SYS-CON

NOW INCORPORATING XML JOURNAL

WebServices JOURNAL

.NET J2EE XML

October 2004 Volume 4 Issue 10

Putting the EDA in SOA

How upcoming Web services message standards will merge event-driven architectures with SOAs

pg20

WSIF & JSR-208

Flexible binding frameworks for today and tomorrow
pg.10

Automating MISMO Processes

Web services bring new advantages
pg.26

Web Service Local Reference

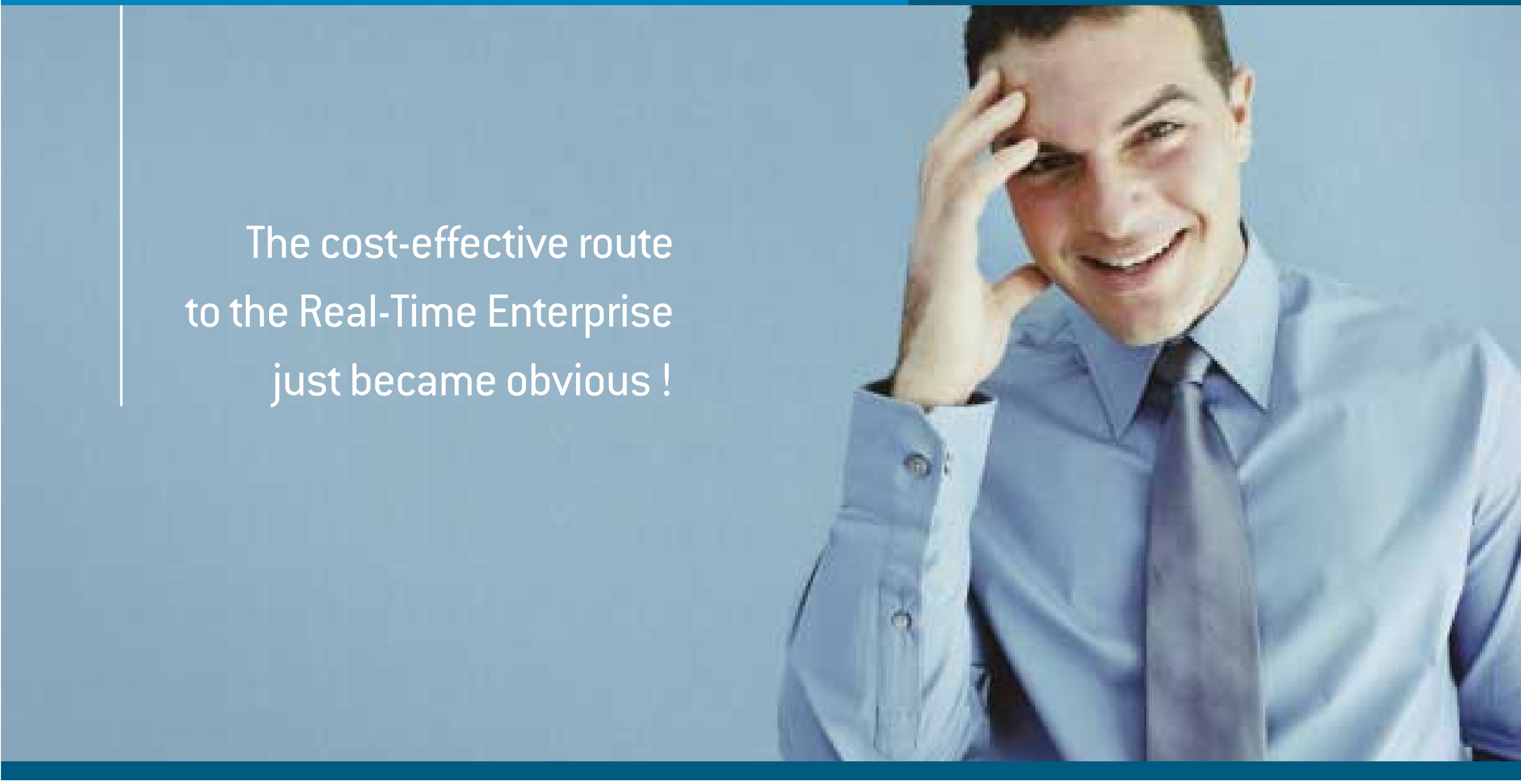
Solving significant concerns
pg.34

What Color Is Your Schema?

Add your own information
pg.52

RETAILERS PLEASE DISPLAY
UNTIL NOVEMBER 30, 2004

$9.99US $9.99CAN


The cost-effective route
to the Real-Time Enterprise
just became obvious !

How to Make Money with Web Services

By Bob Brauer
bob.brauer@strikeiron.com

Search
Tools/Services
Access
Monitor
STRIKE IRON
Trusted/Reliable
Web Services
Business Network
Subscribe
Analyze
Understand

A new way to work with Web services - build them and they will come?

The benefits of Web services are well understood – lower integration costs, maximum reusability, faster deployment, more automation, easier to work with new partners, and so on. You can build them but that does not necessarily mean that people will use them or that you can make money on them. Why? And more importantly, how?

What drives Web services utilization and return on investment?

Here are some key points to consider:

- Accessibility and ease-of-use
- Billing and accounting
- Subscriptions and trials
- Availability, reliability, and security
- Promotions and pricing

What tools and services do you need to commercialize your Web services?

- Tools to improve understanding and use.
- Commerce capabilities to manage subscriptions, accounting, billing, payments, account management, etc.
- Ability to manage free trials and convert trial users to subscribers.
- Service levels that ensure availability, reliability, and security.
- Knowledge about acceptable pricing structures based on value of the data, process, accessibility, and performance.
- A way to deliver and promote them to the appropriate target audience.

Creating a new revenue channel

Your Web services provide value and you need to be reimbursed for that value. However, before that can happen, you need a distribution channel with the infrastructure to publish and sell your Web services. This channel must take care of issues such as delivery, account set up, billing and collections, marketing, and customer support, to free you of having to make that investment.

The answer is Premium Web Services

Premium Web Services provided by StrikeIron and available through the StrikeIron Web Services Business Network™ (WSBizNet™) are the answer.

Whether you are a Web service provider wanting to commercialize your Web services or a company wanting to find a way to manage your Web services without the infrastructure investment, StrikeIron will do the work for you, and you make money in the process.

StrikeIron Premium Web Services are subscription-based Web services that deliver "live" data and functionality over the Internet via XML using a dynamic SOAP interface. Premium Web Services are supported by the commercial services and easier-to-use tools of the StrikeIron WSBizNet to remove barriers to purchase.

Premium Web Services benefits

- An alternate sales channel to reach more customers for minimal cost.
- Web services hosting and delivery services in a secure environment.
- Pricing, billing, collections, and support.
- Reporting for easier maintenance and tracking.

The StrikeIron Web Services Business Network

The StrikeIron WSBizNet is a subscription-based online network of services and tools for easier and faster utilization of Web services by business users and developers.

The WSBizNet allows anyone, with minimal experience, to quickly find, understand, and utilize Web services for accessing information, integrating with existing applications, and assembling new applications.

StrikeIron WSBizNet benefits

- Service levels to meet your business requirements.
- Integrated online StrikeIron Web Services Analyzer for faster understanding of Web services.
- Integrated online Knowledge Base makes it easier to understand and work with your Web services.
- Prime positioning in the WSBizNet.
- Increased visibility in one of the largest Web services business directories.

Delivering Web Services to the World!

For Free Trials and more about how to make money with the StrikeIron Web Services Business Network and Premium Web Services, please visit our Web site at www.strikeiron.com.

StrikeIron, Inc.
+1.919.405.7010
sales@strikeiron.com
www.strikeiron.com

OCTOBER 2004 Volume 4 Issue 9

WebServices JOURNAL
.NET J2EE XML

XML JOURNAL

Inside WSJ

Open Up!

Portals–Get what
you want easily

company, product and service names maybe trademarks or service marks of others.

Ask the Experts

Prolifics is 1 of only 3 WebSphere Service Providers retained by IBM. Open up with portals! With 26 years of industry expertise, Prolifics' certified WebSphere portal experts deliver a personalized, secure, unified view to your enterprise assets.

Our portal solutions build agility into your business — enabling you to reach new markets, achieve customer satisfaction beyond expectation, and ease collaboration within your enterprise. Our teams of specialists work together with your business experts from gathering the business requirements, translating them to a technical specification, to delivering the final production application.

Portal Executive Assessment

Prolifics experts help you articulate the value of portals to your business sponsors and perform a quick business review identifying portal benefits as it applies to your organization.

Portal Workshops and Proof of Concepts

Prolifics demonstrates valuable portal features and translates the breadth of your portal objectives into a step-by-step roadmap to meet your current and future needs.

Portal Jump Start

Prolifics' consultants work with you from the start to install, configure and jump start your team on using WebSphere Portal Software.

Portal Development and Mentoring

Prolifics' innovative approach to building portals drives down your overall cost and transforms the way you conduct business. We mentor and arm your team with the skills to manage, maintain and extend the portal environment.

Prolifics

Winner of 2003 IBM BUSINESS PARTNER LEADERSHIP AWARD

116 John Street, New York, New York 10038
800.675.5419
solutions4websphere@prolifics.com
www.prolifics.com

Prolifics®

IBM Business Partner

WebServices .NET J2EE XML JOURNAL

INTERNATIONAL ADVISORY BOARD
Andrew Astor, David Chappell, Graham Glass, Tyson Hartman,
Paul Lipton, Anne Thomas Manes, Norbert Mikula,
George Paolini, James Phillips, Simon Phipps,
Mark Potts, Martin Wolf

TECHNICAL ADVISORY BOARD
Bernhard Borges, JP Morgenthal, Andy Roberts,
Michael A. Sick, Simeon Simeonov

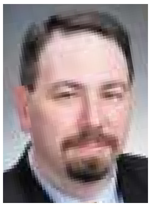
EDITORIAL
EDITOR-IN-CHIEF
Sean Rhody sean@sys-con.com
XML EDITOR
Hitesh Seth
INDUSTRY EDITOR
Norbert Mikula norbert@sys-con.com
PRODUCT REVIEW EDITOR
Brian Barbash bbarbash@sys-con.com
.NET EDITOR
Dave Rader davidr@fusiontech.com
SECURITY EDITOR
Michael Mosher wsjsecurity@sys-con.com
RESEARCH EDITOR
Bahadir Karuv, Ph.D Bahadir@sys-con.com
TECHNICAL EDITORS
Andrew Astor aastor@webmethods.com
David Chappell chappell@sonicsoftware.com
Anne Thomas Manes anne@manes.net
Mike Sick msick@sys-con.com
EXECUTIVE EDITOR
Gail Schultz gail@sys-con.com
EDITOR
Nancy Valentine nancy@sys-con.com
ASSOCIATE EDITOR
Jamie Matusow jamie@sys-con.com
ONLINE EDITOR
Martin Wezdecki martin@sys-con.com

