when security policy changes. A security administrator should simply alter the protection given to affected components.

Moreover, mixing security code with business logic compromises both goals if developers make mistakes. When the security code in a component has a defect, people may accidentally access information they shouldn't and attackers may exploit the defect to gain unauthorized access. Of course, mistakes are unavoidable. That's why we test software. But it's a lot harder to test the security of every application component individually than a security system as a whole. The difference is somewhat analogous to reading every document in our hypothetical filing system for its fidelity to security policies rather than simply testing the integrity of the locked filing cabinets. However, we shouldn't blame application developers for

mixing security code and business logic. We should blame middleware security models. Most of them simply do not support the types of policies many enterprises have, such as only an account holder can access his account. Unless these security models begin supporting a much more dynamic type of security, developers really have no choice.

Middleware security models also fail enterprises in goal (2). Keeping attackers out requires a united front from all the elements in a distributed system. Cooperation is the key to this united front. Middleware sits between front-end processors and back-end databases. The middleware security system must be prepared to accept as much information as it can from the front-end processors about the security context of their requests and must be prepared to offer as much information as it can to back-end databases about the context of its requests. Moreover, it must be prepared to cooperate with special security services that work to coordinate the efforts of all these tiers. Middleware security models offer little, if anything, to support such cooperation. This failing affects many aspects of application security.

Authentication

Authentication is the first line of defense. Knowing the identity of requesters enables the application layer to decide whether to grant their requests and poses a barrier to attackers. All authentication schemes work in fundamentally the same way. They offer a credential to establish identity and provide a means to verify that credential. However, there is a wide variation in the form of credentials and verification mechanisms. Each enterprise's choices of authentication schemes depend on a number of factors, including the sensitivity of protected resources, expected modes of attack, and solution life cycle cost. In most cases, enterprises already have one or more authentication schemes in place, so middleware must work with them by accepting their credentials and engaging their verification mechanisms. Without this cooperation, the enterprise must use a lowest common denominator scheme like passwords, potentially limiting the use of such middleware to low-value applications.

The problem of Web single sign-on (SSO) is even more difficult. The motivation for SSO stems from the distributed nature of Web applications. From the user perspective, a single application may actually encompass different software components

running on different servers and operated by different organizations. Users don't want to resubmit credentials every time they click a link that happens to take them to a page running in a different location. Their experiences should be seamless. The previous problem of working with existing authentication schemes requires only understanding credential formats and integrating with verification mechanisms. However, with Web SSO users don't even want to provide credentials in many circumstances. Establishing a user's identity without seeing his credentials requires sophisticated behind-the-scenes communication between the two servers involved in handing off a user session. There are a number of proprietary solutions and some emerging standards for this communication, but it is likely that a given application may have to support multiple approaches for the foreseeable future, so an open model is necessary.

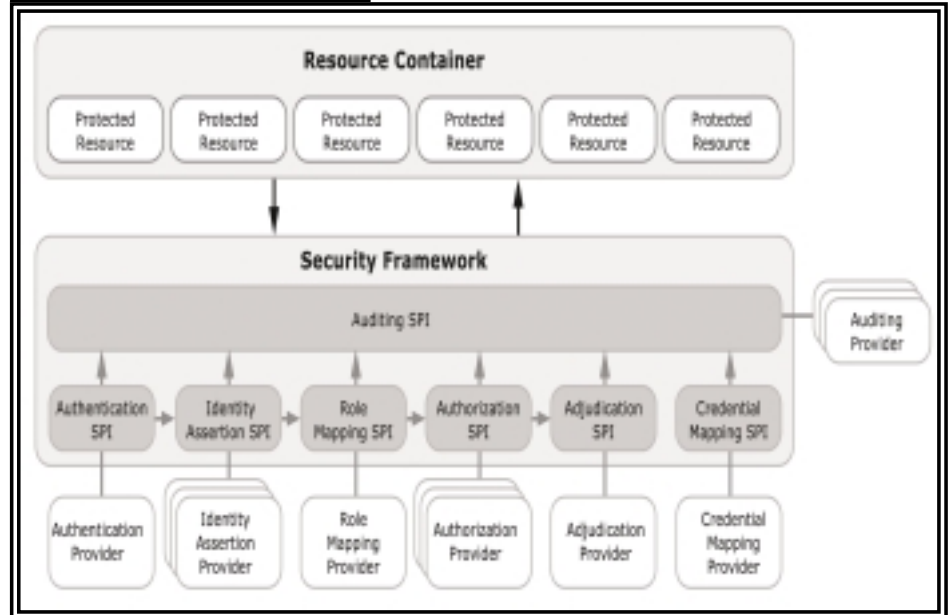
Working with other Web application components involves cooperation on the front end, but middleware infrastructure must also cooperate on the back end. Databases have been around a long time and enterprises take database security very carefully. They really don't trust the front end and middleware layers. If an attacker were to compromise either one of these layers, he could potentially issue a sequence of database requests that would return a large fraction of all the data it maintains. Also, if the front-end or middleware components have defects, they could

unintentionally request data for the wrong user, resulting in an embarrassing disclosure of private information. Therefore, many enterprises want to bind each database request to a particular end user, including the appropriate credentials that establish the user's identity. Applications must be prepared to propagate this information.

Authorization

Once an application has established the requester's identity, it must decide whether the set of existing security policies allows it to grant the request. Typically, middleware infrastructure such as J2EE uses a static, role-based system. During user provisioning, security administrators explicitly assign roles to users and then update these assignments as conditions require. During component deployment, security administrators indicate the roles allowed to access the component. At runtime, if a request comes from a user with the necessary roles the application grants the request. This static approach ignores the dynamic nature of many business policies. Consider the policies governing bank accounts, expense reports, and bank tellers. For bank accounts, customers should only be able to access their own accounts. For expense reports, a manager can provide an approval only up to a set amount and never for his own expenses. For bank tellers, they only fulfill the teller role when they're on duty. In even more sophisticated policies, authorization depends on the combination

FIGURE 1



Security Framework Architecture

of roles assigned to a user, as well as the content of the request. Middleware infrastructure must explicitly support these dynamic policies or at least provide enough context to specialized security services that do.

The need for dynamic authorization raises the issue of administration. We definitely don't want to force security administrators to become experts in programming languages like Java. Certainly there will be unusual situations that require some custom programming, but routinely updating the dollar threshold for expense report authorization shouldn't require it. At a more mundane level, we don't want them to dig through XML-formatted deployment descriptors and then redeploy components to update role assignments. Security administrators need a well-designed graphical user interface that lets them perform all of their routine tasks and most of their nonroutine ones at runtime. Managing user lists and their assigned roles, changing the level of protection for components, and configuring dynamic constraints should all require just a few moments.

A more complicated headache for security administrators comes in migrating from one authorization service to another. Due to the complexity of authorization decisions, many enterprises rely on specialized services and all applications delegate such decisions to them. When it comes time to perform a major version upgrade or switch to a different service, administrators face a quandary. When do they switch over to the new provider? The concern lies with defects or configuration problems in the new service. They don't want to switch over only to experience a massive case of improper

authorizations or mistaken rejections. What they'd really like is to use both systems simultaneously and note when the old and the new service differ in their decisions, but this approach requires an even greater ability for the middleware infrastructure to cooperate with the rest of the security ecology.

Auditing

If an application could simultaneously use two different authorization services, a difference of opinion would be a noteworthy event and administrators would want to know about it. Unfortunately, most middleware infrastructure neglects this type of security auditing. Proper auditing is not simply a matter of writing information to disk somewhere. To support their duties to *verify*, *detect*, and *investigate*, administrators need records of all security events in a single location, active notification of certain especially important events, and the ability to quickly search the records.

Security administrators are responsible for ensuring the enforcement of the enterprise policies regarding information access. Obviously, they must first specify these policies, hopefully using a productive interface as described above. Then they must verify the actual enforcement of these policies by periodically inspecting the audit trail. Government regulations or commercial contracts may require such audits. Administrators sample a representative set of transactions and track their paths through various application components to ensure the correct enforcement of security policies at each step. They need a consolidated audit trail or they'll have to spend a significant effort on manually assembling

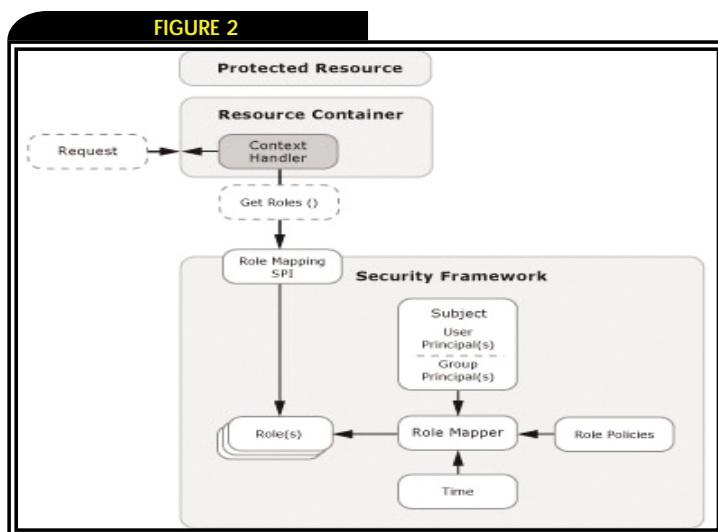
logs from different locations. They need detailed records or they won't be able to determine full compliance.

Responding to potential breaches is the other primary responsibility of security administrators. Responses involve two steps, detection and investigation. First, they need the ability to specify conditions under which the security system will actively notify them. These conditions could involve transaction values, such as transfers over a million dollars, or a pattern of events, such as a spike in the number of clients connecting using weak encryption when accessing sensitive functions. Once they receive notification, administrators must be able to quickly search the logs to determine if there has been an actual breach and the extent of any damage. These searches may involve complex criteria and must execute against the live audit trail so they can track a particular attack as it unfolds. These requirements make the auditing subsystem a substantial piece of software in its own right that middleware providers must devote a substantial effort to perfecting.

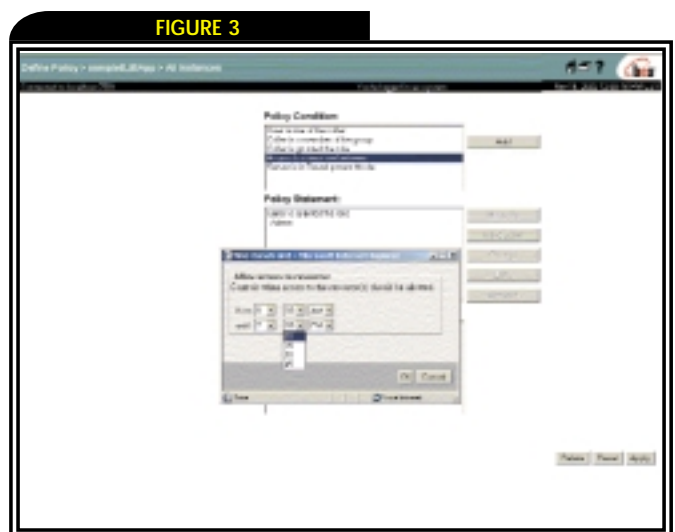
As we've seen, there are many challenges in application security. But like the essence of application security itself, the solution is also rather simple.

1. A clean, elegant abstraction between security policy and business logic
2. A simple, declarative interface for managing security policies in real time
3. An open, flexible architecture for integrating with security services

These practices avoid the problem of mixing security and business logic, streamline security administration, and enable



Dynamic role mapping process



Administering role mapping

FileNet
www.filenet.com

cooperation with the rest of the security ecology. The WebLogic Security Framework delivers these critical capabilities.

Security Framework Architecture Overview

The goal of the WebLogic Security Framework is to deliver an approach to application security that is comprehensive, flexible, and open. Unlike the security realms available in earlier versions of WLS, the new framework applies to all J2EE objects, including JSPs, servlets, EJBs, JCA Adapters, JDBC connection pools, and JMS destinations. It complies with all of the J2EE 1.3 security requirements, such as JAAS for objects related to authentication and authorization, JSSE for communication using SSL and TLS, and the SecurityManager class for code-level security.

The heart of the architecture is the separation of security and business logic. Business logic executes in an appropriate container, be it a JSP, servlet, or EJB. When the container receives a request for an object it contains, it delegates the complete request and its entire context to the Security Framework. The framework returns a yes or no decision on whether to grant the request. This approach takes business logic out of the security equation by providing the same information to the security system that is available to the target object. They each use this information to fulfill their dedicated responsibility: the framework enforces security policy and the object executes business logic.

When the Security Framework receives a delegated request, it manages security processing (see Figure 1). This processing is very flexible, with fine-grained steps not found in many systems, such as dynamic role mapping, dynamic authorization, and adjudication of multiple authorizers. At each step, it delegates processing to an included third-party or custom provider through the corresponding service provider interface (SPI). This architecture enables WLS to route the information necessary to each service provider so that applications can take full advantage of specialized security services.

Service Provider Integration

The Security Framework per se manages security processing. Each step requires execution by a service provider. WLS 7.0 includes providers for every step, but they

simply use the framework SPIs. Any other provider has access to the same facilities. These SPIs include:

- **Authentication:** Handles the direct verification of requester credentials. The included provider supports username/password and certificate authentication via HTTPS.
- **Identity Assertion:** Handles requests where an external system vouches for the requester. The included provider supports X.509 certificates and CORBA IIOP CSiv2 tokens. Because the Security Framework can dispatch requests to different providers based on the type of assertion, you can support a new external system by simply adding a provider for that system type.
- **Role Mapping:** Handles the assignment of roles to a user for a given request. The included provider supports dynamic assignment based on username, group, and time.
- **Authorization:** Handles the decision to grant or deny access to a resource. In the future, WLS will support many dynamic features, such as the evaluation of request parameter values. The Security Framework supports simultaneous use of multiple authorizers coordinated by an adjudicator.
- **Adjudication:** Handles conflicts when using multiple authorization providers. When all the authentication providers return their decisions, the included provider determines whether to grant the original request based on either the rule “all must grant” or the rule “none can deny.”
- **Credential Mapping:** Handles the mapping of application principals to backend system credentials. As shown in Figure 1, it's not part of the process leading to an access decision because it's invoked when an object makes a request rather than when an object receives a request. The included provider supports username/password credentials and is used internally for J2EE calls and Web SSO.
- **Auditing:** Handles the logging of security operations. As shown in Figure 1, it is slightly different from the other SPIs because it is invoked whenever a provider of any kind executes a function. The included provider supports reporting based on thresholds and writes all reported events to a log file. The Security Framework supports simultaneous use of multiple auditors, making it easy to integrate with external logging systems.

These clean SPIs make it possible to plug and unplug different providers as the security ecology evolves, benefiting everyone involved. BEA can individually upgrade the providers included with WLS. Specialist security vendors can easily make their services available to J2EE applications by coding their products to the appropriate SPIs and many have already done so. Moreover, enterprises can quickly implement customized security processing where necessary. Instead of adapting your security posture to suit the middleware, the middleware adapts its security processing to you.

From an administrator's perspective, selecting from available providers is simply a matter of pointing and clicking. Using the WLS console, you expand the Realms node and then expand the Providers node. For a given provider, you select one of the available provider instances and configure its properties.

Backwards Compatibility

As described above, the WebLogic Security Framework revolutionizes application layer security. However, you may have invested a substantial effort in configuring the security realms used in WLW 6.x. You might not want to upgrade your security model immediately, so the framework offers a realm adapter for backwards compatibility. This adapter is the complete security subsystem from WLS 6.x and the framework treats it just as any other service provider that implements the authentication and authorization SPIs. At server startup, the adapter extracts access control definitions from the deployment descriptor just as before. At runtime, it accepts authentication and authorization requests delegated from the framework through the corresponding SPI. From your perspective, WLS 7.0 security behaves just like 6.x security. From the server's perspective, the realm adapter is fully integrated into the 7.0 Security Framework. Once you decide to upgrade, you can easily import the security information from 6.x definitions. You can even perform simultaneous authorization with the realm adapter and the Security Framework's native provider to verify proper behavior of the upgrade.

In some cases, you may be using the 6.x realm's integration with the distributed user management systems in Unix or Windows NT. In these cases, you might want to continue using this integration for authentication but want the benefits of the dynamic authorization from the Security Framework.



Component Source

www.componentsource.com/bea

Therefore, it has an option to use the 6.x realm adapter only as an authentication provider. It's interesting to note how the flexibility of the framework's SPI architecture cleanly addresses what might otherwise be a very tricky backwards compatibility issue.

Security Integration Scenarios

The Security Framework offers a lot of flexibility. Let's look at a few specific examples of this. In some cases, the solution may not be completely finished, but the planned design demonstrates the superiority of an open framework approach.

Perimeter Authentication

In many cases, a party other than WLS's own authenticator vouches for the identity of a requester. It may be the SSL layer of WLS. It may be a Kerberos system. It may be an intermediary Web service. In these cases, the third party provides a token that the application can verify. As long as it trusts the third party, it can accept a verified token as if it were the original user credential.

The Security Framework employs a straightforward mechanism for working with such systems. All a third party has to do is put its token in an HTTP header. The Security Framework examines the token and dispatches an appropriate service provider based on the token type. If an X.509 certificate from mutual SSL authentication comes in, the framework dispatches

a provider that can verify the certificate chain to a root certificate authority and perhaps check the current validity of the certificate using the Online Certificate Status Protocol. If a Kerberos ticket or WS-Security token comes in, the appropriate provider decodes the token and performs the necessary verification.

Once this verification is done, the provider maps the identity in the credential to a local user. The framework calls back to the JAAS with this local user, which then populates the Principal object as specified in J2EE 1.3. This approach is fully compliant with the appropriate standards yet still offers flexibility. A third-party provider or enterprise development team can integrate any authentication technology with WLS as long as they can populate an HTTP header. Integrating WLS applications with Web SSO solutions is easy because most of them, including SAML, already use cookies or HTTP headers.

Role Associations

Most application security models employ the concept of roles, which provide a layer of indirection between users and resources that increases the ease of administration. Roles are like Groups, but more dynamic. Typically, a security administrator assigns a user to a group upon provisioning and then changes this assignment only when the user's job responsibilities change. Roles change more often, perhaps even from request to request based on specific condi-

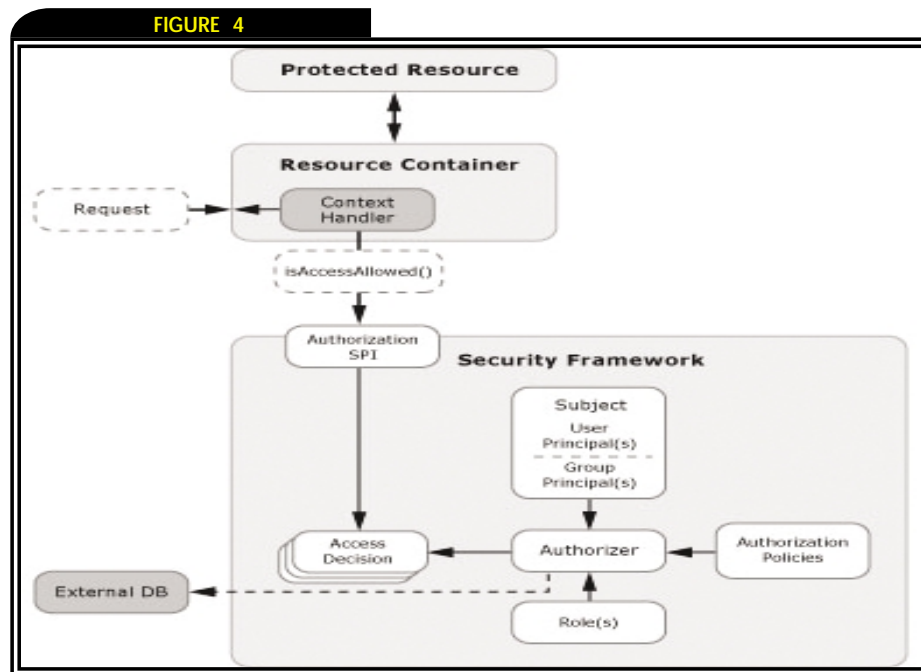
tions. The Security Framework supports both Groups and Roles.

Administrators can set up roles so that they embody a logical identity such as Teller or name a set of logical permissions such as Deposit, Withdraw, and Transfer. It's really a matter of design. The first approach is more focused on the logical role of the user and the second approach is more focused on the logical role of the resource. The Security Framework distinguishes between globally scoped roles, which apply to all resources in an installation, and resource-scoped roles, which apply only to specific resources. Globally scoped roles are intended primarily for managing different levels of administrative privileges. Administrators will configure and manage resource-scoped roles.

The Security Framework enables service providers to dynamically assign roles based on context (see Figure 2). The included provider can take into account username, group, and time. For example, suppose a bank had an AccountManagement EJB restricted to users with a Teller role. A user would fulfill the Teller role if he or she were a member of the DayTeller group and the local time was between 8 a.m. and 6 p.m. Administrators could also use this time feature to set up role assignments that automatically expire, which might be especially useful for very sensitive information such as human resources data. Figure 3 shows how easy it is to set up this dynamic assignment. The role-mapping SPI actually supports the use of additional information such as the parameters of the method call. Therefore, custom providers could offer even more flexible role mapping. Consider the case of a CFO and expense accounts. A custom provider could grant the CFO an Approver role unless the Employee parameter of the request were the CFO himself. That way, he couldn't approve his own expenses.

Credential Mapping

As discussed above, enterprises often want to tie each request of a back-end database, packaged application, or legacy system to the ultimate user. Therefore, when a J2EE object accesses a back-end system on behalf of a user, it has to supply the appropriate credentials to the system. The basic problem is mapping a J2EE Principal to back-end system credentials. The included service provider solves this problem for the most common case of username/password credentials. Each WLS instance has an embedded LDAP directory



Parametric authorization process

Altaworks
www.altaworks.com



ATTN: Developers

STEP UP
to the mike
and be...

Go to
<http://developer.sys-con.com>

HEARD!

**Calling Sleek
Geeks Everywhere!**

Make sure you have your
finger on the pulse of
i-Technology...bookmark
<http://developer.sys-con.com>
today.

i-Technology

News

i-Technology

Views

i-Technology

Comment

i-Technology

Debate



© COPYRIGHT 2002, SYS-CON MEDIA
WWW.SYS-CON.COM

**SYS-CON
MEDIA**

in which it can store the encrypted user-name/password pair for every valid combination of Principal and back-end system.

Parametric Authorization

One of the classic application security problems is making an authorization decision based on the content of the request or the target object. Approval thresholds are a common case where you want to evaluate the value of these parameters. First, you would create a set of roles such as Manager, SeniorManager, and Director. Then, you would create a set of policies that authorizes approval requests for each role based on the value of the Amount parameter, such as \$5,000 for a Manager, \$10,000 for a Senior-Manager, and \$20,000 for a Director. Most middleware does not currently address this issue. A future version of the included authorization service provider will allow such decisions based on the content of the request (see Figure 4). In fact, the authorization SPI already supports the use of method call parameters in access decisions, so you could build a custom provider with these capabilities today.

Authorization based on the content of the target is a little trickier. For example, you may want to inspect an Account object to get the value of AccountHolder before you decide to authorize a withdrawal. Unfortunately, in the general case, this type of visibility can break encapsulation and present a security vulnerability. If the security system can access any data in the system, it presents a tempting target for compromise. It will soon be possible to write custom code to perform this type of operation in select cases. A future version of the SPI will enable you to access context besides the method call parameters, such as the EJB primary key. You could then create a very specialized provider that examined the primary key of the Account object targeted by a Withdrawal request to determine the correct row in an account database. The provider would make its own call to this database, using special credentials to retrieve the account holder and compare it to the Principal. This solution might require some manual mapping of account holder values to Principal values, but it would work. The dotted line from the authorizer to the external database in Figure 4 illustrates such a solution.

Note that you would have to write almost the exact same code to perform this check in the Account object itself. However, the service provider approach partitions the security logic and the business logic. Different people can maintain the two types of logic and


changing one does not introduce the risk of potentially breaking the other. This flexibility is important if you consider the actual complexity of banking applications that handle minor accounts, joint accounts, and business accounts. Maintaining the security policies for such an application could be a full time job in and of itself.

Conclusion

The WebLogic Server 7.0 Security Framework doesn't impose a rigid security model that hinders security integration with other system elements and forces the costly workaround of mixing security code with business logic. Instead, it adopts an open processing model so that application components can seamlessly cooperate with the rest of the enterprise security ecology. Moreover, its processing model delivers a clean abstraction of policy enforcement from business logic that lowers the cost of administering security policies and decreases the chance of security breaches.

Both application developers and security administrators benefit from the Security Framework. Developers no longer have to shoulder the responsibility and potential embarrassment of mixing application and security code. Administrators don't have to become experts in middleware paradigms to meet security requirements. When someone has to write special security code, he or she only has to do it once - everyone can use it and it's easy to maintain.

"The goal of the WebLogic Security Framework is to deliver an approach to application security that is comprehensive, flexible, and open"

The key to the Security Framework's benefits lies in its open service provider model. Third-party security vendors can easily integrate their solutions with WebLogic and enterprises can quickly create custom security modules. Most important, an open model means that enterprises do not have to wait for the middleware vendor to adopt new security technologies because there are plenty of hooks for future innovations. 

PANACYA
www.panacya.com



BY BRIAN WALCK
VADIM ROSENBERG

AUTHOR BIO...

Brian Walck is a director of product management at Cisco Systems. His responsibilities include developing products for Cisco's content switching product line, as well as developing core strategies in content billing and provisioning. He sits on the Network+Interop advisory board, and has authored numerous white papers.

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

CONTACT...

bwalck@cisco.com
vadimr@bea.com

Article reproduced here with permission of CISCO SYSTEMS, INC.



MAXIMIZING PERFORMANCE, AVAILABILITY, AND SECURITY OF BEA WEBLOGIC CLUSTERS

Through advanced clustering capabilities, BEA WebLogic Server-based e-business applications can be scaled across multiple servers. (Note: WebLogic Server supports multiple types of clustering, only one of which is relevant here – what is referred to as Web Clustering. In Web Clustering, the clustering of the HTTP or presentation layer of the Web application is addressed. This is what is referred to here.) Availability is enhanced by replicating application components and their state, as well as client session state information. In the event of the failure of a server mid-session, the client's session can be restored on another server without loss of session data. The WebLogic Server cluster makes this functionality transparent to clients who view the cluster as a single "virtual server."

As software infrastructure evolved to include specialized applications servers optimized for the delivery of Web content, the network infrastructure also evolved to become "content aware." Cisco Systems is an early pioneer in the development of content switching, a category of

network devices which are designed to optimize the delivery of Web-based applications. These high-performance networking devices understand the nature of Web-based transactions and how to route them to the most capable server at any given moment in time. They are used to increase the scalability, performance, availability, and security of Web-based applications. This article describes how Cisco CSS 11000 Series Content Services Switches ("Cisco Content Switches") can be used to maximize the capabilities of BEA WebLogic Clusters.