PRODUCTION
PRODUCTION CONSULTANT
Jim Morgan jim@sys-con.com
LEAD DESIGNER
Richard Silverberg richards@sys-con.com
ART DIRECTOR
Alex Botero alex@sys-con.com
ASSOCIATE ART DIRECTORS
Louis F. Cuffari louis@sys-con.com
Tami Beatty tami @sys-con.com
Andrea Boden andrea@sys-con.com

CONTRIBUTORS TO THIS ISSUE
Mohamad Afshar, Brian Barbash, Bogdan Blaszczak,
J. Scott Bushey, Mike Champion, John Deeb, Dave Shaffer,
Andrew Bradfield, Marc Breissing, Tom Glover, Tyson Hartman,
Pankaj Kothari, David Linthicum, Eric Norlin, Mike Mosher, Darren
Platt, Venkat Ragunathan, Sean Rhody, Rick Robinson, Jonathan
Sapir, Hal Steger, Kirstan Vandersluis,
Deepak Vohra, Ajay Vohra

EDITORIAL OFFICES
SYS-CON MEDIA
135 CHESTNUT RIDGE ROAD, MONTVALE, NJ 07645
TELEPHONE: 201 802-3000 FAX: 201 782-9637
WEB SERVICES JOURNAL (ISSN# 1535-6906)
Is published monthly (12 times a year)
By SYS-CON Publications, Inc.
Periodicals postage pending
Montvale, NJ 07645 and additional mailing offices
POSTMASTER: Send address changes to:
WEB SERVICES JOURNAL, SYS-CON Publications, Inc.
135 Chestnut Ridge Road, Montvale, NJ 07645

Web Services, Part II: The Search for the Optional Standards



WRITTEN BY
SEAN RHODY

We've been covering Web services technologies for quite some time now, almost three years. In that time – I think it amounts to two eternities in Internet time – we've seen all sorts of interesting things occur. Cooperation, coopetition, even the creation of a group whose sole purpose is to make sure that the standards really are standard. (Quis custodiet ipsos custodies? That's Latin for "who's watching the WS-I?) But one of the more interesting developments, one that's really not been discussed greatly, is the number of "optional standards."

Yes, I know, that sounds like an oxymoron, like jumbo shrimp. But the reality of Web services is that it's made up of so many components that some just cry out to be used, while others are left out in the cold at many corporations.

The first candidate for the title of "Left Out 2004" would, of course, be the perennial favorite and Web services demon UDDI. Let's face it, the idea of a single public UDDI registry as an analog to DNS is deader than a doornail. And for good reason. This relegates whole sections of Web services books to the fireplace. Go to chapter 2, rip out all the UDDI pages, and use them for kindling. You'll sleep better at night – at least that night – because you'll be nice and warm. Now realistically, UDDI might be useful in certain circumstances. Certain limited circumstances. However, not enough that organizations who use other standards but ignore UDDI need to worry about saying they do "Web services". You do.

Truthfully, the standards bodies haven't helped matters any either. There are too many standards, at too granular a level, and too much competition. Need an example of how ***not*** to build an external interface? Just look at the standards on transaction and coordination. We need one, single standard, folks, not WS – Transaction and WS-Coordination and of course WS-I'll invent my own protocol and call it a standard. Half of these standards are "optional" or "future release" because no one can tell which one is going to gain acceptance. And frankly, I don't want to have to worry that the vendor that implements WS-Coordination also implements WS-Orchestration. They're all facets of the same problem – federating Web services to provide composite services, with transactional integrity. Note to OASIS and the W3C: "More standards – Bad. Consolidation – Good."

Adoption of other standards is sometimes a matter of scale. A single Web services implementation rarely requires full-on management, and if it's internal, it may not require extensive security either. Raise your hand if you want to group security under management anyway. See that? It's consolidation at work. Already we've condensed a whole group of standards.

But when an organization is up to its eyebrows in Web services, can't figure out where they are, or who's using them, or how to ensure quality of service, that's when these "optional standards" become critical. Critical mass requires a second set of Web services technologies – ones that ensure those simple Web services built on the "standard standards" work together in real world, complex situations.

This month, we're focusing on standards themselves, and examining two of the "optional standards" (UDDI and BPEL4WS) in some detail. UDDI, as you can tell, is one of my favorites. BPEL4WS, on the other hand, is something I think will be very useful, if we can get the vendors who proposed it to actually implement it in their products, instead of just talking about it in a future release. But in the meantime, we'll just have to build our Web services with the "Sort of Standards," and the "Proprietary Extensions." Enjoy.

About the Author

Sean Rhody is the editor-in-chief of *Web Services Journal.* He is a respected industry expert and a consultant with a leading consulting services company.

sean@sys-con.com

WebServices .NET J2EE XML JOURNAL

PRESIDENT AND CEO
Fuat Kircaali fuat@sys-con.com
VP, BUSINESS DEVELOPMENT
Grisha Davida grisha@sys-con.com
GROUP PUBLISHER
Jeremy Geelan jeremy@sys-con.com

ADVERTISING
SENIOR VP, SALES & MARKETING
Carmen Gonzalez carmen@sys-con.com
VP, SALES & MARKETING
Miles Silverman miles@sys-con.com
ADVERTISING DIRECTOR
Robyn Forma robyn@sys-con.com
ADVERTISING MANAGER
Megan Mussa megan@sys-con.com
ASSOCIATE SALES MANAGERS
Kristin Kuhnle kristin@sys-con.com
Beth Jones beth@sys-con.com
Dorothy Gil dorothy@sys-con.com

SYS-CON EVENTS
PRESIDENT, SYS-CON EVENTS
Grisha Davida grisha@sys-con.com
NATIONAL SALES MANAGER
Jim Hanchrow jimh@sys-con.com

CUSTOMER RELATIONS/JDJ STORE
CIRCULATION SERVICE COORDINATORS
Edna Earle Russell edna@sys-con.com
Linda Lipton linda@sys-con.com

SYS-CON.COM
VP, INFORMATION SYSTEMS
Robert Diamond robert@sys-con.com
WEB DESIGNERS
Stephen Kilmurray stephen@sys-con.com
Matthew Pollotta matthew@sys-con.com

ACCOUNTING
FINANCIAL ANALYST
Joan LaRose joan@sys-con.com
ACCOUNTS PAYABLE
Betty White betty@sys-con.com
ACCOUNTS RECEIVABLE
Shannon Rymsza shannon@sys-con.com

SUBSCRIPTIONS
SUBSCRIBE@SYS-CON.COM
1-201-802-3012
1-888-303-5282
For subscriptions and requests for bulk orders,
please send your letters to Subscription Department
Cover Price: $6.99/issue
Domestic: $69.99/yr (12 issues)
Canada/Mexico: $89.99/yr
All other countries: $99.99/yr
(U.S. Banks or Money Orders)

Worldwide Newsstand Distribution
Curtis Circulation Company, New Milford, NJ
Newsstand Distribution Consultant:
Brian J. Gregory / Gregory Associates / W.R.D.S.
732 607-9941 - BJGAssociates@cs.com

For list rental information:
Kevin Collopy: 845 731-2684,
kevin.collopy@edithroman.com;
Frank Cipolla: 845 731-3832,
frank.cipolla@epostdirect.com

SYS-CON Publications, Inc., reserves the right to revise,
republish and authorize its readers to use the articles
submitted for publication.

SYS-CON MEDIA

Web Services, Are We There Yet?

WRITTEN BY
TOM GLOVER

We're all familiar with the Gartner "hype cycle," we've experienced the roller coaster ride it describes as we've worked to move a new technology towards becoming relevant. Web services has been evolving for approximately 36 months, and many already claim victory. However, while there's reason for optimism, much must be done before Web services become meaningfully successful.

Web services' value rests upon the simple but powerful concept that all Web services offer interfaces defined using a standard set of XML-based specifications. In the past developers integrating software needed to consider issues such as "what hardware are the services deployed on," "what middleware do they run on," "what languages are they written in," and even "what tools were used to develop them." Web services XML-based interfaces make these issues much less relevant; developers integrating Web services should be able to do their jobs more rapidly and with more confidence due to the use of common Web services standards to define the integration points for the services.

We've made great progress in the last 36 months. The following specifications have been developed and broadly adopted by the market (and so may be called standards), are broadly supported (so they may be used by application developers), and have proven their value through their use in application development:

- ***XML Schema:*** Helps define the data passed between Web services and other software components that use them. (www.w3.org/XML/Schema)
- ***Simple Object Access Protocol (SOAP):*** Defines how messages sent to and received from Web services are formed. (www.w3.org/TR/soap/)
- Web Service Description Language (WSDL): Defines how Web services interfaces are defined (www.w3.org/TR/wsdl/)
- ***Universal Description, Discovery, and Integration (UDDI):*** Provides information concerning registries that discover Web services, (www.oasis-open.org/committees/uddi-spec/doc/tcspecs.htm)
- ***Web Services Security (WSS):*** Defines mechanisms for securing Web services from various security threats, many of which cannot be addressed via HTTPS

Several issues remain:

We need to complete the standard set. Much work is under way; however, we can't yet declare that we have all the needed specifications or that they've matured into standards.

- ***The specifications must be made usable:*** Web services interfaces that may be described using a single specification are few enough to exclude them from any practical discussion. Services interfaces are generally described using sets of specifications. We must identify broadly useful standards sets and refine them to clearly specify the characteristics of "conformant" interfaces. This work is underway in the Web Services Interoperability Organization (WS-I). WS-I is nearly finished profiling the existing Web services standards set, and will include other standards that emerge in future profiles.
- ***We need software that supports use of the specifications:*** The publication of specifications, including WS-I Profiles, isn't sufficient to realize Web services' potential. The delivery of supporting tools and middleware must follow. While many vendors support using some of these standards we have not yet reached a point where it may be assumed. This means that we're not yet able to implement e a Web service on an arbitrary platform with confidence that it may be integrated into an application. In addition, vendor implementation of emerging Web services specifications is unpredictable. We need to find a way to have vendors move forward in a more uniform manner, addressing a common set of specifications in a common order.
- ***The specifications must be put to work:*** Web services become truly relevant as the pool of useful Web services grows, and as these services are integrated into a growing set of business solutions.

Much must be done if Web services are to deliver their "everybody wins" potential. And there's more. Service-oriented architecture (SOA), one of the buzzwords moving along the hype cycle this year, is an application architecture for integrating services together to form business solutions. SOA isn't particularly meaningful if we don't have a set of services available to integrate. Web services is the leading candidate for a serviceable (pardon the pun) technology foundation.

The bottom line is that we're not there yet, and more rides on our succeeding than ever before. I hope you'll all join in!

About the Author

Tom Glover is the president and chairman, Web Services Interoperability Organization, and is the senior program manager of Web services standards for IBM Software Group. Prior to working with WS-I, Tom served as general program manager of UDDI.org and was responsible for guiding the organization as it developed the Universal Description, Discovery and Integration Specification. He was also managing director of the UDDI Business Registry Operators' Council.