WebLogic Clustering in Action

A single client transaction may require multiple HTTP operations to complete. A simple example of this might be the presentation interface for a Web-based stock trading portal. A client might go through several steps in executing a trade (getting quotes, doing research, placing the order, receiving a confirmation, etc.) The application keeps track of the "state" of this transaction through each of its subsequent phases. In order to provide protection from failure, this session state can be replicated to another server. In the event of failure of the primary server, the session can be reinitiated to a new server (either the server which hosted the state replica or a third server that recovers the session state from it). WebLogic supports three different

USING CISCO CONTENT SWITCHING TECHNOLOGY
TO MAXIMIZE END-USER EXPERIENCE

methods for replicating and recovering session state: database replication (via JDBC), file-based replication, and in-memory replication.

File and database replication are similar. Each server in the cluster maintains connections to a shared file server or database server. State information is written to a file or to a database record as it is created or changed. Upon failure of a server in the cluster, subsequent client requests are routed to another available member of the cluster (more on this shortly). The new server reads the session ID (established at the initiation of the session and stored in a cookie or encoded in the URL of the request) and fetches the associated state from the database or file system. The new server can now continue processing the client's transaction.

In-memory replication sends state information from a primary server to a designated backup, where it is maintained in memory. Upon failure of the primary server, the client will be routed to a new server. This new server will either have the state replica (in which case it designates a new server as the backup, creates a backup copy of the session state, and assumes the primary role) or it will fetch the state from the backup (in which case it assumes the primary role, cleans up the session state from the old backup, elects a new backup server, and creates a new state replica on the backup). Like the session ID, the backup server ID is encoded in the cookie or URL, allowing the new server to recover session state from the backup. The new server can now continue processing the client's transaction.

Request Routing

Because multiple servers in a cluster are capable of servicing a particular set of client requests, some mechanism must be used to route client requests made to the "virtual server" to one of the real servers in the cluster. The first, and simplest, goal of a request routing mechanism is stated as follows:

1. To balance load across the available servers in the cluster

For transactions that span multiple HTTP operations (and possibly multiple TCP connections), once a client session has been established with a particular server cluster member, subsequent operations should be directed to the same member until the session completes. This will reduce the overhead and latency associated with fetching session state from another server for each successive operation, thereby improving user response time and improving overall utilization of the cluster.

Therefore a second goal of request routing for WebLogic clusters is:

2. To maintain "persistent sessions" between clients and cluster members who are cooperating to perform complex transaction logic

Simple DNS-based load balancing available in public domain DNS servers such as BIND will allow the operator to configure multiple host addresses in association with a particular domain name (e.g., www.my.store.com). Clients attempting to access this site will be resolved to each of the configured addresses in "round robin" fashion. Clients will cache the returned name-to-address mapping for the time-to-live configured in the DNS server by the domain name owner. (Note: Although this is the specified behavior for DNS clients, actual behavior varies considerably by operating system and by browser.) Subsequent requests to the same host name will be directed to the same server (using the cached mapping) – thereby performing a crude type of session persistence. A major downfall of DNS-based schemes, however, is that they are incapable of routing around a server failure. Once a client is "bound" to a server, that binding remains in place until the time-to-live for the cached name-to-address mapping expires. Clients can, therefore, continue to attempt to contact a host even though that server has failed or been taken out of service. Clearly, then, something more than simple round-robin DNS is needed for a robust local request routing mechanism. (Note: DNS-based techniques, when deployed in conjunction with more robust local request routing schemes, are in fact very useful for disaster recovery and for balancing load between multiple geographically separate locations.) It is therefore appropriate to expand our list of request routing goals.

3. To rapidly detect and route around server and process failures

Proxy Web server-based request routing takes the approach of front ending the cluster with an additional server which accepts incoming client requests and directs them to the appropriate server. A BEA WebLogic Server acting as a proxy for third-party Web servers is an example of this approach. By participating in the clustering protocol, the proxy is able to detect failure of servers quickly and route around them. Session persistence is provided via examination of the session cookie. Because it is a full member of the cluster, the proxy understands the session cookie format and the associa-

tions between sessions and servers.

Because they are general-purpose application servers, however, these proxies are not optimized for the task of request routing and are, therefore, lower performance than specialized load-balancing appliances.

Load-balancing appliances are general-purpose PCs with software tuned for load-balancing applications. Much like first-generation routers, these general-purpose architectures use a single, centralized CPU for all load-balancing decisions as well as for packet forwarding. Additional functionality is added by utilizing off-the-shelf PC boards (such as SSL acceleration cards) and writing drivers to integrate them with the rest of the system. By focusing on a specific task, these appliances provide better performance, redundancy, and features than their proxy server-based counterparts. Ultimately, however, the hardware architecture of appliances becomes the limiting factor in the performance and scalability of the cluster. For enterprise-class applications, performance must be included as one of the goals for request routing. This goal may be stated as follows.

4. To scale in accordance with the anticipated client request volume and data transfer rates

In much the same way that general-purpose computing architectures became a limiting factor to scaling for first-generation routers, general-purpose PC architectures have become a limiting factor for request routing. Recognizing this, Cisco Systems pioneered the development of the technology now known as "content switching." The core intellectual property associated with content switches is a switching architecture that separates the processing of the control functions associated with request routing (such as server selection, session establishment, and health checking, etc.) from the processing associated with the packet forwarding functions of request routing (such as network address translation [NAT], TTL decrementing, MAC address replacement, etc.) This fundamental separation of function allows Cisco Content Switches to provide both superior features and function as well as superior performance compared to PC-based load-balancing equipment.

Cisco Content Switches Provide High Server Farm Availability

From a logical perspective, Cisco Content Switches are deployed in front of the WebLogic cluster, as shown in Figure 1. Client requests to the cluster are directed to

a Virtual IP address (VIP), representing the cluster to the outside world. A Cisco Content Switch receives connections and HTTP requests from the outside world and routes them to the appropriate member(s) of the cluster based on configured policies. These policies will take into account the failover and persistence mechanisms discussed earlier.

Cisco Content Switches provide a complete set of features for maintaining a high-availability WebLogic cluster, including switch redundancy, session state failover, and advanced health checking.

Switch Redundancy

Cisco Content Switches are typically deployed in redundant pairs. In the event of a failure of the primary switch, the secondary switch will take over. Depending on the configuration of session state redundancy, this failover may take place without disrupting the client-to-server connection. Certain configurations may use both switches as simultaneously active and backup for certain groups of servers or content types – a configuration known as “Active-Active” redundancy.

Session State Redundancy

For some applications, it is critical for the Cisco Content Switch failover to occur without disrupting existing client-to-server connections. For such applications, the content switches can be configured to maintain session state information on both primary and secondary switches. In the event of a failure, the redundant switch will forward packets associated with the existing connection between client and server as soon as the underlying network reconverges.

Advanced Health Checking

Cisco Content Switches use both active and passive techniques to monitor server health. By periodically probing servers, the content switch will rapidly detect server failures and quickly re-route connections to available servers. A large variety of health checking features are supported, including the ability to verify Web servers, SSL servers, Application servers, databases, FTP servers, streaming media Servers, and others.

Cookie Switching Technology Offers Flexible Session Persistence Options

As I previously discussed, for performance reasons, once the initial selection of a server within the cluster is made, it is important to keep connections from the

same client routed to the same server until the transaction is completed. This is referred to as “session persistence.” It is absolutely critical that the content switch leave the session cookies sent from server to client and from client to server intact. Cisco Content Switches also support two different methods for achieving session state persistence for WebLogic clusters.

Active Cookie Insert

When using this method, the content switch is configured to insert a unique cookie for each server when a new session is detected. This cookie is then sent by the client back to the server and used by the content switch to keep the session routed to the same server.

Cookie Matching

In this mode of operation, the content switch is configured to look for a specific string in the cookie that indicates the server which initially set it. This may either be a string that is uniquely identified in the session cookie for this purpose or a different cookie set by the WebLogic programmer. Although this method is potentially more complex from a programming perspective, it gives the Web application designer the most flexibility in controlling the application behavior. (*Note: Cookie matching can be used for numerous purposes, including dynamic load shedding and third-party routing. These techniques will not be discussed here.*)

Cisco Content Switches Enhance Site Security

Cisco Content Switches can protect the WebLogic Cluster in four different ways.

Access Control Lists

By constructing access control lists on content switches, the operator can control who has access to the real IP addresses of the cluster members as well as who has access to the switches themselves.

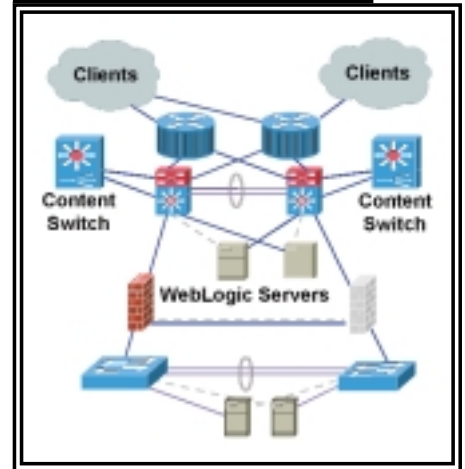
Network Address Translation (NAT)

Cisco Content Switches perform NAT from the Virtual IP address (which represents the cluster to the outside world) to the real IP addresses of the cluster members. This allows the server cluster members to be numbered using private IP address space. More importantly, it hides the details of the internals of the cluster configuration.

Denial-of-Service Protection

Because Cisco Content Switches partici-

FIGURE 1



Deployment of BEA WebLogic Servers and Cisco Content Switches.

pate in both TCP and HTTP, they are in an ideal position to detect and stop TCP-and-HTTP-based denial of service attacks – before they can impact a server.

Firewall Load Balancing

If the application generates enough traffic to warrant additional firewalls, Cisco Content Switches can be used to load balance multiple firewalls.

SSL Acceleration Improves Performance and Enables Persistence

Running SSL on WebLogic Servers is a tremendous drain on server resources. Cisco SSL Acceleration technology (available in Cisco CSS 11500 Content Switches and Catalyst 6500 switches) can offload SSL processing, enabling server resources to be focused on value-added WebLogic functions. In addition, as persistence information used by the Cisco Content Switches is inside the HTTP header, it is no longer visible when carried inside SSL-encrypted sessions. Cisco SSL Acceleration technology terminates these sessions prior to applying content switching decisions, enabling all of the persistence options previously discussed to become available for secure sites.

High Scalability and Performance Options

In addition to raw switching performance, Cisco Content Switches provide capabilities that can be leveraged to scale the performance of WebLogic Servers as well as the performance of client to Server connections.

continued on page 35

Web Services

.NET J2EE XML JOURNAL

LEARN WEB SERVICES. GET A NEW JOB !

Subscribe today to the world's leading Web Services resource

The Best
.NET
Coverage
Guaranteed!

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*



Component-Level Performance Monitoring with Dirig PathFinder

DIRIG SIMPLIFIES WEB APPLICATION PERFORMANCE MANAGEMENT



Reviewed by Ralph Decker



Company information:

One Indian Head Plaza,
6th Floor
Nashua, NH 03060
603 889-2777

E-mail: info@dirig.com

Sales, service, and presales information:
603 889-2777, ext. 885

E-mail: sales@dirig.com

Test environment:
Windows 2000 running
BEA WebLogic 6.1,
Oracle9i Enterprise edition;
Apache WEb Server

Complex, distributed Web applications have been the source of headaches for many Web managers. In the past, pinpointing performance issues and transaction failures has been a tedious process. Monitoring tools did a good job identifying whether a network application was slow, but did a terrible job of helping Web developers and IT managers identify which component was causing the slowdown. In August 2002 Dirig Software released BEA WebLogic support of its flagship performance management tool, Dirig Agent with Fenway Management Extensions (FMX), which gave IT managers a component-level view of complex Web applications. KeyLabs tested Dirig's latest FMX add-on tool, which creates a topographical map of the path a transaction takes through a Web application.

Web Application Home Run

Dirig PathFinder 1.0 is an add-on product to Dirig Agent with FMXplus version 3.5 that helps IT managers identify performance problems in distributed Web applications. Dirig PathFinder is

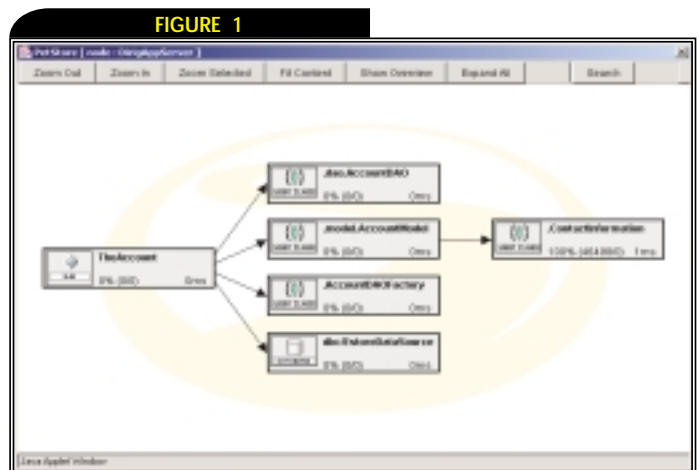
an analyzing engine that parses J2EE applications and dynamically graphs a transaction's path through a Web application. It automatically discovers the possible paths that transactions could take through an application's infrastructure. Working in much the same way as a network-mapping tool that discovers network switches and routers, Dirig PathFinder is an application-mapping tool that discovers JavaServer Pages, Enterprise JavaBeans, and Java Database Connectivity.

Dirig PathFinder automatically discovers components, component relationships, and component dependencies in J2EE applications. When used in conjunction with FMXplus, Dirig PathFinder automatically instruments the applications to provide configuration files that report number of calls, performance statistics, and response times, as well as exceptions and failures of individual methods. It is a complete management solution for your J2EE Web services, running from high-level process monitoring down to the method level performance and failure analysis.

Out of the Ballpark

In addition to monitoring FMXplus-instrumented applications, Dirig PathFinder also incorporates external performance metrics. For example, Dirig PathFinder will track external services and protocols and provide system statistic monitoring such as disk access, memory metrics, CPU utilization, and swap file statistics. In fact, log file monitoring can be configured to use any ASCII-based log file, and vendor API support can be used to monitor other vendor applications.

Dirig has created a number of Specific Application Managers (SAMs) that allow QA professionals to set policies, reports, and custom alerts for applications such as Oracle, Microsoft SQL Server,



Dirig Pathfinder creates a visual map of Web application components showing component dependencies

Microsoft Exchange, MySQL, IBM DB2, Sybase and Apache Web servers. SAMs allow users to download and import predefined rules allowing users to quickly manage applications with no additional coding or modifications

Dirig PathFinder gathers data from various sources by talking directly to network data stores. For example, FMXplus's sampling facility will collect from the Windows Registry, from any database using ODBC, and from application APIs. Additionally, protocol simulation can verify that applications and services are running, and confirm response times using a myriad of protocols (e.g., PING, ECHO, HTTP get request, SMTP, POP3, IMAP, DNS, FTP, DHCP, LDAP, etc.). These allow IT managers to correlate data from not only the monitored application, but from the network infrastructure, operating systems, and Web servers as well.

In the Dugout

With so much data coming from so many different sources, it is important to have a console that can organize and present the information logically. Dirig PathFinder's Management Server performs this job. Once installed on a separate box, the management server makes sense of all the information that is collected via a Web interface. All data can be viewed in real time, or historical reports can be generated to help resolve application performance problems or failures. The management server allows the user to configure the agents, set up alerts, observe data collected from many agents, define reports, and view real-time performance statistics.

Dirig PathFinder installs an agent on each server that is to be monitored. Currently supported operating system platforms are Sun Solaris, IBM AIX, Linux, Hewlett-Packard HPUX, and Microsoft Windows. The agent can be managed and configured from the management console using a Web browser or can be

configured from a command line. Dirig software also integrates with network management systems including Aprisma Spectrum, Micromuse NetCool, HP OpenView, and SNMP.

We set up a BEA WebLogic test environment to evaluate FMXplus with Dirig PathFinder. Using Windows 2000 and WebLogic as the application server and Windows 2000 and Oracle as the database server, we ran load and stress tests against WebLogic's Pet Store sample application. We also ran additional tests using Apache as the front-end Web server, but leaving the rest of the test bed the same.

The Pet Store application was instrumented and monitors were configured for the operating system, hardware, and Oracle. As requests and transactions were made against the application, Dirig PathFinder showed the response times of the Pet Store application components. Dirig PathFinder made it easy to trace different transactions through the application.

We used the OpenSTA scripting tool to generate a 250-concurrent user load against the WebLogic application server. We were impressed with Dirig PathFinder's ability to let us drill down into the individual components that made up the Pet Store application.

The Score

Just as in manufacturing, managers need to know where the bottlenecks are occurring in an assembly line. Developers and QA professionals will like Dirig PathFinder's ability to assist in locating transaction bottlenecks. QA teams can use Dirig PathFinder to provide detailed statistics for the response times of each individual component. Problems in specific methods can be identified. Reports can be designed to show the most problematic components. All of the reports are customizable to fit the needs of the developer or

tester. Once identified and corrected, fixed code can be verified.


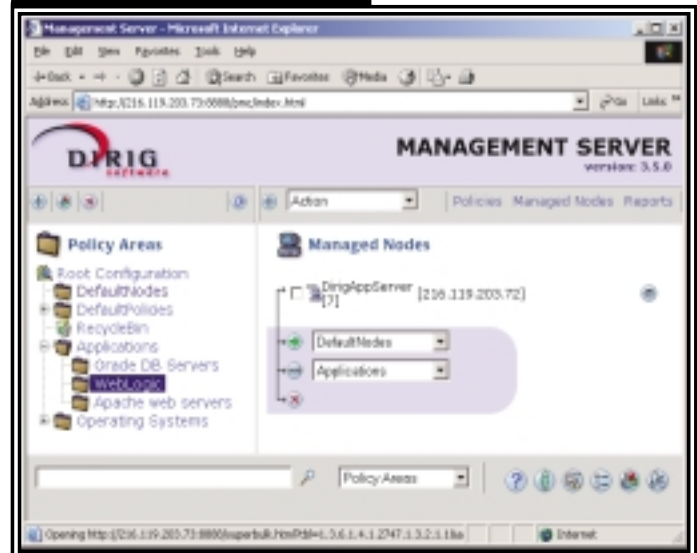
In a production environment, Dirig PathFinder provides invaluable notification and alerting. Thresholds can be configured to notify of failures and performance-related issues. Dirig PathFinder not only notifies of application failure, but also identifies the specific methods responsible for the loss of performance. PathFinder can be configured to restart services that fail or to launch executables or scripts to corrective actions. 

FIGURE 2



Dirig Management Server provides a convenient way to configure component performance metrics

AUTHOR BIO...

Ralph Decker is KeyLabs' Master Architect and Director of Research & Development. Ralph has over 13 years in software testing and QA, and has helped hundreds of clients design new testing methodologies and procedures for their own computer application software and hardware. In addition to his BA in languages, Ralph is a Certified Information Systems Security Professional CISSP.

CONTACT...

rdecker@keylabs.com



transaction MANAGEMENT

This month, I thought I would take a below-the-surface look at what needs to be done to achieve transactional access to the IBM MQSeries messaging product from WebLogic Server within the context of an XA transaction managed by WebLogic's JTA subsystem. Of course, from the outset I would like to note that WebLogic Server itself boasts a very capable and reliable messaging system – which is getting more capable with every release – but it is a fact of life that MQSeries for various reasons (most predating the existence of Java, never mind the JMS messaging standard) has a large installed base, so good interoperability with it is a requirement whatever your infrastructure religion.

To aid code portability, JMS provides for connection factories, queues, and topic objects to be defined prior to application runtime and stored using a JNDI provider, so using all these standards from within the WebLogic Server environment should be simplicity itself, right?

Well, almost. There are some wrinkles that come from MQSeries restrictions that derive from the implementation of MQ itself. These need to be coded around if WebLogic Server and MQ are to live together in perfect harmony. Most of the code posted on dev2dev is concerned with ironing out these wrinkles so that the two products can coexist in perfect harmony, or at least without shed loads of cacophonous discord, or as MC geek and the OLTP crew might say “so the MQ thing doesn't go in the crapper, BEA Systems have written a wrapper.”

Sending Messages to the Other Side

ENABLING INTEGRATION BETWEEN WEBLOGIC SERVER AND MQ

BY PETER HOLDITCH



AUTHOR BIO

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

CONTACT...

peter.holditch@bea.com

So Tell Me the Good News

Well, Okay, I will. The first piece of good news if you are looking for programmatic integration between WebLogic Server and MQ is that there is working code to achieve just that available on BEA's dev2dev Web site at <http://dev2dev.bea.com/code/codedetailcontent.jsp?productType=weblogic+server&codeType=alpha+code&filepath=components%2Fdev2dev%2Fcodelibrary%2Falpha+code%2Falphacodewlsmqseries.htm>. However, there are some features of its implementation that bear closer scrutiny to get further insight into what needs to be done to interface these two environments. If you're not into intricate details, I recommend that you proceed directly to the conclusion now for an early bath – I am about to start indulging my love of the obscure...

You're still here? Good, another implementation trivia lover – I can see that we're going to get on famously. Take a deep breath; I'm sure you're going to enjoy the trip!

Enjoy the Trip!

On the face of it, part of the MQSeries product is a Java wrapper that implements the JMS programming interface and supports the XA exten-

So What Is This Wrapper Up To?

Rather than a full top-down review of the whole thing, I will spend the rest of the article reviewing the purpose of a subset of it. This will hopefully provide a level of understanding both of the XA implementation of MQ, and of the WebLogic Server transaction manager.

The first part of the wrapper is a class that maps the MQ JNDI entires from an external JNDI directory into the WebLogic JNDI tree. On the way, it interposes its wrapped versions of the XQQueueConnectionFactory or XATopicConnectionFactory. The net result is that the set of connection/topic factories present in MQSeries becomes visible in the WebLogic Server environment, with a wrapper to do the rest of the integration magic.

The wrapper at the base of the remaining mystery is a wrapper for the MQSeries XA resources. As XA resources are dispensed from the MQ factories, they in turn are wrapped, and it is in this resource wrapper that the business end of things happens.

First, the constructor of the resource wrapper registers each XA resource – once per server – under a resource name that is either given or constructed from the server and domain names and the MQ resource string. The wrapper also allows the MQSeries integration to take advantage of an optimization in the WebLogic transaction manager, namely dynamic enlistment.

So, I hear you cry, what's that? The JTA specification allows only for static enlistment, which means that all resources must register their existence at startup, and from this point on the application server must assume that all resources registered with it participated in every transaction



and enlist them in JTA transactions on that basis so they are sent a commit for each transaction on the off chance that they participated. With dynamic registration, resources can enlist with JTA lazily as they are invoked in a transaction - meaning that the transaction manager knows the correct set to coordinate at commit time, improving performance. The resource name associated with a resource is used by WebLogic as a branch qualifier for the transaction. It is also used in recovery so that outstanding transactions can be completed from transaction log records. This means that these resource names must be assumed to be constant across multiple server reboots to allow for transaction recovery.

Finally, an XA resource provides a method called `isSameRM` that allows a transaction manager to determine conclusively if two participants in a transaction are really one and the same resource. The wrapper provides its own implementation of this method with different behavior from the out-of-the-box implementation from IBM. The reasons for this are slightly obscure, but here goes... IBM's implementation of `isSameRM` bases the decision on the `QueueManager`. The wrapper is designed to base the `isSameRM` decision on the name associated with the `XAConnectionFactory` (remember the significance of the resource name in terms of the transaction branch?). This allows multiple branches to be

used for a particular MQSeries QM. This is important when multiple MQ sessions are to be manipulated in a transaction. Due to the behaviors of the WL TM and the IBM JMS implementation, the direct use of multiple MQ sessions in a single transaction can result in XA errors. The IBM JMS requires that each XAResource instance, that is associated with a particular Session, be enlisted and delisted from the transaction. The WebLogic Transaction Manager implements a delayed delistment optimization, which delays calling `XA.end` (the XA call that performs the delistment) until absolutely necessary (transaction suspended, commit called, etc.). This optimization helps avoid multiple start/end calls when the same resource is accessed multiple times within a transaction. However, because each JMS session has a unique XAResource object, and their `isSameRM` methods will return true for each other, WebLogic's Transaction Manager will treat them as if they were the same. The first session accessed will receive an `XA.start` call to enlist it in the current transaction. When the JMS operation returns, the optimization will delay the `XA.end` call to delist the resource. When an operation is invoked on the second session, `XA.start` will not be called because the Transaction Manager thinks that it is the same resource and it knows it has already called `XA.start` on it. Following that, when the JMS call is made the IBM driver will respond with

a 2072 "Syncpoint not available" error because MQSeries didn't get the `XA.start` it was expecting - since as far as it is concerned this is a different resource - so it decides that the call is being made outside of a global transaction.

Come Up for Some Air, Here's The Really Good News

Phew, you can come up for air now.... This transaction stuff sure provides some late night entertainment. Or perhaps a cure for insomnia. It depends on your viewpoint I suppose.

The good news is that WebLogic Server versions greater than 6.1sp3 provide an out-of-the-box transactional MQ Series bridge that allows you to map WebLogic JMS queues and topics to MQ ones without writing a single line of code - just configure the source and destination queues to be bridged and away it goes. Even better news is that the next release preview bird has told me that a transparent JMS wrapper is in the works. This will allow you to configure foreign queues into the WebLogic environment where the server will transparently take care of all this wrapping.

Yet another reason why both above and below the J2EE standards surface WebLogic Server will remain ahead of the pack in terms of ease of use - bringing faster time to market to all who use it as a platform for their applications.

That's it from me for now - I'll delist until next month. Happy transacting. 🍷

FEATURE

continued from page 30

Server Farm Partitioning

Cisco Content Switches can partition components of a single Web application across multiple cluster members. For example the following two URLs: <http://www.mycompany.com/quotes/getquote.jsp> and <http://www.mycompany.com/trades/order.jsp> could be located on two different servers even though the domain name is the same. This allows the application developer to easily scale the application to multiple servers without many code modifications. Furthermore, it maximizes the cache coherency (and thus, performance) of the servers by keeping requests for the same pages on the same servers.

Additionally, Cisco Content Switches may be used to push requests for cacheable content (such as image files) to a set of caches which may serve them more cost effectively than the application servers themselves.

HTTP 1.1 Connection Remapping

When using HTTP 1.1, clients may request multiple URLs over the same TCP connection. This allows the client and server to run more efficiently by reducing the overhead of TCP connection maintenance.

When content is partitioned across multiple servers and/or caches as described in the preceding section, it is important to be able to send multiple HTTP GETs for different pieces of content that might arrive over the same HTTP 1.1 connection to different servers. Effectively, the client side HTTP 1.1 connection must be "remapped" from one server to another. Cisco Content Switches perform this function to maximize connection efficiency.

Summary

BEA WebLogic Servers include advanced clustering capabilities to scale applications. Cisco Content Switches maximize the performance, availability, and security of BEA WebLogic Server clusters by understanding the nature of Web-based transactions and routing them to the most capable server at any given moment in time. This combined solution results in maximum end-user experience.