Is your SOA really

Building, testing and maintaining SOA solutions requires a flexible and collaborative approach to diagnostics. Finger pointing and confusion occur when all parties do not have a complete and common understanding of a problem. Mindreef® SOAPscope® 4.0 connects developers, testers, support personnel, and operations by combining the testing and diagnostic tools for each discipline and making it easy to share complete problem data. SOAPscope is the only end-to-end diagnostic system for Web services.

Developers • Testers • Operations • Support

NEW! SOAPscope 4.0

Test - Agile testing approach for both developers and testers allows what-if scenarios without coding or scripting. Includes the industry's most complete interoperability suite and the flexibility to really test the edge cases.

Capture - Easily capture and assemble complete problem data into SOAPscope's workspace. Capture SOAP messages off the wire, test cases, or re-produced problems. Capture a snapshot in time of a WSDL including all related resources.

Share - Exported workspaces support sharing among team members preserving complete problem data including messages, WSDLs, and notes. Sharing workspaces is a great way to support web service users.

Solve - Diagnostic suite of analysis, comparison and visibility tools simplifies the understanding of complex XML, yet are powerful enough for the industry's most protocol-savvy experts.

Try SOAPscope **FREE** at www.Mindreef.com

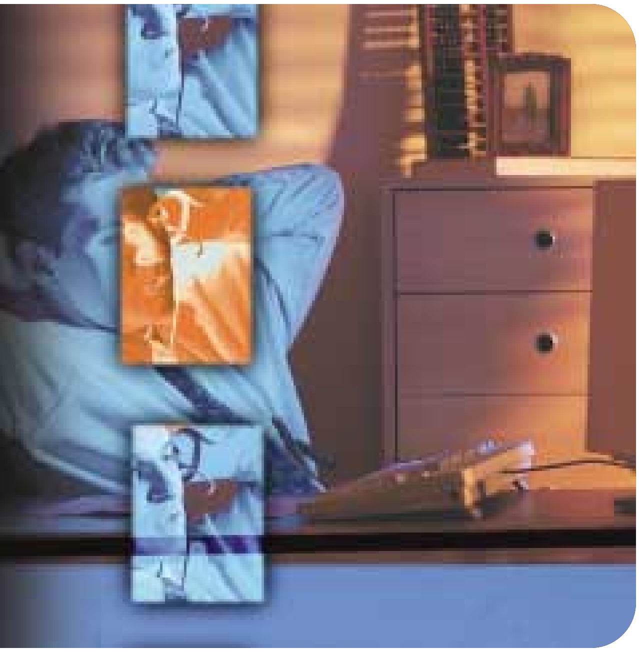
"SOAPscope has been invaluable in testing and diagnosing our enterprise scale Web Services. The new Workspace feature in SOAPscope 4.0 is an excellent way to support customers and collaborate as a team. A must have for every WS-Developer"

Simon Fell
Senior Member of
Technical Staff at
salesforce.com and
author of PocketSOAP

Mindreef, Inc.
www.Mindreef.com

WSIF & JSR-208

Flexible binding frameworks for today and tomorrow



There's a common misconception that Business Process Execution Language for Web Services (BPEL) is useful only if all of your systems are Web services. This article describes how Web Services Invocation Framework (WSIF) enables BPEL to orchestrate nearly any legacy system as if it were a Web service – without having to explicitly wrap or publish it as one. It also highlights how JSR-208 will standardize this capability in the not-too-distant future.

WRITTEN BY
DAVE SHAFFER,
JOHN DEEB,
&
MOHAMAD AFSHAR

Introduction

As Web services begin to take hold as an enterprise integration strategy, BPEL has rapidly become the undisputed standard for business process integration. BPEL provides a standard, portable language for orchestrating services into end-to-end business processes and builds upon a decade of progress in the areas of business process management, workflow, and integration technologies. Built from the ground up around XML and Web services, BPEL is supported on both the Microsoft .NET and Java platforms.

Many existing systems, however, do not currently expose Web services interfaces, and systems architects may not want to create Web services wrappers for all their legacy systems, either for performance reasons or to simplify implementations. This leaves some enterprises wondering if they can effectively use BPEL to compose business processes before they've adopted Web services (strictly defined as a SOAP envelope, a WSDL interface, and an XML data model, and usually HTTP as a protocol) throughout their IT infrastructure.

While a quick read of the BPEL specification may leave the impression that all integration components must be exposed as pure Web services, this is not true. BPEL requires you to provide a WSDL interface for services that will be incorporated into a business process – it doesn't constrain the protocol used to interface to a system. In other words, BPEL just requires that the services look like Web services.

WSIF to the Rescue

Web Services Invocation Framework (WSIF) is a flexible binding framework from Apache that lets you access back-end systems via a variety of transports, protocols, and even data models, while providing a WSDL interface to the client–in this case, the BPEL process that will invoke the service. WSIF enables a best-of-both-worlds strategy where back-end systems can be accessed through native protocols and data formats, but can still be incorporated into 100% standard BPEL processes just like full-fledged Web services. Developers can still expose or wrap legacy systems as Web services, however, and WSIF is not needed when all the systems to be orchestrated have Web services interfaces. But the capability to use the BPEL standard to orchestrate services that are not Web services lets you realize the full power of the BPEL standard – even within existing heterogeneous IT environments.

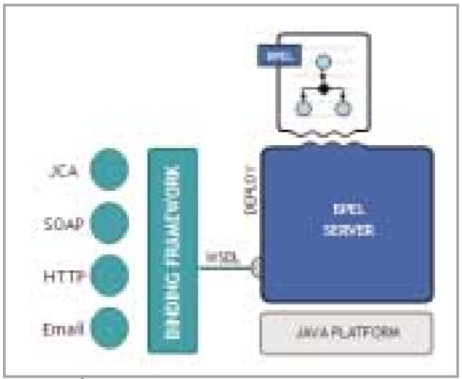


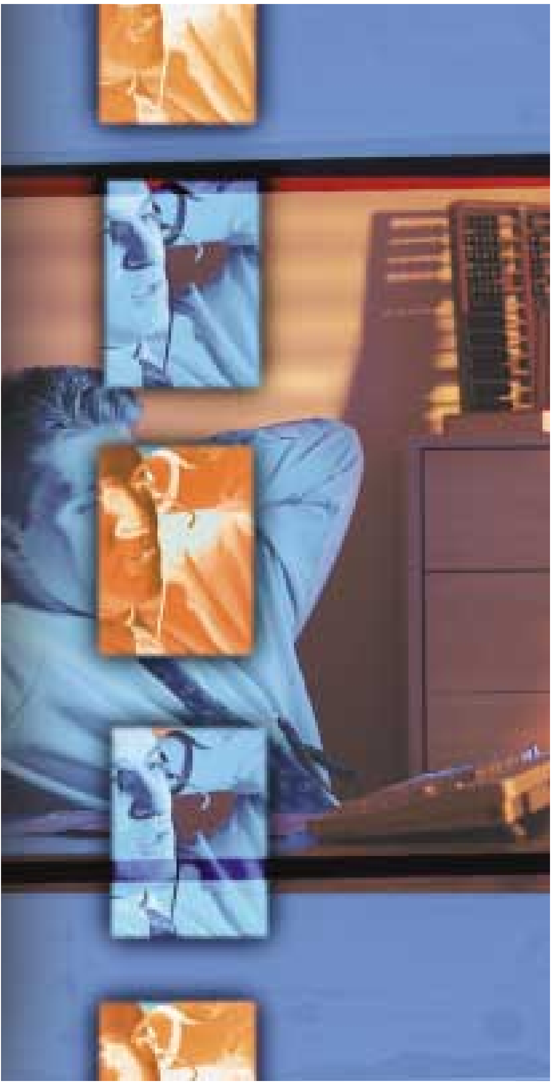
FIGURE 1 WSIF and the underlying J2EE platform

This approach is not theoretical as BPEL

software from both IBM and Oracle supports it. The overall architecture is shown in Figure 1.

Let's look at a few use cases to see how WSIF is being employed in the real world:

- A Global 500 consumer packaged goods supplier wanted to be able to orchestrate requests to a large number of SAP systems using its preferred Java Connector Architecture (JCA) vendor, for which it had a source license. A WSIF binding for JCA let the supplier orchestrate SAP Business Programming Application Interfaces (BAPIs) directly from its BPEL processes.
- A prescription drug supplier had existing HTTP GET interfaces in front of mainframe batch processes and wanted to be able to include those in its BPEL flows. A custom WSIF binding to support HTTP GET requests was built. Now, the WSDL interface to this HTTP "Web service" means that standard BPEL processes can interact with the mainframe applications without requiring support for a new interface.
- A media research vendor built and supported EJB interfaces for hundreds of back-end services but did not want to wrap them in SOAP, for performance reasons and speed of implementation. Several options were considered, including using the WSIF Java binding to communicate with Java objects as if they were Web services, as well as embedding Java code directly in the BPEL process.



When Is the WSIF Binding Approach Appropriate?

These companies are considering providing true Web service interfaces to their back-end systems – and some still plan to do so in the future to take advantage of a standard integration interface and the management and security features it provides. But BPEL plus WSIF offers a pragmatic approach that preserves the value of a business–process integration standard while being minimally invasive to the existing IT environment.

For any given project, a few key questions can help determine whether the best approach is a true Web service or a WSIF binding to a back-end system. These include:

- Will a Web services/SOAP interface let other clients who may not support WSIF access the service?
- Is a Web services management tool being used?
- What are the other management, administration, and performance criteria for the production environment?

Regarding Web services management, a SOAP Web service will be under the control of the management platform, whereas a direct WSIF binding to a back-end system may bypass the management facilities. Even for companies that have made a broad commitment to Web services, a WSIF binding to a back-end system can be a short-term solution. In any case it's a good option to have.

A Detailed Example

Some tools support WSIF bindings, including JMS, JCA, Java, and others that developers can use out of the box. Let's look at how an SAP BAPI that provides inventory information can be integrated into a BPEL process using a JCA binding. The business process in this case is an availability-to-promise flow where a BPEL process is used to automate the response to a customer's product availability request (see Figure 2). The BPEL flow checks inventory in an SAP system, and interacts with other systems that may or may not be Web services before returning a response.

To integrate the SAP BAPI into this BPEL process, a developer would first use JCA to select and expose the BAPI. Most JCA providers provide a GUI interface by which the back-end system functions can be browsed and selected graphically.