Reference

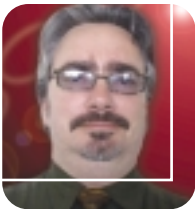
- www.bea.com/products/weblogic/server/paper_wls_clustering.pdf 🍷

This is the first in a series of articles from the Office of the CTO at BEA Systems. As my main area of expertise and interest is application architecture, my role within the CTO's office allows me to explore how BEA's customers and products interact around applications – architecture, development, and integration..

Importance of Application architecture

THE KEY TO A HEALTHY DEPLOYED APPLICATION

BY GORDON SIMPSON



Over 10 years ago I was “promoted” to the title of application architect and the word “architect” got me thinking. What does an application architect do? Remember, the title wasn't as in vogue as it currently is. I came up with the following definitions – they made sense at the time and still seem to today.

This article is a general introduction and not a master class. I'm going to focus on when application architecture is useful and what it gives us. The specifics of “how to” and “goodness” will be very interesting, too, but we'll have to cover that at another time.

OK, What Is It and Why Do I Need It?

Simply put, architecture is big picture design. The architect's job is to do abstract structural design for a specific purpose. The goal is to ensure that the form and function of a particular structure is appropriate today and remains so throughout its expected lifetime.

In our applications world, an architect needs to create levels of design that span multiple business problems and continue to deliver appropriate behaviors as technology and business requirements evolve. The result is a framework that will enable the business applications to be reformed over time. The value of architecture is not some magic formula, but to provide a structure to support change.

Architecture can be applied in various “levels” and at various times. What follows is the basic route to success but, as with many things, your mileage may vary. What I hope you come away with is what we should be thinking about and why.

Enterprise Architecture – To Infinity and Beyond

The goal of successful enterprise architecture is to explore the entire business and define an application and infrastructure framework that has the potential of delivering workable solutions for the foreseeable future.

Is that all? Oh yes! It's also good to remember that many systems developed in the '80s are still in our active inventory. So balance is required – not too shortsighted but not too much detail.

The key is to identify the business aspects that are core and the others that might change significantly. This will allow us to frame the risk when looking at the specific areas to support. With a solid business perspective, current technologies and future science can be assessed. Although new technologies might stimulate new business, technologies are the tools, not the goal – business is the key.

The results should support business growth or shrinkage, and replacement of application and technology components over time. Change is a constant – the architecture's aim is not just to withstand it but also to enable it. The exact structure is not important but the focus must be correct and the framework must be appropriately flexible to evolve. The enterprise architecture will also provide the framing and guidance for the next levels of architecture and design.

Technical Architecture: Infrastructure By Any Other Name Might Be Plumbing

Those aspects of technology that cross business solutions, and can be framed separately, are what I think of as technical or infrastructure architecture. This area of “architecture” benefits most from the enterprise work above. The high-level framing provides a focus that is hard to synthesize out of a collection of projects. Without it there is a risk of “infrastructure for infrastructure's sake.” Infrastructure tends to be the domain of the engineers and should not be focused on “the” right answer. The requirements of a good technical architecture, the diversity of applications to be supported, are such that a single right answer is unrealistic.

A better goal is a set of solutions that support varying degrees of “perfection” at well-understood cost and risks. The engineering “focus” can be a

AUTHOR BIO...

Gordon Simpson is senior director of technology in the Office of the CTO for BEA Systems, Inc. Gordon plays a key role in directing BEA's enterprise application platform strategy. He has been involved in all aspects of enterprise applications development since the mid-70s.

CONTACT...

gsimpson@bea.com



powerful tool in defining the solution sets but it must be tempered with the architect's broader perspective. Success is not about technology but about supporting secure and robust business applications at the appropriate cost and time.

Domain Architecture: The Few vs The Many

Domain architecture focuses on one area of the business and, because of its focus, results in a more specific design. Think of it as building a bank within a skyscraper. The original building architect had to allow for floors to support the weight of a vault and a large open area for the business floor. The

OUR CAST

The Star

Business user: An old chestnut but applications are for the business and not IT. Architecture is about making sure they are happy today and in the future. They are the ones with the real problems to solve.

The IT players

Scientists: The scientists spend their time working on Information Science, i.e. IT in the purely abstract, looking at ways to exploit other sciences that might or might not deliver something to our everyday IT lives.

Engineers: Engineers are those who study the science, deeply understand it, and then apply it as abstract, or at least non-specific, problems. For example, an engineer strives to understand the bandwidth capabilities of a particular network technology, not how it solves a particular problem.

Craftspeople: The majority of the IT world – they build, construct or assemble applications. The development of a successful application is a craft, a set of skills learned and honed over time by experience. You need experience, plus a strong understanding of the engineering, plus knowledge of how your specialty interacts with others. Pretty much everyone we think of as an architect or engineer is probably a craftsman.

The Featured Performer

Architects: An architect provides a detailed framework in which solutions can be delivered. Needs a rich knowledge of the technology, the engineering and the various crafts. Must understand the business needs of the entity they are designing for. Cannot live in an ivory tower but must stay engaged through construction. Architects can usually be identified by their worried expressions. The proof of a good architecture may not be seen for many years but the benefits have to be shown immediately. 🍷

bank architect could then work within those constraints and focus on the business needs of the branch: automated tellers, security cameras, and so on.

The goal is still to create a long-lived framework; the specifics of the contemporary solution can be resolved by further design. Analysis of the business needs driving the architecture must include interaction with other business entities, today and in the future. The product of poor domain architecture is "silos," the focus of many a project for the last few years.

The real challenge is to support the business domain to the maximum while remaining a good enterprise citizen. Pragmatically, this is the majority of architecture so it is essential that it be done as architecture and not as a project design – the right level of detail and vision.

Project Architecture: Are You Sure We Have Time For That?

Actually, that's a good question. By the time we get to individual projects, the development of a forward-looking framework seems a little like overkill. We need good design and hopefully that is done within the scope of an existing architectural framework. So we should really discuss the role of an architect at the project level.

Projects are concrete, they have rigid requirements, and they require compromise. It is the role of an architect to bring the broader perspective, architecture, to the construction process. If no architecture exists, then it becomes essential for the architect to provide as much vision and breadth to the design as the timelines will allow. The role of an architect on a project is key if the resulting deployed application is to be both robust and well - behaved. While architecture is about the bigger picture, architects should be ready to get engaged on the details.

So What Next?

Hopefully this has given you a useful and pragmatic perspective on the value of architecture. You probably noticed that J2EE or Web services didn't come up. We architect applications for business needs and technology's role is a supporting one. We should consider them in our process as they might open up new business requirements. It would have been appropriate to talk about them as examples of prefabricated infrastructure, but in my experience it's not a smart idea to base architecture on contemporary technologies.

To be continued... 🍷

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!



Now in More than 5,000

Bookstores Worldwide – Subscribe **NOW!**

Go Online & Subscribe **Today!**

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



WebLogicDevelopersJournal.com

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





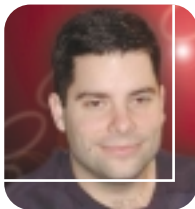
WEB SERVICES

WebLogic Server provides mechanisms to expose standard J2EE components and Java classes as Web services. It typically processes an RPC-style Web service by determining the operation to invoke from the SOAP message, deserializing the operation's parameters in the SOAP message to Java, and then invoking the back-end component.

Using SOAP Message Handlers with WebLogic 7

A STANDARD, FLEXIBLE SYSTEM THAT CAN SAVE YOU TIME

BY MICHAEL GILBODE



AUTHOR BIO...

Michael Gilbode is an architect for Anexinet Corp. in Philadelphia, and is experienced in large-scale J2EE projects built on the BEA WebLogic Platform.

CONTACT...

mgilbode@anexinet.com

Sometimes, however, this behavior is not sufficient and a Web service needs access to the SOAP message. SOAP message handlers can intercept SOAP messages in both the request and response to do additional processing of the SOAP message. SOAP message handlers are defined by the Java API for XML-based RPC (JAX-RPC). Handlers enable programmatic access to the SOAP message and are often used to process SOAP message headers. SOAP message headers may contain information about such things as transactions, security, and routing. Processing specific to these headers can be done in SOAP message handlers. Handlers can also be used to add functionality such as caching, encryption and decryption, logging, and auditing to a Web service. Handlers may also be used to process SOAP attachments.

Each WebLogic Web service consists of one or more operations. Operations can be implemented using a combination of back-end components and SOAP message handlers. WebLogic supports Web service operations that are implemented with back-end components only, back-end components and SOAP message handlers, and SOAP message handlers only.

SOAP message handlers are assembled into handler chains – ordered lists of SOAP message handlers that can be applied to the service client or the service endpoint. A WebLogic Web service can optionally define one or more handler chains. These handler chains are then associated with appropriate Web service operations.

Internally, WebLogic Web service operations are implemented as handler chains. The handler chain begins with an authentication handler, followed by any user-defined handlers, then optionally by a component invocation handler that can invoke a back-end component.

Message Handler Life Cycle

SOAP message handlers must be implemented as stateless instances. The JAX-RPC runtime system considers all instances of a particular handler class to be equivalent. This allows the JAX-RPC runtime system to pool handler instances, although this is not required.

The life cycle of a handler consists of two states. A handler either does not exist, or is in a ready state. A handler instance has two life-cycle methods. The `init()` and `destroy()` methods allow initialization of the handler when the handler instance is created, and cleanup when the handler instance is destroyed.

Once a handler is in its ready state, its various `handle()` methods can be invoked. On a service client the `handleRequest()` method is invoked before the actual SOAP message is sent, and the `handleResponse()` method is invoked before the actual response is received. On a service endpoint, the `handleRequest()` method is invoked before the actual service endpoint is invoked, and the `handleResponse()` method is invoked before the actual response is sent to the client. On a service client the `handleFault()` method is called before receiving a SOAP fault, and on the service endpoint the `handleFault()` method is called before generating a SOAP fault.

Designing the Message Handler

When you design SOAP message handlers you must decide the number of handlers, the order in which the handlers will be executed, and whether to invoke a back-end component.

Any arbitrary number of handlers can be configured on both the service client and the service endpoint. The order in which handlers are invoked is significant. For example, consider a handler chain on a service endpoint that is configured with an encryption handler and a caching

handler that inspects the SOAP message payload. In order for the caching handler to be able to read and process the message, the encryption and decryption handler must have already decrypted the SOAP message.

A SOAP operation in a WebLogic Web service may or may not be designed to invoke a back-end component. However, a message handler can change this behavior at runtime. A handler can prevent back-end components from being invoked by returning false from its handleRequest() method. In fact, this prevents any of the rest of the handler chain from being invoked. Further processing of the handler chain can also be prevented by returning false from the handleResponse() and handleFault() methods. For example, consider a caching handler. If the incoming request has cached results, the cached results could be returned and further processing of the handler stopped. The following code shows an example of this:

```
public boolean handleRequest(MessageContext context) {
```

```
    SOAPMessageContext smc =
    (SOAPMessageContext)context;
    if (Cache.exists(smc.getMessage())) {

smc.setMessage(Cache.get(smc.getMessage()));
        return false;
    }
    return true;
}
```

SOAP message handlers in a handler chain can also share state with each other. Each of the handle() methods in a handler class accepts a MessageContext as a parameter. The MessageContext instance that is passed to the handle() methods is shared among all handler instances in the handler chain for a single request, response, or fault processing on a service endpoint. Listing 1 shows two handlers that share a property named "myProperty". Handler1 sets this property, and Handler2 uses this property.

Features such as aborting the handler chain processing by returning false in a handle() method and sharing state among handlers introduce interdependencies between

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



Listing 1

```
public class Handler1 implements Handler {
...
public boolean handleRequest(MessageContext context) {
    context.setProperty("myProperty", "myValue");
    return true;
}
...
}
public class Handler2 implements Handler {
...
public boolean handleRequest(MessageContext context) {
    Object value = context.getProperty("myProperty");
    return true;
}
...
}
```

Listing 2

```
<handler-chains>
<handler-chain name="MyHandlerChain">
  <handler class-name="example.LogHandler">
    <init-params>
      <init-param name="logDirectory" value="c:/temp" />
      <init-param name="severityLevel" value="verbose" />
    </init-params>
  </handler>
</handler-chain>
</handler-chains>
```

Listing 3

```
QName serviceName = new QName(TARGET_NAMESPACE, "LogHandler");
QName portName = new QName(TARGET_NAMESPACE, "LogHandlerPort");
List handlerChain = new ArrayList();
Map logConfig = new HashMap();
logConfig.put("logDirectory", "c:/temp");
logConfig.put("severityLevel", "verbose");
handlerChain.add(new HandlerInfo(LogHandler.class, logConfig, null));
Service service = factory.createService(serviceName);
HandlerRegistry registry = service.getHandlerRegistry();
registry.setHandlerChain(portName, handlerChain);
```

**IT'S JUST
A CLICK
AWAY!**



**SYS-CON
MEDIA**



handlers in a handler chain. These interdependencies should be planned for and designed in advance because they affect how the SOAP operation is processed.

Implementing the Message Handler

A JAX-RPC handler must implement the `javax.xml.rpc.handler.Handler` interface. To make handler development easier, JAX-RPC provides the `javax.xml.rpc.handler.GenericHandler` abstract class. This class provides default implementations of the life-cycle methods and the different handle methods. Only the methods required for the specific handler implementation should be overridden.

The life-cycle methods of a handler should be used to initialize and clean up resources within a handler instance. If a handler makes use of a resource such as a database connection, this can be established in the `init()` method and cleaned up in the `destroy()` method.

Handlers can also obtain configuration information when they are initialized. This configuration information is passed as a part of the `HandlerInfo` object that is passed to the `init()` method of the handler instance. For example, a logging handler may be given information such as severity level and log directory in its configuration. This would be obtained as follows:

```
public void init(HandlerInfo handlerInfo) {
    Map handlerConfig =
handlerInfo.getHandlerConfig();
    String severityLevel =
(String)handlerConfig.get("severityLevel");
    String logDirectory =
(String)handlerInfo.get("logDirectory");
}
```

The `handleRequest()` and `handleResponse()` methods may be invoked in a different context depending on whether the handler is deployed on a service endpoint or a service client. This may be important for handlers such as encryption handlers. On the service client the `handleRequest()` method should encrypt the message and the `handleResponse()` method should decrypt the message. However, on a service endpoint, `handleRequest()` should decrypt the message, and `handleResponse()` should encrypt the message. This can be handled by creating client handlers and service handlers. However, since the behavior is essentially the same, this can also be accomplished by passing configuration information to the handler to tell it

whether it is on a service client or service endpoint. (Source code listing `EncryptDecryptHandler.java` shows an example of this; the source code for this article can be found at www.sys-con.com/weblogic/sourcecode.cfm).

Configuring the Message Handler

WebLogic Web services are configured using a deployment descriptor file. This file is named `web-services.xml` and is deployed in the `WEB-INF` directory of the WAR file that is hosting the Web services. Web service operations, back-end components that implement the Web service, and handler chains associated with the Web service are specified in this file.

This file is relevant to SOAP message handlers in that it is used to define handler chains, pass configuration information to handlers, and associate handler chains with Web service operations. Handlers and handler chains are defined in the `handler-chains` section of the Web service deployment descriptor. The `handler-chains` section contains one or more handler-chain definitions, each of which contains one or more handler definitions. Listing 2 is an example of a `handler-chains` section of a `web-services.xml` file. (A complete example is shown in the source code file `web-services.xml`.) This deployment descriptor defines a single handler chain named "MyHandlerChain" that contains a single handler. The information contained in the handler elements is used to create the `javax.xml.rpc.HandlerInfo` instance that is passed to the `init()` method of the handler instance. The parameter names and values defined in the `init-params` section of the descriptor are put into a `Map`, which is available through the `getHandlerConfig()` method of the `HandlerInfo` instance. This allows a handler instance to obtain configuration information at deployment time.

Using SOAP Message Handlers on the Client

The `web-services.xml` file is used to configure message handler chains and associate them with SOAP operations on the server. However, handlers can also be used on the service client. The JAX-RPC specification does not specify any standard packaging and deployment model. This work is currently under development in the J2EE 1.4 specification and JSR 109. Since there is no current standard client container for JAX-RPC Web services, SOAP message handlers must be configured programmatically on the service client.

In order to associate a handler chain with

a Web service, the developer must programmatically create the handler chain. A handler chain is simply a list of `HandlerInfo` objects. Listing 3 shows how to configure a log handler on a client. (A complete listing is shown in the source code `Main.java`.)

"SOAP message handlers provide a standard, flexible method for implementing Web Services"

Packaging the Web Service

WebLogic Web services are packaged as standard J2EE applications. The following steps should be taken to assemble a WebLogic Web service that uses message handlers:

1. **Create `web-services.xml` file**
2. **Compile handler classes**
3. **Build Web application:** Place `web-services.xml` file into `WEB-INF` directory of the Web application and the handler classes into `WEB-INF/classes`.
4. **Build an enterprise application:** The Web services Web application should be included in the enterprise application.
5. **Deploy the enterprise application to WebLogic Server:** The source code file `build.xml` shows an example of an Ant script to assemble a Web service that invokes a handler chain and a that component.

Summary

SOAP message handlers provide a standard, flexible method for implementing Web Services. They allow additional processing when exposing a back-end component as a Web service without requiring modification of the back end component. SOAP message handlers can be very useful when implementing WebLogic Web services.

Reference

- **SOAP 1.1 specification:** www.w3.org/TR/2000/NOTE-SOAP-20000508
- **JAX-RPC 1.0 specification:** <http://java.sun.com/xml/downloads/jaxrpc.html#jaxrpcspec>
- **JSR 109 – Implementing Enterprise Web Services:** [ftp://www6.software.ibm.com/software/developer/library/ws-jsr109-proposed.pdf](http://www6.software.ibm.com/software/developer/library/ws-jsr109-proposed.pdf)
- **JSR 151 – J2EE 1.4:** http://java.sun.com/j2ee/j2ee-1_4-pfd-spec.pdf

THE LARGEST WEB SERVICES, JAVA, XML AND .NET CONFERENCE AND EXPO IN THE WORLD

Boston
2003

London
2003

Berlin
2003

Hong Kong
2003



Register by
February 14th
SAVE UP TO \$400
Register by
March 14th
SAVE UP TO \$200

3rd Annual

Web Services Edge

Conference & Exposition



Connecting the Enterprise with Web Services, Java, XML & .NET



March 18-20, 2003
Hynes Convention Center
Boston

**Call TODAY
TO REGISTER**
201-802-3058
[www.sys-con.com/
webservicesedge2003
east/registernew.cfm](http://www.sys-con.com/webservicesedge2003east/registernew.cfm)

Featuring FREE Tutorials, Training Sessions, Case Studies and Exposition

Special Insert: Web Services Edge East Conference & Expo

MEDIA SPONSORS



OWNED BY
SYS-CON MEDIA

PRODUCED BY
SYS-CON EVENTS

Special Insert: Web Services Edge East Conference & Expo

Featuring FREE Tutorials, Training Sessions, Case Studies and Exposition



We are pleased

to bring the latest edition of the very successful Web Services Edge Conference & Expo to the city of Boston this March 18-20, 2003. Now in our third year, we will continue to build on our past success to make available the most current and relevant information to you, our valued attendee.

With the widespread adoption of Web services across the industry, developers are facing new challenges. In this year's conference program, we will address these challenges with our most comprehensive program to date. Web Services Edge 2003 will provide practical approaches and solutions to overcome the hurdles in developing and deploying Web services in today's competitive markets.

Once again Web Services Edge will feature four distinguished tracks - Java, Web Services, .NET, and XML - along with the newly added Vendor Track. The Vendor Track will allow specific sponsors, along with their customers, a platform to present their latest technical developments and real world applications across all the related technologies.



Your three days will include highly informative keynotes, conference sessions, tutorials, industry-leading university certification programs, case studies, and demo presentations. The Expo Hall will be open on March 19 and 20, featuring the largest grouping of quality exhibitors prepared to field your questions and solve your development needs.

All the best,
SYS-CON Events

Features & Attractions

TO ALL CONFERENCE & EXPO REGISTRANTS

- **3 Days Packed with Education and Training**
- **5 Keynotes & 3 Panel Discussions**
- **60 Hard Hitting and Current Seminars**
- **FREE .NET with Russ' Tool Shed Tutorial**
- **Web Services & XML Tutorials**
- **Java University Certification Training**
- **HOT** Industry-Leading Certification Programs
- **NEW** Extended Learning: Evening Conference Session & Certification Classes
- **INFORMATIVE** Round Table Discussions
- Opening Day Welcome Reception
- SAMS Meet the Authors Hot Topics Lounge
- **COMPELLING** Case Studies & Best Practices
- **FEATURED** Product Demonstrations on the show floor
- **RIVETING** Real-time SYS-CON Radio Interviews

Keynotes and Highlighted Speakers

Dave Chappell

VP, Chief Technology Evangelist, Sonic Software
Dave Chappell is the vice president and chief technology evangelist for Sonic Software. He has more than 18 years of industry experience building software tools and infrastructure for application developers, spanning all aspects of R&D, sales, marketing, and support services. Mr. Chappell has also been published in numerous technical journals, and is currently writing a series of contributed articles for Java Developer's Journal.



Eric Newcomer

Chief Technology Officer, IONA

In the role of Chief Technology Officer at IONA, Eric is responsible for IONA's technology roadmap and the direction of IONA's Orbix E2A e-Business Platforms as relates to standards adoption, architecture, and product design. Eric joined IONA in November 1999, and most recently served as IONA's Vice President of Engineering, Web Services Integration Products. Eric is a member of the XML Protocols and Web Services Architecture working groups at the W3C and IONA's Advisory Committee representative to UDDI.org.



Simon Phipps

Chief Technology Evangelist, Sun Microsystems

Simon Phipps, currently chief technology evangelist at Sun Microsystems, speaks frequently at industry conferences on the subject of technology trends and futures. He was previously involved in OSI standards in the 1980s, in the earliest collaborative conferencing software in the early 1990s, and in introducing Java and XML to IBM.



John Magee

Vice President, Oracle9i Oracle

John Magee is Vice President, Oracle9i at Oracle. He has more than 14 years experience in the enterprise software industry and has held positions in product development, product management, and product marketing. In his current role, he manages technical product marketing for Oracle's application server and development tools products, and is responsible for evangelizing Oracle technology initiatives around J2EE, XML and Web services.



Event Sponsors

ORACLE

Microsoft



ALTOVA

Rational
the e-development company

COMPUWARE

REGISTER BY PHONE-201-802-3058



SUN MICROSYSTEMS Java™ University Program

Web Services Programming Using Java™ Technology and XML

Tuesday, March 18, 2003
9:00 am - 5:00 pm

Who Should Attend

Web services designers and programmers, application developers and programmers using the Java programming language who have experience using the Java™ 2 Platform, Enterprise Edition (J2EE™).

Prerequisites:

Experience using the Java programming language and basic knowledge of XML.

Overview:

This one-day seminar provides in-depth knowledge of Web services and shows how to develop Web services using the Java programming language and XML, the technologies of portable code and portable data respectively. The session will start with an introduction on fundamental concepts and characteristics of Web services. This will be followed by a detailed explanation of how to implement, how to describe, how to register, how to discover, and how to invoke Web services using core Web services standards - Simple Object Access Protocol (SOAP), Web Services Description Language (WSDL) and Universal Description, Discovery, and Integration (UDDI). In addition, the ebXML standard, which defines the framework for the global electronic marketplace, will be talked about in detail. Also, the tools for building and deploying Web services will be discussed. Each topic will be presented with concrete examples and demonstrations when possible.

Attendees will also learn how to use standard Java APIs for Web services, mainly Java API for XML Messaging (JAXM), Java technology API for XML-based RPC (JAX-RPC), and Java technology API for XML Registries (JAXR), for developing and deploying Web services.

Benefits

- Learn the fundamental concepts and characteristics of Web services
- Gain detailed understanding on core Web services standards: SOAP, WSDL, UDDI
- Gain detailed understanding on ebXML, the standard framework for electronic business
- Learn Java programming language APIs for Web services - JAXM, JAX-RPC, JAXR

Outline

- Web services and Sun™ Open Net Environment (Sun ONE) overview
- Web services standards
- Java APIs for Web services
- J2EE technology and Web services

Java™ 2 Platform: Architect Certification Fast Path

Wed, March 19, 2003
9:00 am - 5:00 pm

Who Should Attend

This session is designed for enterprise application architects, system analysts, experienced technologists and developers using Java™ technology seeking certification as an architect for the Java 2 Platform, Enterprise Edition (J2EE™).

Prerequisites

Understanding the benefits of Java technology solutions; experience with object-oriented analysis and design; familiarity with concepts of distributed computing.

Overview

Gaining recognized competency architecting J2EE platform-based solutions is vital to your success as an architect and increases your career opportunities.

Developed and presented by Mark Cade, this one-day session helps prepare attendees to pass the Sun Certified Enterprise Architect for J2EE Technology exam. Cade provides an overview of the components comprising the J2EE architecture as a whole, emphasizes the incorporation of J2EE technology into an architecture, and reviews the exam's testing objectives. Multiple real-world case studies demonstrate correctly architected J2EE technology-based solutions and pinpoint key topics within the exam.

Additionally, you'll learn how to interpret exam objectives, what each of the three exam phases contains, and guidelines and resources to use after the course.

Benefits

- Receive an intensive review of the topics covered on the Sun Certified Enterprise Architect for the Java 2 Platform, Enterprise Edition Exam
- Increase understanding and knowledge of architecting solutions using J2EE technology
- Understand the system qualities: scalability, availability, extensibility, performance, and security
- Understand trade-offs of different architectural choices
- Describe the benefits and weaknesses of potential J2EE technology-based architectures.
- State benefits and costs of persistence management strategies
- Review case studies of J2EE technology-based architecture
- Review practice tests and questions

Outline

- Architect examination overview
- Part multiple choice
- Part assignment
- Part essay

Java™ 2 Platform: Programmer Certification Fast Path

Thursday, March 20, 2003
9:00 am - 5:00 pm

Who Should Attend

This session is designed for programmers who have some exposure to the Java™ programming language, and are ready to prepare for the Sun Certified Programmer for Java 2 Platform exam.

Prerequisites

Object-oriented software development experience and familiarity with the syntax and structure of Java technology-based development.

Overview

The development community recognizes that certified competency in developing solutions using Java technology is vital to productivity, reaffirms your value to your organization, and increases your career advancement opportunities.

This valuable session, developed and delivered by Philip Heller, author of the two leading Java technology certification preparation manuals and president of Philip Heller Associates, helps to prepare you for the Sun Certified Programmer for the Java 2 Platform exam. In a comprehensive one-day seminar, Philip provides code-level, detailed review of the Java skills and knowledge you need to confidently approach the exam.