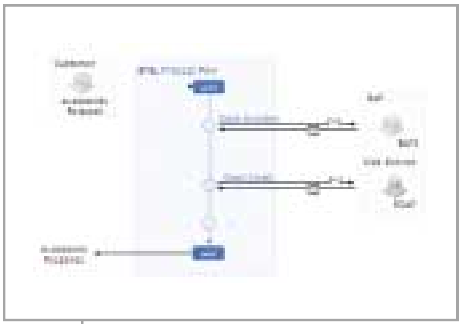


FIGURE 2 Order-to-promise BPEL flow

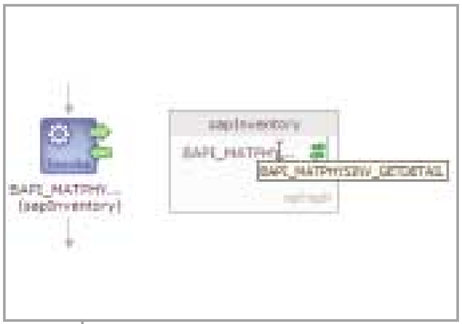


FIGURE 3 SAP inventory service partnerLink

Next, a WSDL needs to be created to define the JCA method as a WSDL operation and to specify the appropriate WSIF binding for this operation. This WSDL can be generated automatically at design time; an example of the generated operation and binding is shown in Listing 1. Note that the operation looks like any other WSDL operation and the binding specifies a built-in JCABinding (see Listing 1).

Once a WSDL exists, a BPEL process can call this service as if it were any regular Web service. For example, the BPEL process that invokes this list would set up a partnerLink for it, just like any other Web service:

```
<partnerLink name="sapInventory"

partnerLinkType="ic1:inventoryServiceLT"

partnerRole="inventoryServiceProvider"/>
```

And then invoke the service:

```
<invoke partnerLink="sapInventory"

portType="ic1:sapInventoryService"
    operation="BAPI_MATPHYSINV_GETDETAIL"
        inputVariable="request"
        outputVariable="response"/>
```

Figure 3 shows the visuals for the SAP inventory service partnerLink and the iBPEL activity to invoke the SAP service/BAPI. Note that these visuals, like the use of this service in both BPEL and the design tool, are exactly the same as for SOAP Web services.

In addition to built-in WSIF bindings, it's also important to consider how much effort is required to create new WSIF bindings. For example, a single consulting engineer wrote the custom HTTP binding described in the customer use case of a prescription drug supplier in two days. Any developer who wants to use it needs only to create a WSDL describing the interface to each HTTP "service." With a few more days of work, a tool could also be created to help generate this WSDL. New bindings are registered using a simple API provided by WSIF.

> "As the standardization process continues, the use of flexible binding frameworks such as WSIF is likely to grow"

Next Up: Standards for Binding (JSR-208)

WSIF is an open source framework available from Apache and is supported by several Java/J2EE BPEL vendors. However, because bindings are such a key component of business process integration, the process to create a standardized binding framework has already begun. Java Business Integration (JSR-208) proposes to standardize the ways in which J2EE assets, and therefore BPEL servers implemented on J2EE platforms, can bind to back-end systems. JSR-208 will define a standard business protocol representation that is then mapped to a specific communication protocol by a binding. In addition, JSR-208 will standardize the packaging and deployment of bindings. In this way, JSR-208 will let enterprises implement 100% standard processes, including both the process orchestration language and back-end connectivity.

Summary

As the standardization process continues, the use of flexible binding frameworks such as WSIF is likely to grow. However, WSIF already provides a de-facto standard to bind to back-end systems using native protocols and data formats, while exposing a Web service interface to a business-process client. In addition, the fact that WSIF is open source technology and is supported by the major J2EE-based BPEL vendors has already contributed to its broad adoption. By combining the WSIF binding framework with the BPEL Web service orchestration language, a 100% standard approach to business-process integration will become possible.

References

For more information

- *WSIF:* http://ws.apache.org/wsif/
- *JSR-208:* www.jcp.org/en/jsr/detail?id=208
- *BPEL engines and the code examples in this article:* http://otn.oracle.com/bpel

Listing 1

```
  <portType name="InventoryService">
    <operation name="BAPI_MATPHYS-
INV_GETDETAIL">
      <input
message="tns:checkInventoryInput"/>
      <output
message="tns:checkInventoryOutput"/>
    </operation>
  </portType>

  <binding name="JCABinding"
type="tns:ic1">
    <jca:binding/>
    <format:typeMapping
encoding="Java" style="Java"/>
    <operation name=" BAPI_MATPHYS-
INV_GETDETAIL ">
    <jca:operation
methodName="BAPI_MATPHYSINV_GETDE-
TAIL"/>
      <input/>
      <output/>
    </operation>
  </binding>
```

About the Authors

Dave Shaffer has been helping customers use the Oracle BPEL Process Manager since 2001, managing implementation projects, providing technical training, and ensuring successful implementations. Prior to joining Oracle, Shaffer was a principal consultant at Collaxa, a managing director at Eleven Acceleration, and manager of a professional services group at Apple Computer.

■■■ david.shaffer@oracle.com.

John Deeb is a member of the product management team for Oracle Application Server 10*g*. He focuses on enterprise integration, business process management, and business activity monitoring (BAM).

■■■ john.deeb@oracle.com.

Mohamad Afshar, PhD, is director of product management for Oracle Application Server 10*g*. His focus includes middleware architecture, Web services vision, and evangelism. Mohamad was most recently the CEO of a startup that built a Java/XML-based infrastructure for powering automated mobile speech applications. His new startup is one of the leaders in BAM. He holds a PhD in parallel database systems from the University of Cambridge, has held numerous speaking engagements, and has been published in both business-oriented and academic journals.

■■■ mohamad.afshar@oracle.com.

SOA and the Art of Riding the Enterprise Service Bus, Part II

The need for the ESB and other infrastructure technologies

In the first article in this series (Vol. 4, issue 9), I described some of the middleware requirements necessary to support a full-function service-oriented architecture (SOA), and introduced the role of the enterprise service bus (ESB) in meeting those requirements.

WRITTEN BY
RICK ROBINSON

In this article, I will go on to consider some specific situations in which ESB capabilities could be deployed. I will also describe how ESB capabilities relate to other architectural components in an SOA, and briefly discuss suitable technology options for implementing them.

The Capabilities of the Enterprise Service Bus

Now I'll summarize and categorize some of the ESB capabilities driven by the requirements we discussed. Whilst some of the capabilities are quite basic, some (such as autonomic or intelligent capabilities) represent significant steps towards an on-demand operating environment (see References). By analyzing which capabilities are applicable to specific SOA scenarios, we can refine our understanding of which technologies would be suitable candidates for implementing the ESB. In particular, we will go on to consider which of these capabilities are the minimum necessary to constitute a useful ESB.

Communication Capabilities

The ESB should support communication of service interactions through a variety of protocols. It may do this by providing a communication infrastructure, or by leveraging or integrating with existing infrastructures, such as HTTP or other middleware technologies, such as WebSphere MQ. The ESB should be able to route service interactions through a variety of protocols, and transform from one protocol to another where necessary.

Service Interaction Capabilities

The ESB should support the concepts of SOA, particularly the use of interfaces and policies to declare service operations and quality of service characteristics, respectively. The ESB should also support service-messaging models consistent with those interfaces, and capable of transmitting the required interaction context, such as security, transaction, or message correlation information. The ESB should also support the need to publish, discover, locate, and bind to services. Increasingly these capabilities will be based around Web services standards such as WSDL, SOAP, WS-PolicyFramework, and UDDI.

Integration Capabilities

To support SOA in a heterogeneous environment, the ESB needs to integrate with a variety of systems that do not directly support service-style interactions. These may include legacy systems, packaged applications, or other enterprise application integration technologies. This might require support for protocols (e.g., JDBC, FTP, EDI, etc.), adaptors (e.g., J2C or WebSphere Business Integration Adaptors), or for client APIs for various languages (e.g., Java, C+, C#) and platforms (J2EE, .NET, CORBA, etc.).

Quality of Service Capabilities

The ESB will support service interactions with different values to the business; it should therefore be capable of supporting various qualities of service. This might require support for atomic or compensated transactions, various levels of delivery assurance, error and exception handling processes, etc. These capabilities may need to be supported even when service interactions take place through unreliable protocols such as HTTP, and when a service is delivered to a consumer by aggregating several services from different providers. Ideally, these capabilities should be controlled by declarative policies associated with the services involved.

Security Capabilities

The ESB needs to both provide a security model to service consumers and integrate with the (potentially varied) security models of service providers. Both point-to-point (e.g., SSL encryption) and end-to-end security capabilities (e.g., identity pass-through, trust models, WS-Security etc.) will be required. The ESB can also support broader security concerns by providing centrally controlled auditing, logging, or systems management, etc.

Message Processing Capabilities

In a realistic integration scenario, there are likely to be multiple messaging and data models. The ESB will play a role transforming between these, whether between legacy data formats (e.g., COBOL copybooks) and XML, between basic XML formats and Web services messages, or between different XML formats (perhaps transforming an industry standard XML message to a proprietary or custom XML format). This will require the ESB to have some form of message processing or mediation capability, and the ability to manipulate the format and data of messages. In some cases, the ESB

will need to make decisions (such as where to route service requests) based on the formats or content of messages associated with services.

Management and Autonomic Capabilities

As with any other middleware, the ESB will need to integrate with management and monitoring systems. Increasingly, management software will need to monitor business transactions that pass through several service interactions, so the ESB will become a key monitoring and management point. As organizations migrate towards an SOA and onwards to an on-demand operating environment, the infrastructure will need to evolve to perform more sophisticated function, such as billing, demand-based routing, or policy-based behavior (e.g., the ability to select service providers dynamically based on the quality of service they offer compared to the business value of individual transactions). All these imply that the ESB will have an increasingly sophisticated capability to monitor and alter its own behavior.

Minimum ESB Capabilities

Of course, not all the capabilities described above will be relevant to all the situations in which some form of ESB will deliver technical and business value. In that light, we need to consider what the minimum capabilities of an ESB should be in order to address the requirements we considered in the first part of this article, i.e., the ESB should support:

- Decoupling the consumer view of services from their implementation
- Decoupling technical aspects of service interactions
- Integrating and managing services in the enterprise

The minimum capabilities required to achieve these goals are:

- ***Communication capabilities:*** Support routing and addressing for at least one messaging style (such as request/response, or publish/subscribe), and at least one transport protocol that is or can be made widely available. This enables location transparency and service substitution, enabling the decoupling of the consumer view of services from their implementation.
- ***Integration capabilities:*** Support several integration styles or adapters. It should enable services to be provided based on these integration capabilities, enabling the decoupling of technical aspects of service interactions and the integration of services in the enterprise.
- ***Service interaction capabilities:*** Support an interface definition format and associated messaging model (such as WSDL and SOAP) to allow the decoupling of technical aspects of service interactions.
- ***Management and autonomic capabilities:*** Provide a consistent administration model across a potentially distributed infrastructure, including control over naming, routing, addressing and transformation capabilities. This enables the management of services in the enterprise.