Benefits

- Receive an intensive review of the advanced topics covered on the Sun Certified Programmer for the Java 2 Platform Exam
- Increase your understanding and knowledge of Java programming language syntax and structure
- Prepare for the exam by reviewing practice tests and questions
- Gain a strong understanding of Java technology fundamentals

Outline

- Operating on data
- Shifting
- Shallow and Deep Comparison
- The Literal String Pool



XML Certified Developer Fast Path

Tuesday, March 18, 2003
9:00 am - 5:00 pm

Audience

This tutorial is for programmers who have some knowledge of XML and related technologies and would like to pass the IBM Certified Developer Test 141 on XML and Related Technologies.

Prerequisites

Background in object-oriented programming and knowledge of Hypertext Markup Language (HTML). Exposure to XML and related technologies.

Overview

XML is the foundation of two important emerging

technologies: Web Services and the Semantic Web. XML expertise and certification is critical for developers who want to remain competitive in the current tight IT job market. The practice tests and questions in this course are specially designed to teach you XML essentials and the key concepts to successfully pass IBM® Test 141 on XML and related technologies.

Outline

- Well formed XML documents
- XML Infoset
- XML namespaces
- Document analysis and modeling
- Document Type Definitions (DTDs)
- XML schemas
- The SAX API
- The DOM API
- XPath and XSLT

- XSL Formatting Objects (XSL FOs)
- Formatting XML with CSS
- XLink and XPointer
- XML Encryption
- XML Signatures
- SOAP, UDDI, and WSDL
- XML architectures based on business and technical considerations
- Optimization and testing of XML applications

Presenter Bio: Joel Amoussou is Founder and Chief Learning Architect of XMLMentor. Joel is the author of the first XML training course specially designed to prepare developers for IBM® Test 141 on XML and related technologies. Joel has created XML content management applications for the aerospace, pharmaceutical, and publishing industries.

Featuring FREE Tutorials, Training Sessions, Case Studies and Exposition

Special Insert: Web Services Edge East Conference & Expo

Special Insert: Web Services Edge East Conference & Expo

Featuring FREE Tutorials, Training Sessions, Case Studies and Exposition

Conference at-a-Glance

	JAVA	WEB SERVICES	.NET	
TUESDAY MARCH 18	8:00AM – 4:00PM Registration Open			
	9:00 – 9:50AM	(JV1) Squeezing the Best Out of Java	(WS1) Security: SAML, WS-Security and related issues	(NT1) .NET Framework Overview
	10:00AM – 10:50AM	John Magee, ORACLE		
	11:00AM – 11:50AM	(JV2) Testing Your Java Using JUnit	(WS2) Web Services Management	(NT2) Introduction to ASP.NET
	12:00PM – 2:00PM	Break		
	2:00PM – 2:50PM	Panel - WS-I "A road Map for Web Services Standards"		
	3:00PM – 3:50PM	(JV3) Building/Deploying the Ant Way	(WS3) Web Services Integration	(NT3) Introduction to Web Services
	4:00PM – 4:50PM	(JV4) Unlocking the Secrets of JDK1.4	(WS4) Using Web Services to Integrate J2EE and .NET Enterprise Applications	(NT4) How To Build Mobile Solutions Using the Microsoft Mobile Internet Toolkit
WEDNESDAY MARCH 19	8:00AM – 4:00PM Registration Open			
	9:00AM – 9:50AM	(JV5) Java and .NET	(WS5) Combining BPM and BRM technologies: a major step towards corporate agility	(NT5) ASP.NET with Visual Studio.NET
	10:00AM – 10:50AM	KEYNOTE - Sun Microsystems, Speaker TBA		
	11:00AM – 6:00PM	EXPO OPEN 11:00 a.m. - 6:00 p.m.		
	11:00AM – 11:50AM	(JV6) To Not Swing is to SWT! The Swing Alternative	(WS6) Web Services Fundamentals: UDDI, WSDL, XML	(NT6) Best Practices for .NET Development
	12:00PM – 2:00PM	BREAK & EXPO		
	2:00PM – 2:50PM	Panel - Web Services & .NET		
	3:00PM – 3:50PM	(JV7) Talking Back to the Server; the SOAP Way	(WS7) Portals and Web Services	(NT7) Best Practices for ADO.NET Development
	4:00PM – 4:50PM	(JV8) Unlocking the Power of XML	(WS8) Web Services: Next Steps After the Hype	(NT8) Developing Pocket PC applications using the Smart Device Extensions for Visual Studio .NET
	THURSDAY MARCH 20	8:00AM – 4:00PM Registration Open		
9:00AM – 9:50AM		(JV9) Writing SOAP Services	(WS9) Web Services Best Practices	(NT9) How to Debug with .NET
10:00AM – 10:50AM		KEYNOTE - Microsoft, Speaker TBA		
11:00AM – 6:00PM		EXPO OPEN 11:00 a.m. - 4:00 p.m.		
11:00AM – 11:50AM		(JV10) Working with Data the JDO Way	(WS10) Web Services Startups: Telltales of the Future	(NT10) XML and Web Enabling Legacy Applications Using BizTalk
12:00PM – 2:00PM		BREAK & EXPO		
2:00PM – 2:50PM		PANEL - "The Future of Java", Moderated by Alan Williamson		
3:00PM – 3:50PM		(JV11) Enterprise: The Next Generation	(WS11) Web Services Interoperability: The Last Mile	(NT11) Migrating Visual Basic Applications to Visual Basic.NET
4:00PM – 4:50PM		(JV12) Moving Around the Limitations of J2ME	(WS12) Web Services Case Study	(NT12) How to Develop an End-to-End .NET-Connected Application

The Largest and Most Complete i-Developer Conference in the World

Special Insert: Web Services Edge East Conference & Expo

Featuring FREE Tutorials, Training Sessions, Case Studies and Exposition

XML	VENDOR
(XM1) XML - A Managers Guide	Session TBA
(XM2) OASIS Standards Update	XMLSPY 5 Altova
(XM3) A Definitive Introduction to XML Schemas	SOAP and Java - Parasoft
(XM4) XML in Print - XSL:FO	Session TBA
(XM5) XML in Financial Services	Session TBA
(XM6) Case Study: XML in Life Sciences Oracle	Pattern Driven Application Development- Compuware
(XM7) Using XML for EAI - Best Practices	Managing the Developer Relationship - Sun Microsystems
(XM8) Take XML with You: XML and Mobile Computing	Session TBA
(XM9) Analyzing XForms IONA	Session TBA
(XM10) XML Query	SOAP Security- Rational
(XM11) XPath & XSLT 2.0 BEA	Why Web Services Management? - HP
(XM12) Third Generation XML Tools	Session TBA

Conference Overview

Java Technology Track



The Java track has been specifically designed to allow you to squeeze as much information out of each session as possible. This track is

designed for the Java developer, and will be led by industry-leading speakers and authors. Not a track for the beginner or the novice, this track is designed for the experienced developer who wishes to catch up on the latest techniques and APIs.

The Java Track has been designed with you, the more experienced Java developer, in mind. We know you don't have a lot of spare time, and we've designed the track to ensure that your time is maximized and you are armed with all the necessary tools to take your development to the next level.

Microsoft .NET Track



Microsoft .NET represents a major evolution in how applications are developed deployed and managed on the Microsoft platform. The

.NET Framework gives developers an object oriented development environment for building all types of applications including desktop, client/server, dynamic web page, wireless devices, server based as well as complete support for XML Web Services and the related XML standards. The sessions in the .NET Track will give you a broad as well as deep understanding of the capabilities in the .NET Framework and how applications built on .NET are easily integrated with applications running in a heterogeneous environments including mainframe, UNIX and J2EE platform.

Web Services Track



The Web Service track is focused on issues and topics that are at the forefront of development efforts in Web Services. Although the current

specifications provide a minimum set of protocols, issues such as security, transaction management, service management and coordination remain in flux. This track presents some of the leading authorities in the field on these urgent topics and addresses all of the questions that currently concern designers, developers and consumers of Web Services.

XML Technology Track



Whether you're looking to understand different XML standards, application techniques, or development tools; or using XML to develop

the next generation of Web applications and services, the XML Track is your ultimate training, collaboration, and innovation ground. Sessions include fast-track, in-depth training on XML Schemas and XSL-FO. We will update you on standards development and offer a comprehensive review of the various technologies related to XML that are essential for today's IT manager. The XML Track is armed with real-world applications of XML in financial services, life sciences, enterprise and B2B integration, and mobile computing. We will discuss new developments around XForms, a recent W3C Standards which marks another era of standards based application development; XPath and XSLT 2.0 XML; and Query.

The XML Track explores the technology and standards, real-world applications, and trends which will set the course for the future.

Microsoft® FREE .NET Web Services Tutorial



Russ' Tool Shed

Wednesday, March 19, 2003

9:00 a.m. – 5:00 p.m.

Join Russ as he shows you how to use Visual Studio.NET

9-12:15 Intro to Web Services using VS.NET by Russ Fustino

One of the key ideas behind the .NET strategy is the concept of software as a service, or in short, Web Services. This session will explain what a Web Service is and provide an overview of its related technologies like XML, SOAP and UDDI. We will demonstrate how the .NET Framework makes it easy to implement them for new and existing applications. This session will also provide concrete best practices for building XML Web Services using Visual Studio.NET. We'll answer many common questions like: How will my Web Service scale? How can my XML Web Services enable interoperability with Web Services from other vendors as well as within my own organization? We'll delve into building highly reliable and secure Web Services. Also, we will discuss issues such as dealing with complex data types using WSDL (Web Services Description Language), as well as securing SOAP messages using encryption. We'll see how developers can use enterprise level XML Web Services to simplify customer solutions.

1-2:30 - Advanced Web Services Using ASP .NET Thom Robbins

This session we will explore some of the more advanced areas of SOAP in ASP.NET's support for Web Services. ASP.NET Web Services are the preferred way for Web developers to expose Web services on the Internet. The goal is quick, easy, and high-performing SOAP services. We will look at how to use the SOAP extension classes to create some very interesting applications on top of the core SOAP architecture found within .NET Framework. For instance, you can implement an encryption algorithm or screen scraping on top of the Web Service call. We'll dig into more advanced topics, explore the SOAP headers, and see ways to ensure security in our Web Services.

2:45-4:15 - .NET Remoting Essentials Thom Robbins

Microsoft .NET Remoting is the .NET technology that allows you to easily and quickly build distributed applications. All of the application components can be on one computer or they can be on multiple computers around the world. .NET Remoting allows client applications to use objects in other processes on the same computer or on any other computer to which it can connect over its network. During this presentation we will discuss what you will need to know to get started with .NET Remoting. We will talk about how .NET Remoting compares with DCOM, how to host remoted objects in a variety of applications, how to call remoted objects from a client application, how to control the life time of remoted objects, and how to secure remoting applications.

To learn more, visit
www.sys-con.com/webervicesedge2003

Advisory Board



Sean Rhody
Editor in Chief,
Web Services
Journal
Partner, CSC



Alan Williamson
Editor in Chief, Java
Developers Journal
Chief Technology
Officer, n-ary



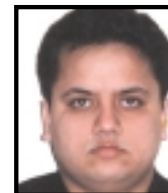
Derek Ferguson
Editor in Chief, .NET
Developers Journal
Chief Technology Evangelist,
Expand Beyond Corp.



Bob Familiar
.NET Architect,
Microsoft
New England



Thomas Robbins
Senior Technology Specialist,
Microsoft
New England



Hitesh Seth
Editor in Chief, XML Journal,
Chief Technology Officer, ikigo



J.P. Morgenthal
Chief Services Architect,
Software AG

Hotel Arrangements Are Easier Than Ever!

Special arrangements have been made with some of Boston's finest hotels, priced well below regular rates. Hotels are located within walking distance, if not connected to the Hynes Convention Center. To learn more about hotel savings, call ETI at (800) 829-2281 or (201) 444-0060 (direct) or fax reservations to (201) 444-0062. To make online reservations visit www.expotravel.com by Feb 24th, 2003.

SPECIAL RATES

Official Hotels	Address	Single	Double
Sheraton Boston Hotel	39 Dalton Street	\$159	\$159
Hilton Boston Back Bay	40 Dalton Street	\$149	\$149

RESERVATIONS

To make reservations call (800) 829-2281 or (201) 444-0060 (direct). Fax reservations to (201) 444-0062. Credit card information is required to guarantee reservations and expedite confirmation. Confirmations will be mailed directly from the hotel, time permitting. All changes and cancellations should be made directly through ETI.

REGISTRATION FORM

CONFERENCE: March 18 – 20, 2003 EXPO: March 19 – 20, 2003

John B. Hynes Veteran Memorial Convention Center • Boston, MA

THREE WAYS TO REGISTER FOR CONFERENCE

- 1) **On the Web:** Credit Cards or "Bill Me" Please make checks payable to SYS-CON Events
- 2) **By Fax:** Credit Cards or "Bill Me" 201-782-9651
- 3) **By Mail:** 135 Chestnut Ridge Road, Montvale, New Jersey 07645, Attention: Registration

Please note: Registrations are not confirmed until payment is received.

Please complete sections 1, 2, 3 and 4

1 YOUR INFORMATION (Please Print)

Mr.

Ms.

First Name _____ Last Name _____

Title _____

Company _____

Street _____

Mail Stop _____

City _____

State _____ Zip _____ Country _____

Phone _____

Fax _____ E-Mail _____

2 PAYMENT METHOD: (Payment in full due with registration)

Check or Money Order Enclosed (Registration confirmed upon receipt of payment)

Check # _____ Amount of Check \$ _____

Charge my Visa MasterCard American Express Discover

Name on card _____

Card # _____ Exp. Date _____

Signature _____

Billing Address (if different from mailing address) _____

3 PLEASE INDICATE YOUR CONFERENCE CHOICE:

Total Registration fee \$ _____

	Before 2/14/03	Before 3/14/03	On Site
<input type="checkbox"/> GP Gold Passport includes Edge Conference march 18-20, and Select one: Web Services Programming Using Java™ Technology and XML (Mar.18) Java™ Fast Path: Architect (Mar.19) Java™ Fast Path: Programmer (Mar. 20)	\$1,495.00	\$1,695.00	\$1,795.00
<input type="checkbox"/> 3D Three Day Conference (Does not include Sun Java™ Education)	\$1,295.00	\$1,495.00	\$1,695.00
<input type="checkbox"/> 2 Day Conference (Does not include Sun Java™ Education) (select any two days: <input type="checkbox"/> Tue, <input type="checkbox"/> Wed, <input type="checkbox"/> Thurs.)	\$1,195.00	\$1,295.00	\$1,395.00
<input type="checkbox"/> 1 Day Conference (Does not include Sun Java™ Education) (select any one day: <input type="checkbox"/> Tue, <input type="checkbox"/> Wed, <input type="checkbox"/> Thurs.)	\$495.00	\$695.00	\$895.00
<input type="checkbox"/> JU1 Sun Java™ University Class Select one: Select one: <input type="checkbox"/> Web Services Programming Using Java™ Technology and XML (Mar.18) <input type="checkbox"/> Java™ Fast Path: Architect (Mar.19) <input type="checkbox"/> Java™ Fast Path: Programmer (Mar. 20)	\$695.00	\$895.00	\$995.00
<input type="checkbox"/> JU2 Sun Java™ University Class Select two: Select one: <input type="checkbox"/> Web Services Programming Using Java™ Technology and XML (Mar.18) <input type="checkbox"/> Java™ Fast Path: Architect (Mar.19) <input type="checkbox"/> Java™ Fast Path: Programmer (Mar. 20)	\$1,295.00	\$1,495.00	\$1,595.00
<input type="checkbox"/> JU3 Sun Java™ University Class Select three: Select one: <input type="checkbox"/> Web Services Programming Using Java™ Technology and XML (Mar.18) <input type="checkbox"/> Java™ Fast Path: Architect (Mar.19) <input type="checkbox"/> Java™ Fast Path: Programmer (Mar. 20)	\$1,495.00	\$1,695.00	\$1,895.00

4

A. Your Job Title

- CTO, CIO, VP, Chief Architect
- Software Development Director/Manager/Evangelist
- IT Director/Manager
- Project Manager/Project Leader/Group Leader
- Software Architect/Systems Analyst
- Application Programmer/Evangelist
- Database Administrator/Programmer
- Software Developer/Systems Integrator/Consultant
- Web Programmers
- CEO/COO/President/Chairman/Owner/Partner
- VP/Director/Manager Marketing, Sales
- VP/Director/Manager of Product Development
- General Division Manager/Department Manager
- Other (please specify) _____

B. Business/Industry

- Computer Software
- Computer Hardware and Electronics
- Computer Networking & Telecommunications
- Internet/Web/E-commerce
- Consulting & Systems Integrator
- Financial Services
- Manufacturing
- Wholesale/Retail/Distribution
- Transportation
- Travel/Hospitality
- Government/Military/Aerospace
- Health Care/Medical
- Insurance/Legal
- Education
- Utilities
- Architecture/Construction/Real Estate
- Agriculture
- Nonprofit/Religious
- Other (please specify) _____

C. Total Number of Employees at Your Location and Entire Organization (check all that apply):

	Location	Company
10,000 or more	01 <input type="checkbox"/>	01 <input type="checkbox"/>
5,000 - 9,999	02 <input type="checkbox"/>	02 <input type="checkbox"/>
1,000 - 4,999	03 <input type="checkbox"/>	03 <input type="checkbox"/>
500 - 999	04 <input type="checkbox"/>	04 <input type="checkbox"/>
100-499	05 <input type="checkbox"/>	05 <input type="checkbox"/>
100 or less	06 <input type="checkbox"/>	06 <input type="checkbox"/>

D. Please indicate the value of communications and computer products and services that you recommend, buy, specify or approve over the course of one year:

- \$10 million or more
- \$1 million - \$9.9 million
- \$500,000 - \$999,999
- \$100,000 - \$499,999
- \$10,000 - \$99,999
- Less than \$10,000
- Don't know

E. What is your company's gross annual revenue?

- \$10 billion or more
- \$1 billion - \$9.9 billion
- \$100 million - \$999 million
- \$10 million - \$99.9 million
- \$1 million - \$9.9 million
- Less than \$1 million
- Don't know

F. Do you recommend, specify, evaluate, approve or purchase wireless products or services for your organization?

01 Yes 02 No

G. Which of the following products, services, and/or technologies do you currently approve, specify or recommend the purchase of?

- Application Servers
- Web Servers
- Server Side Hardware
- Client Side Hardware
- Wireless Device Hardware
- Databases
- Java IDEs
- Class Libraries
- Software Testing Tools
- Web Testing Tools
- Modeling Tools
- Team Development Tools
- Installation Tools
- Frameworks
- Database Access Tools / JDBC Devices
- Application Integration Tools
- Enterprise Development Tool Suites
- Messaging Tools
- Reporting Tools
- Debugging Tools
- Virtual Machines
- Wireless Development Tools
- XML Tools
- Web Services Development Toolkits
- Professional Training Services
- Other (Please Specify) _____

SYS-CON Events, Inc. and SYS-CON Media make no warranties regarding content, speakers or attendance. The opinions of speakers, exhibitors and sponsors do not reflect the opinion of SYS-CON Events and SYS-CON Media and no endorsement of speakers, exhibiting companies products, or sponsors is implied.

sessions and schedule are subject to change without prior notice.

No solicitation by anyone other than official exhibitors, sponsors or marketing partners is permitted. Such behavior is cause for expulsion without refund.



If you require special assistance covered under the Americans with Disabilities Act, please call 201-802-3058 by March 4, 2003.

CANCELLATIONS, SUBSTITUTIONS, REFUNDS
Fax written request to SYS-CON Registration 201-782-9651. Requests for refunds received prior to February 15, 2003 will be honored, less a 10% handling charge; requests received after February 15, 2003, and before March 1,

2003, will be honored less a 20% handling charge. No requests for refunds will be honored after March 1, 2003. Requests for substitutions must be made in writing prior to March 14, 2003. No one under 18 is permitted to attend. No warranties are made regarding the content of sessions or materials. Speakers,

Special Insert: Web Services Edge East Conference & Expo

Featuring FREE Tutorials, Training Sessions, Case Studies and Exposition



Meet with the industry experts, professionals, and managers building today's Web Services enterprises!

Conference
March 18-20, 2003

Exposition
March 19-20, 2003

Hynes Convention Center, Boston

Past Exhibitors & Participants



The World's Leading Java Resource!

JAVA DEVELOPERS' JOURNAL

Here's what you'll find in every issue of *JDJ*:

- Industry insights
- The latest software trends
- Technical expertise
- Career opportunities
- In-depth articles on Java technologies

Sign up **ONLINE** at www.javadevelopersjournal.com

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

Subscribe Today & **SAVE 30% Off** the annual cover price

ANNUAL COVER PRICE
~~\$71.88~~
YOU PAY
\$49.99
YOU SAVE
30% Off the annual cover price



WLDJ ADVERTISER INDEX

ADVERTISER	URL	PHONE	PAGE
Altaworks	www.altaworks.com	603-598-2582	25
BEA eWorld	www.bea-eworld.com	404-240-5506	17
BEA Systems	http://dev2dev.bea.com/useworkshop	800-817-4BEA	3
Component Source	www.componentsource.com/bea	888-850-9911	23
Dirig Software	www.dirig.com/illusion/weblogic	603-889-2777	51
FileNet	www.filenet.com	512-434-5935	21
Java Developer's Journal	www.javadevelopersjournal.com	888-303-5282	49
Neon Systems	www.neonsys.com/BEApaper	800-505-NEON	13
PANACYA	www.panacya.com	877-726-2292 x3344	27
Precise Software Solutions	www.precise.com/wldj	800-310-4777	52
Programmer's Paradise	www.programmersparadise.com/BEA	800-445-7899	9
Sitraka (now part of Quest Software)	www.sitraka.com/performance/wldj	800-663-4723	2
Sitraka (now part of Quest Software)	www.sitraka.com/jclass/wldj	800-663-4723	11
SYS-CON Media	http://developer.sys-con.com	201-802-3020	26
SYS-CON Publications	www.sys-con.com	888-303-5282	39, 49
Web Services Edge 2003	www.sys-con.com	201-802-3069	41
Web Services Journal	www.wsj2.com	888-303-5282	31
WebLogic Developer's Journal	www.weblogicdevelopersjournal.com	888-303-5282	37
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.

The world's leading i-technology publisher

HOME
SUBSCRIBE
ABOUT
NEWS
EMPLOYMENT
CONTACT
DIRECTIONS

Publications

- WebServices
- JAVA DEVELOPERS' JOURNAL
- wireless
- WebLogic
- XML JOURNAL
- COLDFUSION
- WebLogic
- JAVA DEVELOPERS' JOURNAL
- PowerBuilder JOURNAL

SUBSCRIBE NOW TO THE FINEST TECHNICAL JOURNALS IN THE INDUSTRY!

800 513-7111
www.sys-con.com

The world's leading i-technology publisher

News & Developments

Macromedia ColdFusion MX for WebLogic Released

(San Francisco) – Macromedia, Inc., has announced the availability of Macromedia ColdFusion MX for BEA WebLogic Server. This rapid server scripting environment for creating Internet applications brings the ease of use and productivity of ColdFusion to the standards-based Java technology architecture offered by BEA Systems.

Macromedia ColdFusion MX for BEA WebLogic Server is part of the ColdFusion MX for J2EE Application Servers product line. Designed to enable developers to construct rich Internet and Web services applications using high-level scripting, it can increase

development productivity and lower costs across IT organizations. A trial download of Macromedia ColdFusion MX for BEA WebLogic Server is available at www.macromedia.com/go/cfmx. www.macromedia.com



NetIQ Announces WebTrends for BEA Solutions

(San Jose, CA) – NetIQ Corporation, a leading provider of systems management, security management, and Web analytics solutions, has announced two WebTrends



solutions: WebTrends for BEA WebLogic Server and WebTrends for BEA WebLogic Portal, designed to enable BEA customers to take advantage of WebTrends Web analytics.

WebTrends for BEA WebLogic Server provides complete translation of BEA parameters, including all of the visitor activ-

ity data required for Web analysis. WebTrends for BEA WebLogic Portal is designed to analyze BEA behavior tracking data, including information on user interaction within the portal. www.netiq.com

Momentum in Europe Demonstrates Success of BEA Platform Vision

(San Jose, CA) – BEA Systems, Inc., the world's leading application infrastructure software company, has announced activities with 382 new customers in Europe and signed over 70 new partners during its third fiscal quarter, ending Oct. 31, 2002.

Across the globe, 78% of all deals over \$500,000 U.S. were multiproduct deals, demonstrating the success BEA has had in building on the functionality of BEA WebLogic Server by adding integration, portal, and application development tools.



BEA also established new partnerships in Europe with 30 additional systems integrators and 41 independent software vendors. www.bea.com

Sun Microsystems and BEA Increase Customer Choice, Reduce Costs

(San Jose and Santa Clara, CA) – BEA Systems, Inc., and Sun Microsystems have agreed to distribute an evaluation version of BEA WebLogic Server with the Solaris 9 Operating Environment System Administrator's Kit. The agreement is intended to maximize customer choice for J2EE application servers for enterprise and Web services, and provide customers with a trial license of BEA WebLogic Server 7.0. Beginning with the January release of Solaris 9, BEA offered a six-month trial license with each BEA WebLogic Server 7.0 distributed via the Solaris 9 System Administrator's Kit.

Solaris 9 is a key component of Sun ONE, an open, integratable product portfolio designed to enable the development and delivery of Java Web services. The combined products provide significant advantages to customers,



including speed of deployment, solution flexibility, interoperability, profitability, and 24/7 operations. <http://sun.com>, www.bea.com

BEA Systems and ComponentSource Form Strategic Alliance

(San Jose, CA and Atlanta, GA) – BEA Systems, Inc., and ComponentSource, the world's largest marketplace and community for reusable software components and tools available. Both companies will work with the global ISV community to create new reusable components,



including Web services and tools for the BEA WebLogic Enterprise Platform and other Java platforms.

Now available are two new ComponentSource-hosted marketplaces. The BEA WebLogic Galleries are community-based marketplaces populated with tools and reusable components (<http://dev2dev.bea.com/code/components.jsp>).

ComponentSource has also launched a dedicated BEA WebLogic Store on its own public marketplace, www.componentsource.com/bea, that will target its developer consumer

base in over 100 countries. www.componentsource.com

FileNET and BEA Systems Form Strategic Technology Alliance

(Costa Mesa, CA) – FileNET Corporation, provider of enterprise content management (ECM) solutions, has formed a technology alliance with BEA Systems, Inc., to develop and market integrated content and process management solutions designed for use with BEA's WebLogic Platform 7.0, which features a J2EE application server combined with development, portal, and integration frameworks.



FileNET will deploy and use BEA WebLogic Server as its premier development, test, and demonstration platform. FileNET's Web Content Management capability currently integrates with BEA WebLogic Server. Under the new agreement, FileNET will expand its relationship with BEA to integrate the entire FileNET ECM suite with the BEA WebLogic Platform. The combined solution will be designed to automate the delivery of enterprise content to applications built on BEA WebLogic Platform, and to provide a unified framework for creating, managing, and approving content accessed via enterprise portals.

www.FileNET.com



Dirig Software

www.dirig.com/illusion/weblogic

Precise Software Solutions

www.precise.com/wldj