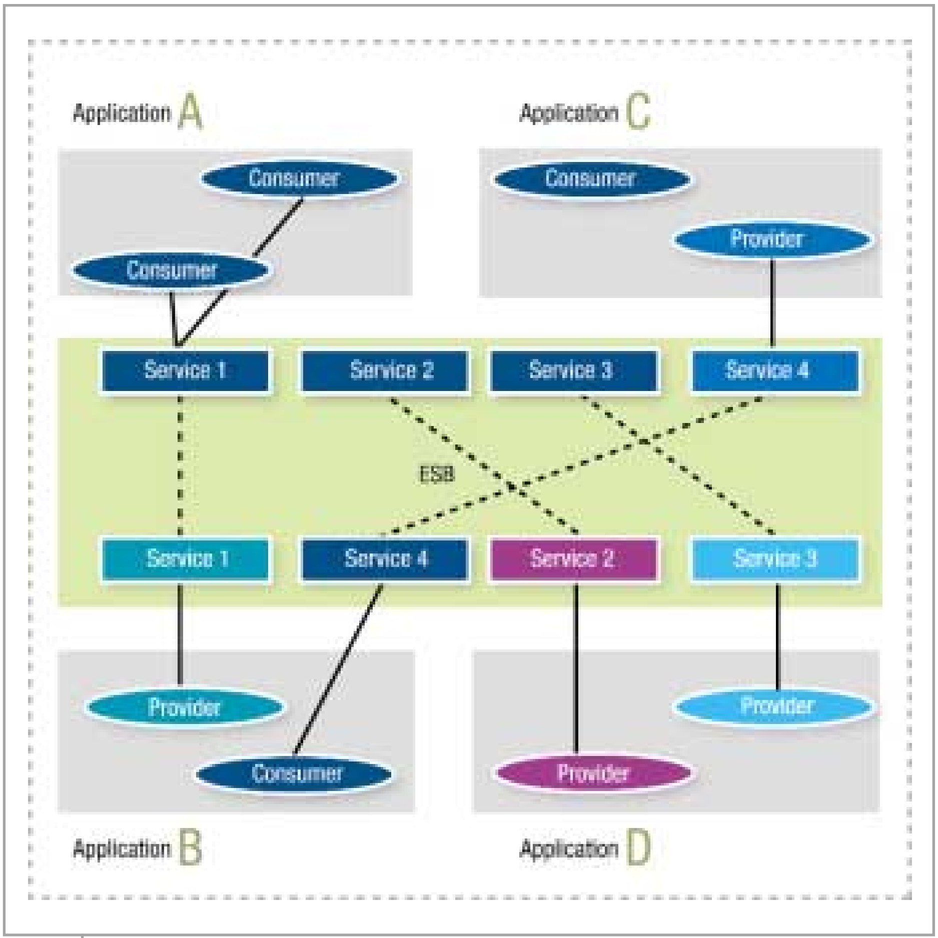


FIGURE 1 **The use of an ESB within an organization to provide a consistent, business-focused means of accessing services (represented in pink) from heterogeneous service providers (represented by varied colors).**

The ESB and Related Infrastructure Components

Now that we understand the requirements for an ESB and the capabilities it provides, we will look at some common situations in which an ESB, or similar infrastructure technologies, might be used, and how that affects the requirements and capabilities. (These uses have been described in more formal terms in the IBM Patterns for e-business [see References]).

The Enterprise Service Bus

The orthodox view of the ESB is of course its use to support SOA within a single organization. It supports the need to reuse function more widely between applications and across lines of business, and to source reusable function both internally and externally. The ESB allows service consumers to access services through an intermediary that presents a business view of available services, regardless of where they are implemented. Figure 1 shows an ESB acting as a service intermediary between applications within an organization.

We can relate this use of ESB technology to our original requirements:

1. ***Decoupling the consumer view of services from their implementation:*** The ESB acts as an intermediary within the organiza-

tion. All consumers source their function from the ESB, rather than from each other. The overall organization is therefore free to change implementations of applications or individual services, without affecting the consumers of the service.

2. ***Decoupling technical aspects of service interactions:*** The ESB provides a decoupling point between consumers and providers. It can transform addresses, security models, assurance models, data formats, etc.
3. ***Integrating and managing services in the enterprise:*** The ESB provides a management capability for interactions between applications and systems. It can provide consistent models for security, logging, and monitoring, or other capabilities, and where necessary map these to the individual models of participating systems.

The ESB Gateway

Intermediary technology such as the ESB can also be applied at a governance boundary – usually between one organization and the outside world, but also within large modular or distributed organizations. Figure 2 depicts this usage. ESB capabilities are required in this situation to fulfill two broad requirements:

- To provide external service consumers with a consistent view of the services available to them, in terms of protocols, service namespace and models of security, transactionality, etc.
- To provide internal service consumers with a consistent view of the external services with which they are permitted to interact, and again provide at least a manageably consistent view of protocols, service namespaces and models of security, transactionality, etc.

This use of ESB technology may be known as a "Service Gateway", a "B2B Gateway," or referred to in several other ways – but the underlying principles are the same as the ESB. In the IBM Patterns for e-business we use "ESB Gateway" as a unifying name that emphasizes the use of ESB technology at a governance boundary, but allows variations in usage such as the "Exposed ESB Gateway" where the boundary is between one organization and the wider world.

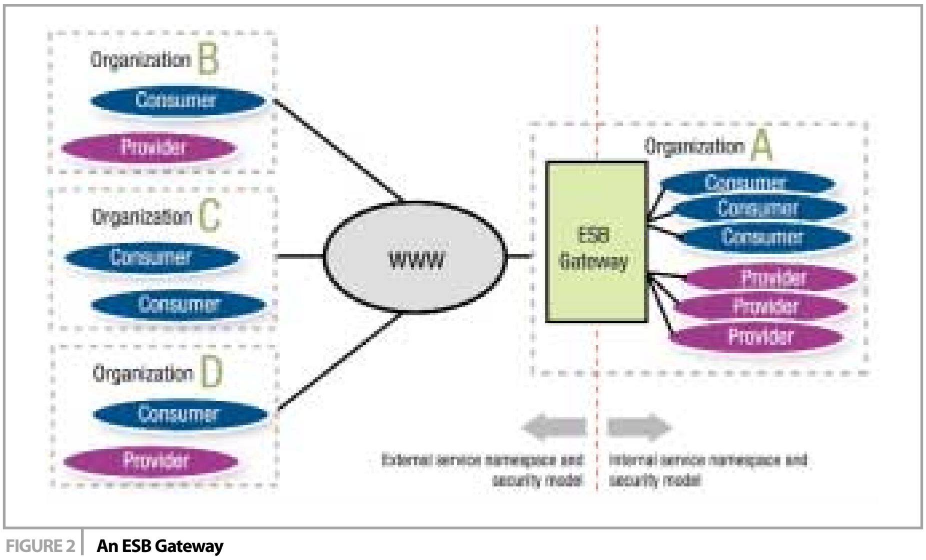


FIGURE 2 | **An ESB Gateway**

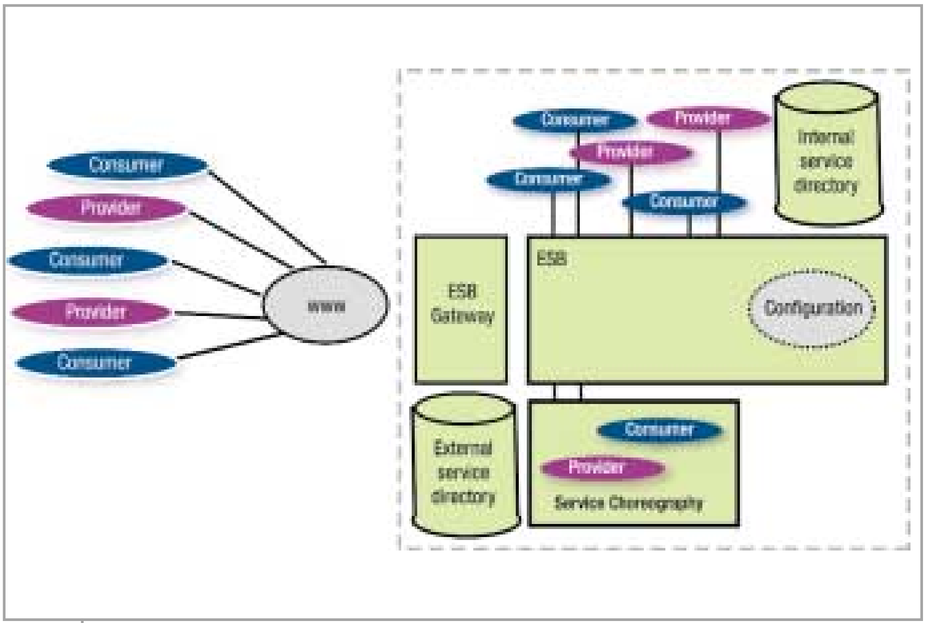


FIGURE 3 | **An overall architecture for SOA**

The ESB Gateway pattern will often emphasize security capabilities, and tends to concern basic routing of service interactions rather than aggregations or composition, but in reality these issues will vary from case to case. The ESB and ESB Gateway components will often be used together when both internal and external services need to be integrated.

We can relate this use of ESB technology to our original requirements:

1. ***Decoupling the consumer view of services from their implementation:*** The gateway provides an intermediary that can publish service definitions to external consumers, and to which those consumers can bind. The organization operating the gateway is then free to change the implementation of services without affecting consumers – it can either change it's own implementation or subcontract services to an external provider.
2. ***Decoupling technical aspects of service interactions:*** The gateway provides a decoupling point between consumers and providers. It can transform addresses, security models, assurance models, data formats etc. to support whatever level of decoupling is required.
3. ***Integrating and managing services in the enterprise:*** The gateway provides a single management point for all external services –

> “The ESB should support communication of service interactions through a variety of protocols”

whether they are provided or consumed from the point of view of the gateway operator.

Other Infrastructure Requirements for SOA

Now that we have seen the role of the ESB and gateway components, we can consider a more complete infrastructure for SOA by adding components such as service directories and service choreographers. We don’t have the space to describe all of these in detail, but by giving a brief overview we can at least place the ESB in a clearer context. Figure 3 illustrates an overall architecture for SOA. The roles of the other components in this architecture relative to the ESB are as follows:

- *The Internal and External Service Directories* catalogue the services available to different groups of service consumers. Service providers will publish services to these directories to be consumed by intermediaries such as the ESB or ESB Gateway. Intermediaries will publish facade services to the directories, to be consumed by service consumers. Note that service directories should be understood to be separate from the service routing information required by the ESB in order to function, although as directory standards such as UDDI evolve, these two roles may become more closely entwined.
- *The Service Choreography Component* aggregates, composes, and choreographs services into larger-grained services or processes. The service choreography component acts as a consumer of facade services provided by the ESB, and as a service provider to the ESB.

Implementation Technologies

The ESB components described above can be implemented either using proprietary technologies, by using open standards, or through custom development. The decisions as to which of these techniques should be used to implement which ESB capabilities are key decisions in implementing SOA today. As we undergo a rapid evolution of ESB technologies, Web services standards, and SOA concepts and techniques, organizations will need to consider a number of issues and how they will evolve over time:

- ***Will an open standards approach offer specific interoperability benefits, and what level of work will be required to achieve that interoperability?*** For example, do all the technologies involved support the required open standards (e.g., perhaps supporting security standards such as WS-Security in addition to basic service interaction standards such as WSDL and SOAP)? Has interoperability been tested, perhaps through compliance to the Web Services Interoperability Organization profiles?
- ***What degree of performance, scalability, integrity, or other service levels are required?*** How do they relate to the capabilities and maturities of standards-compliant versus proprietary technologies?

In this context, a number of implementation options should be considered for the ESB, including both standards-compliant and proprietary integration technologies. In reality, of course, any technology will provide a mixture of standards-compliant and proprietary features; the important task is to match the benefits of both aspects of the technology to individual cases. In this light, some of the implementation options for ESB components include:

- Using specific features of enterprise application integration middleware
- Using “Service Gateway” or “B2B Gateway” technology
- Custom solutions using XML, message queuing and brokering technology, or application server communication and integration capabilities

We don’t have the space here to go into the specifics of individual technologies, their features, or the choices to be made between them. However, the whitepapers and Redbooks in the references contain in-depth information on these topics, consistent with the description of the ESB provided here.

Acknowledgments

Some of this material draws on collaborative work; the following people contributed material to this article or participated in discussions that led to the ideas behind it: Jonathan Adams, Norbert Bieberstein, Bernhard Borges, Beth Hutchison, Rachel Reinitz, Keith Jones, Paul Verschueren, Olaf Zimmerman, Helen Wylie, Kyle Brown, Mark Colan, Paul Fremantle, Daniel Sturman, Dave Clarke, Kareem Yusuf, Chris Nott, and Alan Hopkins.

References

- The IBM Redbook *Patterns: Implementing an SOA using an Enterprise Service Bus*: www.redbooks.ibm.com/abstracts/sg246346.html
- The IBM developerWorks series “Understand Enterprise Service Bus Scenarios and Solutions in Service-Oriented Architecture” parts 1–3: www-106.ibm.com/developerworks/webservices/library/ws-esbscen, www-106.ibm.com/developerworks/webservices/library/ws-esbscen2.html, www-106.ibm.com/developerworks/webservices/library/ws-esbscen3
- *The on demand operating environment:* www-306.ibm.com/e-business/ondemand/us/evolvetech/operating_environment.html
- *Web Services Interoperability Organization:* www.ws-i.org

About the Author

Rick Robinson is an IT architect in IBM Software Services for WebSphere, based at the Hursley Laboratory in the United Kingdom. He has seven years of experience in the IT industry, and his roles have included solution architecture, design, and development. Rick holds a PhD in the physics of superconducting devices from the University of Birmingham. His areas of expertise include distributed systems design, the WebSphere platform, and service-oriented architecture

■■■ rick_robinson@uk.ibm.com

ARE YOUR DISTRIBUTED
APPLICATIONS BROKEN?

Putting the EDA in SOA

How upcoming Web services message standards will merge event-driven architectures with SOAs

The current slate of Web services standards has evolved into a mature set of very useful API's and into service-oriented architectures, or SOAs. Enterprise integration, however, includes many requirements that are not met by SOAs alone. A movement is under way to augment Web services with a new set of standards that address the other side of integration – Event Driven Architectures, or EDAs.

WRITTEN BY
MARC BREISSINGER

The wave that pushed Web services to the forefront of the worldwide developer community's consciousness has left behind a working set of standards that allow organizations to effectively cooperate in processes over the Internet. Web services standards enable organizations to realize some of the goals of integration in a standards-based and, even more importantly, interoperable manner across operating systems, hardware platforms, and application development platforms. While some of those responsible for enterprise-level application development and support have been understandably disillusioned by earlier inter-application communications attempts such as CORBA, Web services have crossed over the often turbulent initial technology adoption stages to become a reality. The standards that have emerged (with the help of organizations like the Web Services Interoperability Organization [WS-I], the World Wide Web Consortium [W3C], and OASIS are not only interoperable but are, even more importantly, being widely implemented throughout many of the Global 2000's IT departments.

Web Services and Service-Oriented Architectures

Many scenarios are well served by the functionality embodied in the current slate of Web services standards. An example scenario: One application requires information such as a credit score from another application. This application uses the SOAP standard to request the information (by passing the DUNS number of the company to check), hold for the answer, and then receive the requested information, at which point the communication between the two applications is completed. This is, on the surface, not a very complex problem and Web services serve it well (see Figure 1).

But what if this were to change later? A process could be designed that executes every week to check up on all of the credit scores of all customers served by a particular application. Maybe the result of this check is so important that a week between updates is too long. Maybe a lowering of a customer's credit rating should be known before completing any sale. This means that every single transaction moving through the application would cause a resulting credit check (these checks cost money, right?) before accepting any purchase order request. This is highly inefficient, especially if the credit ratings of these customers on average change only once or twice a year. This is also very inefficient in that it adds unnecessary load on the computers of the provider of these services, though the provider will not mind at all if they are charging for each transaction.

This on-demand style of information gathering can also cause dependent processes to ask and wait for an answer in places that may not be absolutely necessary. Additionally this method introduces a performance dependency on an outside application. It is very difficult for internal processes with external, ask-and-wait, dependencies to be held to service level agreements.

The "ask-and-wait" method of information gathering is often referred to as synchronous processing – one step must happen after the other, in-sync, for a process to move through on its path to completion. This is the style of process best served by current Web services standards. It's generally very linear in nature. This component-level request/reply functionality is at the heart of the SOA.

But if you go back to the credit check issue, a more efficient method of filling this need would be to allow applications that need up-to-date information to subscribe to this information, which would result in the credit service application publishing update events back to interested applications as the changes occur rather than waiting for the subscribing application to request the information. As Notifications of change are published to the consuming organization, the changes can be placed in a local data structure and can even kick off processes in response to the update event. For example, a particular customer's credit rating is raised a notch by a reputable credit rating organization, such as Moody's. A Notification event is sent to the consuming organization that this credit change has occurred. The consuming organization may wish to respond to this change event with a process that calculates a higher credit limit based on this rating increase. This process would then place the proposed increase in front of a credit officer. This credit officer would then has the opportunity to discuss the new limit with a representative of their customer and to then accept, deny, or change the amount of the credit limit increase. All subsequent orders now take advantage of this newly acquired information (see Figure 2).

This interaction style allows a high degree of trust to be placed on a datastore that is completely under the control of the information-consuming organization, alleviating outside bottlenecks and allowing increased confidence in an organization's ability to comply with service-level agreements and to further control product cycle times. In the high-tech industry, shaving even one second off the cycle from ordering through provisioning, manufacturing, and delivery of a product can save tremendous amounts of money. Additionally, many purchasers that rely on outside vendors for the components of their products do not purchase solely on the price of the goods but often weigh purchase decisions heavily based on a vendor's ability to deliver to a promised schedule. Being able to trust a vendor to deliver what they say they will – when they say they will – allows a customer to execute their business with less inventory, saving money in many ways.

Event-Driven Architectures

Publish-and-subscribe processing and the fire-and-forget messaging at its heart are a part of the functionality embodied in event-driven architectures. The information is updated independently of the processes that utilize the information. The update to the credit scores relieves the business process from having to query for an answer.

The Web Services Notification (WS-Notification) specification, managed by OASIS, will bring the publish/subscribe (EDA) functionality described in the example above to the very SOA-focused current incarnation of Web services standards. This specification enjoys broad industry support from integration industry leaders such as webMethods, IBM, Oracle, and SAP.

WS-Notification is important because it will bring standards-based interoperability across operating systems, hardware platforms, and application development platforms to the messaging world. This has already happened to a very limited extent with JMS where many messaging solution providers are able to achieve some level of interoperability with products from other vendors through JMS APIs. The clear benefit of WS-Notification over JMS is that it doesn't assume that all (or even any) applications utilizing these features will be developed in the Java programming language. WS-notification will additonally offer much greater vendor-level interoperability between messaging infrastructures. This does not mean that JMS will go away; on the contrary, this Web services standard will increase the adoption of EDA functionality between applications and will likely lead to increased demand for all types of messaging infrastructures as long as they can communicate with other infrastructures using WS-Notification.

The current WS-Notification specification provides support for both peer-to-peer and brokered publish/subscribe. In peer-to-peer, publishing and subscribing applications must implement the functionality of the specification (described in the WS-BaseNotification specification). For example, a publishing application must implement a registry to store a hierarchy of publishable document types (Topics) and a

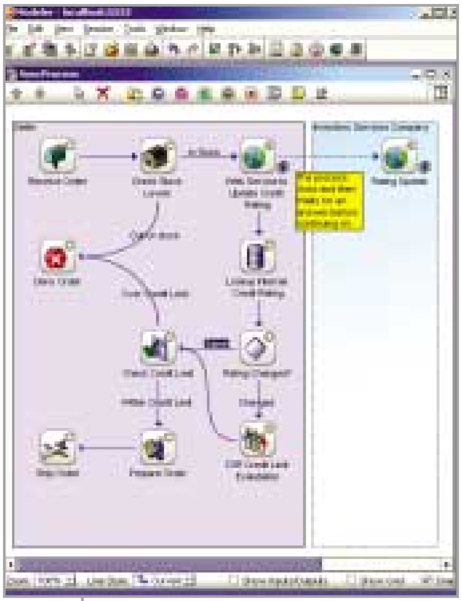


FIGURE 1 **The interaction of a process with an outside service**

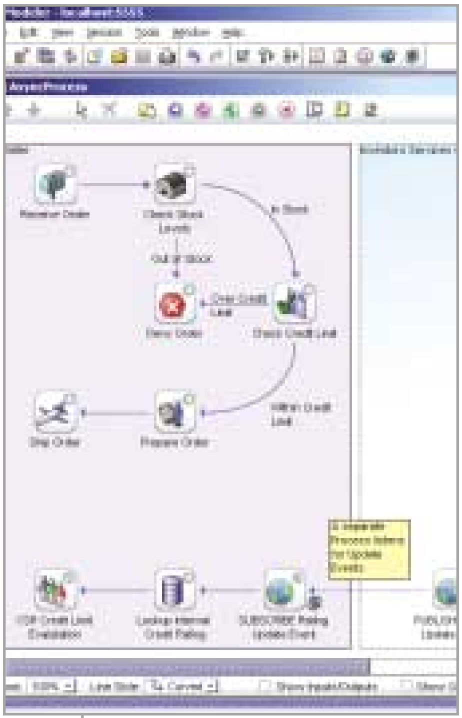


FIGURE 2 **Two business processes operating independently**

directory service to supply this information to interested subscribing applications. Subscribing applications must manage subscriptions that could span many different publishing applications. This is not to mention the support that must be implemented for many of the composable standards that will be required to provide facilities such as authentication, message persistence, message integrity, message encryption, delivery guarantees, etc.

In brokered publish/subscribe (described in the WS-BrokeredNotification specification), some of the burden of implementation of the WS-Notification features and composable features of related standards is off-loaded to a broker. This brokered implementation has one additional benefit, which is that publishers need never know anything about any of the subscribers. This complete decoupling of subscribers from publishers along with the centralized subscription and topic management provides IT departments with more control and allows them to more accurately measure performance against service-level agreements (SLAs).

While the current and planned WS-Notification specification versions bring much needed EDA functionality to the Web services battery of standards, there is still much to be considered before a "pure" implementation of a WS-Notification Broker would stand up to existing mature messaging infrastructures. For example, most current messaging infrastructures have built-in queuing mechanisms to ensure delivery of messages even if the subscribing clients are not available at the moment an event notification is published. This is often referred to as durability.

To illustrate the importance of message durability we will return to the credit check example scenario. After implementing a subscription to credit score changes, the credit scoring information will no longer be requested on-demand from each instance of the order process. How can it be trusted that the information in the locally maintained database is actually up-to-date? What if a change happened to a credit score; a notification was sent but never received? In the previous example, where the credit check is an in-line service in every order process instance, the information (credit score) is assumed to be as accurate and timely as possible. When the credit check is a separate, disconnected subscription process, what guarantees are there that an update has not been missed and that the local datastore correctly reflects the up-to-now credit state of a particular customer?

This information (a changed credit score in this scenario) is so important that it could potentially cost the selling organization a large amount of money if their local datastore had missed even a single credit score update. Some applications of messaging do not require that every single message published by an application be received by a subscribing application. For example, a stock-trading application receives a large number of updates as the price of stocks fluctuates during a trading session. This information, while very important, need not be entirely up-to-date as a price is negotiated for each sale. The stream of updates just gives the purchaser a good barometer of the probable price should a purchase be executed. If one update is missed another will be coming soon that would make the missed message obsolete anyway. If the stock purchaser were away from his/her desk and didn't have their trading software running at the time would it matter that they did not receive stock updates while they were away? No, it would just matter what the latest selling price was. This kind of messaging (where the subscriber does not need all updates, just the latest ones) is often referred to as volatile because it's allowable for them to evaporate if the client is not listening.

In our example, however, it is important that even if the seller's datastore server has crashed and is therefore not listening for credit score updates, that they be stored for eventual retrieval. This is called non-volatile, or durable, messaging – messages must be maintained even if the subscribing client is not listening. Generally, support for durable messaging means that messages intended for offline subscribers are stored until the subscriber is ready to receive them.

With volatile messaging publishing applications and subscribing applications don't need to know anything about each other except for an agreement on the type of message that they are sharing and the mode of transportation. Durable messaging, on the other hand, requires quite a bit more logical cooperation between publishers and subscribers.

Durability has been identified by the WS-Notification Working Committee at OASIS as an important feature that will be addressed in an upcoming version of the specification. With Durability will most likely come other very important enterprise-class messaging features such as shared queues (for balancing of load), ordering (where a client can be sure that it is receiving messages in the exact order in which they are published), policy enforcement (for control of access to particular message types and content), and authentication, just to name a few.

The current incarnation of WS-Notification has been built to address the most ubiquitous pattern in EDAs, volatile publish and subscribe. The future of WS-Notification will bring us not only interoperability between different messaging infrastructures but will also make it much easier to extend the reach of messaging solutions out over the Internet. The work being accomplished by the WS-Notification Working Group is likely to have a profound effect on Web services and on the future of integration. As companies investing in integration solutions become more and more confident that these solutions will interoperate with others, reducing risk, they will invest more in integration.

About the Author

Marc Breissinger joined webMethods in 1999 and is currently vice president and chief architect. He manages webMethods' standards activities and defines the product strategy of their solutions. In his current role, Marc is responsible for shaping webMethods' technology vision and ensuring that vision is translated into industry-leading products. webMethods, the industry's first Web services infrastructure company, is a member of the Web Services Notification working group and is coeditor of the upcoming Web Services Notification specification version 1.2.

■■■mbreissinger@webmethods.com

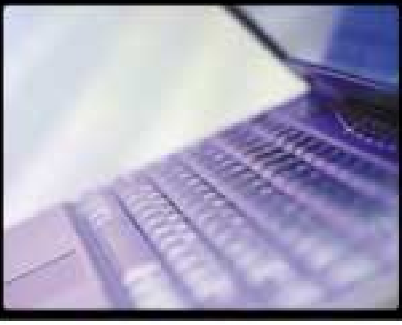
The Gilbane Conference
ON CONTENT MANAGEMENT TECHNOLOGIES

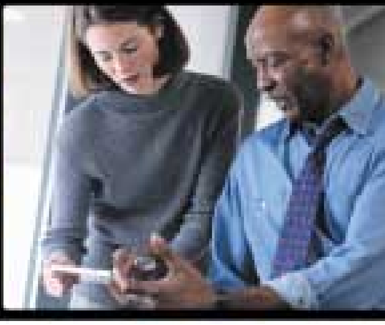
November 30-December 2, 2004
Boston, Massachusetts
The Westin Copley Place

Lighthouse Seminars

The
GILBANE
REPORT

ICROSOFT, ADOBE & XFORMS TO
AKE UP ELECTRONIC FORMS MARKET





Content technologies have become mainstream and need to be part of all major enterprise applications and integrated into IT architectures and infrastructures. Join us to learn what you need to know to formulate a successful strategy from our gathering of content technology experts and practitioners.

What you will take away from the conference:

Attendees benefit from an unbiased, deep, and up-to-date understanding of content management technologies, vendors, trends, and best practices, from the most experienced and respected experts in the field. Our speakers have implemented every kind of content management system across all industries, and have written the books and reports that others depend on.

WHO SHOULD ATTEND:

IT Professionals

- IT Strategists, Managers, Staff
- Content Management Project Managers
- Content Management System Designers
- Intranet, Internet, Extranet, Portal Managers
- Webmasters, Developers, and Administrators
- Enterprise Architects

Business Managers and Strategists

- Product Data Managers
- Information Architects/Knowledge Managers
- Technical Documentation Managers
- Business, Market, and Technology Analysts
- Consultants and Integrators
- Marketing and Product Executives
- Brand Managers

Register NOW!

Register now for the **CONFERENCE PLUS PACKAGE** and receive a **FREE iPOD!**

LIMITED QUANTITY, REGISTER EARLY!

LEARN about successful implementation of content technology through the unique combination of

- **Case Studies**
- **Implementation Techniques**
- **Technology Analysis**

"The Gilbane Conference has provided me with crucial information that I was looking for as my company rolls out its content management solution. I have been able to network with industry experts, the major content management technology players and other companies, like mine, that are faced with the challenge of effectively managing content," says Angie Khumdee, Senior Software Engineer, Motorola CGISS E-Business

For more detailed information, please visit:
www.lighthouseseminars.com or call
Joe Richard at **781.821.6734**

Gold Sponsors

EMC² documentum
GMC
INTERWOVEN
STELLENT™
VIGNETTE™ the e:fficiency experts™

Media Sponsors

Business Wire Smart Solutions for Your News
cms WATCH
EContent
The GILBANE REPORT
InfoWorld GET TECHNOLOGY RIGHT
TRANSFORM
WebServices
XML JOURNAL

Association Sponsors

IDEAlliance® International Digital Enterprise Alliance
OASIS

Forum Systems XWall Web Services Firewall

A solid security solution

Security is important. Anyone in the business of designing, developing, hosting, or managing business applications understands this fundamental statement. Web services present unique challenges such that the integrity and security of the content of the exchanged documents is just as important as the integrity of the communications link between the trading partners.

WRITTEN BY
BRIAN BARBASH

Addressing this broad definition of Web services security is the Forum XWall Web Services Firewall from Forum Systems. Within a network topology, it serves as the entry point to an enterprise's collection of Web services and is available as a hardware or software component. As its name implies, the product serves in a traditional firewall capacity such that it may be used to protect resources from external requests. However, it also provides functionality that addresses the security of the content passed between the host and client.

The Forum XWall system is a highly configurable security tool that provides several components enabling secure Web services. Its network security capabilities include PKCS keys, public certificates and SSL, Access Control Lists, IP filtering, and custom error-handling templates. From a Web services point of view, the security functionality includes intrusion detection, WS-I validation, request filtering, and system alerts. This review looks at the software version of XWall.

Securing Web Services Network Perspective

The firewall provides a Web-based interface for configuring all security parameters. All settings are grouped into the Administration, Resources, and System categories. The Administration section contains the Getting Started instructions (see Figure 1), monitoring functionality, and general gateway policies. The Resources section is where administrators set up the key repository, SSL security policies, access control settings, and error templates. The System section includes settings for the operation of the firewall itself, logging, and configuration import/export.

Basics

In the most basic setup, there are two main steps to securing Web services:

1. Create network policies
2. Establish Web services policies

Network policies, or HTTP server policies, are either local or remote and provide the channels through which network data travels. Local policies protect resources from incoming traffic. Remote policies act as proxies to services on tertiary systems.

The local policies establish the ports that will accept incoming traffic and provide the network-level security functionality. There are five components to this listener when working with the HTTP protocol:

1. List of client IP addresses allowed to access services
2. Protocol used to access services – HTTP or HTTPS
3. Listener IP address, port, and whether basic HTTP authentication is required
4. The Access Control List to apply
5. The template used for error messages

Once incoming network traffic has met the requirements of the local policies, it is passed through to the remote policy. Remote policies are used to configure access to the actual Web services applications hosted on additional servers. There are three components to this policy when working with the HTTP protocol:

1. Protocol used for outbound communications – HTTP or HTTPS
2. The IP address or hostname of the machine on which the desired services exist along with the port and basic HTTP authentication settings
3. A flag indicating whether or not the response from the remote service is to be processed. When turned off, the remote service's response is returned to the calling client unchanged.

For this review, I have established a basic local policy. It establishes a listener on port 8080, restricts IP addresses to a segment of my network, uses the HTTP protocol, and requires basic authentication. I've associated a simple Access Control List with this policy that provides read and execute permissions to a group of one user. I will discuss the remote policy later.

To demonstrate error conditions presented by the local policy, two SOAP messages

Company Info

Forum Systems
45 West 10000 South, suite 415
Sandy, UT 84070
801-313-4400
Fax: 801-313-4401
Toll Free: 1-866-333-0210
sales: twise@forumsys.com

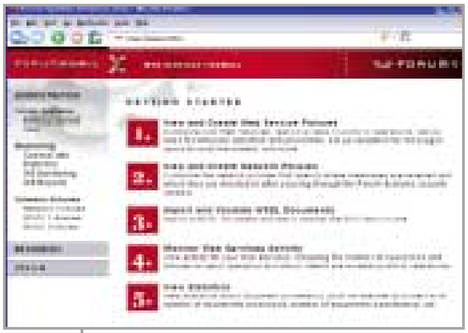


FIGURE 1 **Forum Systems XWall Console: Getting started**

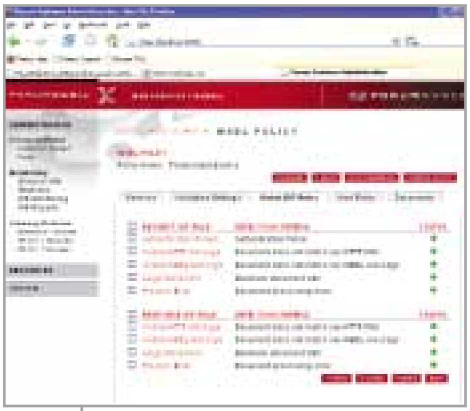


FIGURE 2 **IDP rule settings**

were sent; one from an IP address that falls outside of the security policy and one with incorrect credentials. As expected, the server responded respectively with 403 and 401 HTTP status codes.

Access Control Lists

As I mentioned in the previous example, Forum XWall supports Access Control Lists to restrict user activity. Users may be defined directly in the Web console or may be imported from an LDAP server. For users from an LDAP server, passwords may be imported in either MD5 or SHA encrypted format. Alternatively, system administrators may choose to have user passwords checked dynamically against the LDAP server at authentication time. Once created or imported, users may then be added to groups, which in turn are assigned to various Access Control Lists. Lists are easily assigned to local server policies during the setup of each policy.

Securing Web Services: Content Perspective

Not only does the Forum XWall Firewall provide network level security, it provides security at the Web services message level. Content is protected via WSDL policies, which are derived from the WSDL documents of the services that clients will ultimately access. Essentially, the WSDL file of the desired service is imported to Forum XWall. As an example, I've imported a WSDL file for a temperature service from Xmethods.net. Once the document is imported, the administrator must choose the listener policy that should be applied to this service. For this example, the policy defined earlier will be applied. The next step in the process is establishing the remote policy for the service.

Remote policies are established to provide the pass-through to the actual Web Service to be executed and have similar configuration parameters to local policies. When working with Web services that require basic HTTP authentication, the administrator may choose to propagate credentials provided initially by the client if challenged, or to use a predefined set of credentials.

Once the basic policy is established, Forum XWall's key strengths are available to the administrator. At this point, any operation defined in the imported WSDL file may be enabled or disabled to calling clients. Additionally, separate ACLs may be applied to each operation. This provides for a very flexible access control policy for all configured services.

> "The system effectively addresses the problem of securing Web services applications from both a network and content perspective"

Forum XWall also addresses the security and integrity of the content of SOAP messages exchanged between the client and service. One of the key features is the ability to perform runtime validation of SOAP messages against the WS-I Basic Profile 1.0 specification. For each WSDL policy in the system, WS-I profile tests may be selectively applied to the messages as they pass through the firewall. For any exchange including a document that does not fulfill the tests configured, a SOAP fault is generated and sent to the calling client.

Another powerful feature of the firewall is the Intrusion Detection and Prevention (IDP) rules that may be applied to WSDL policies (see Figure 2). By default, the firewall comes configured with rules to detect authentication failures, invalid HTTP messages, SOAP documents not conforming to any configured WSDL specifications, document processing errors, and documents that exceed a predetermined size.

After all security parameters have been set within a WSDL policy, the service must be made available to calling clients. This is done by publishing a new WSDL document derived from the local, remote and WSDL policy settings configured. Forum XWall provides the option to export the WSDL document as a file or to upload it to a UDDI server.

As an example, I've configured the temperature service with a document size rule to reject any message over 1 byte. All calls to the service received SOAP faults indicating the error. For even higher levels of security, the system may be configured to fail silently and not return a response to the calling client at all.

Summary

Forum Systems XWall Web Services Firewall is a powerful security solution targeted to Web services. The features covered in this review represent only a small portion of its overall capabilities. The system effectively addresses the problem of securing Web services applications from both a network and content perspective. Overall, this is a very solid product that should be considered for Web services applications.

About the Author

Brian R. Barbash is the product review editor for *Web Services Journal.* He is a senior consultant and technical architect for the Envision Consulting Group, a management consulting company focusing on the contracting, pricing, and account management in the pharmaceutical industry.

■■■bbarbash@sys-con.com

Automating MISMO Processes

Web services bring new advantages

■ Implementing industry standards for business processes can do far more than provide a common protocol for operations. Once commodity information or documents are standardized, it makes sense to look at what common actions need to be taken on that data or document – and standardize those as well. From the technology industry's RosettaNet standards for supply chain interaction, to the Automotive Industry Action Group's interoperability initiative for inventory visibility, industries are advancing on standards for implementing business actions and for executing business processes.

The advantage of standardized process lies in the types of business automation it makes possible. With document and communications standards that are relatively mature, and a number of tasks – from credit requests to title, mortgages, and flood insurance checks – that are common to every lender, the mortgage industry is one in which automation is an attractive option with benefits ranging from time and costs saved to more secure transactions.

We're helping mortgage banking customers use Web services to do more than automate simple message exchange, and speed and streamline their operations. From our experience, we anticipate interest in MISMO standards–based interoperability will quicken deployment of Web services over the next 12–18 months. This article explains how Web services can be used to facilitate standards-based operation, using MISMO guidelines as examples.

WRITTEN BY
J. SCOTT BUSHEY

TYSON HARTMAN

What Is MISMO?

According to Fannie Mae, the nation's largest source of financing for home mortgages, in 2002 secondary market investors purchased about 73% of all mortgage loans made in the United States. That year, Fannie Mae also issued its first requirements for the electronic creation, sale, and delivery of mortgages, based on the Mortgage Banking Association's (MBA) Securable, Manageable, Archivable, Retrievable, and Transferable (SMART) document data format standard. The SMART Doc standard was designed to bind together data, page view, and signature(s) into a single electronic file.

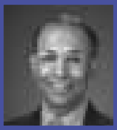
The Mortgage Industry Standards Maintenance Organization (MISMO) has taken the SMART Doc standard a step further, toward standard transactions. The loan origination and management process is a complex one, and MISMO standards for data format are intended to make it easier for banks to interoperate with customers and partners, and enforce regulatory requirements such as encryption and storage of mortgage data.

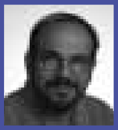
Adoption Drivers

MISMO standards are conducive to the use of technology to automate processes, but they don't specify which technology to use or how to deploy it. Why would or should a lender consider adopting MISMO standards? There are several very compelling business and technological benefits.

When a mortgage is made in the primary market, the lender can hold the new loan, sell the loan to the "secondary mortgage market," or package the loan with other loans and exchange them for securities such as Fannie Mae Mortgage–Backed Securities (MBS). Fannie Mae and other investors, including other banks, make up the "secondary mortgage market." The Electronic

OPPORTUNITY
THREAT





Signatures in Global and National Commerce Act, or ESign Act, signed into law in October 2000, helped increase the industry's interest in paperless transactions and in MISMO standards. The secondary market has come to accept image documents in lieu of paper as well.

Market events reveal some of the business benefits of adopting MISMO standards. In recent years low interest rates caused the volume of applications to skyrocket. Electronic processing seemed like a good idea to customers in the secondary market, who understood faster, accurate loan processing would help with the spike in volume – and generate more revenue. Freddie Mac and Fannie Mae, the largest U.S. home financing organizations and the biggest purchasers of loans on the secondary market, began encouraging lenders to adopt these standards.

Electronic processing also promises long-term benefits of efficiency, security and flexibility. Manual underwriting is time-consuming, and so reduces profit per loan. It also increases risk for errors and even lost documentation, as well as creating opportunities for fraud. Automation of manual activities contributes to faster, more accurate processing. Some forms of fraud are eliminated by the replacement of physical handling of documents with secure transmission over public networks. Electronic processing also establishes a common set of functions that afford flexibility, in that they can be used again for other tasks.

These benefits underscore the attractiveness of automation as a long-term strategy. Whether consumer demand swells or shrinks, participants in the secondary market benefit from processing turn time and a secure environment – the faster they can sell the loan and the less risky the process, the better their profits.

Where Do Web Services Fit In?

Web services seem like a logical way to harness MISMO-standard data and documents to complete transactions. First, though, it's helpful to understand what standards are contained in MISMO.

MISMO specifications encompass two categories: documents and data sets. The SMART Doc standard referenced earlier pertains to specific records that are expected to be kept on file, whether that file is paper or electronic. For example, property title proves ownership. Just as there are rules that specify margins and the location of text for a paper title document, the SMART Doc standard is meant to provide assurance that any electronic documents and action taken on them – signature, for instance – adhere to an industry-accepted standard.

The MISMO data set standards are, in essence, XML-based data format protocols, and they include:

1. An XML architecture encompassing data origination, secondary market, and servicing data.
2. A data dictionary including business definitions and corresponding architecture data element tag names.
3. A data model to serve as a reference tool for the development of XML document type definitions (DTDs), by illustrating the relationship between data elements in the logical data dictionary. (*Note*: The data model is not an XML implementation of the MISMO specification; however, an XML schema is forthcoming in the third version of the MISMO standards.)

The data set standards and the tasks they serve comprise the origination, approval, and secondary sale, recording and secure storage of loan-related documentation. MISMO standards address:

- ***Origination activity:*** Automated underwriting request, mortgage application, and closing
- ***Loan servicing (e.g., collecting monthly payments):*** Servicing setup and transfer
- ***Related services:*** Request for credit, flood, title, and mortgage insurance
- ***Infrastructure for paperless transactions or "eMortgages":*** Data elements and electronic signature capabilities

There are other aspects of mortgage banking for which MISMO standards do not yet exist but that are in development, for investor accounting, appraisal, and directory tools to locate eMortgages.

Most of the MISMO data sets imply a process, suggesting that Web services might be a good way to execute tasks using these MISMO standards. In addition, closer examination shows many of the document type definitions – and the processes they relate to – are interdependent. That interdependency suggests the same documents will be called for a variety of tasks, and that it would be efficient to establish a set of services – that is, repeatable processes independent of the data itself.

Using Web Services for MISMO: Compliant Credit Request and Response

Broadly speaking, MISMO standards for data sets stipulate the format for information that banks want to pass back and forth to complete a process or transaction. For example, data is required to initiate a credit request, but in and of itself the data does not represent a credit report (which would be a document). However, data sets in turn can be classed as information that is simply transferred, and as information that fulfills a request.

To execute a MISMO standard, a service description using WSDL would initiate the action that creates or consumes these data sets, automating the process of transacting various elements of a loan and contributing to the end result, a SMART Doc–formatted record.

The Best Approach

As stated, many of the standards for the mortgage process are document focused. For example, "transfer servicing" is defined in XML. But other standards imply process. Operations such as "credit request and response" and "title request and response" are good examples.

Closer examination of the characteristics of each standard – and what it's intended for – will provide clues to the best approach to a Web service. There are two options for designing a Web service to exchange data. The document-literal approach specifies the format and content of data and tends to be used for message-oriented exchange. The remote procedure call (RPC) approach invokes action on the data conveyed, and tends to be more stateful. Document-literal transactions have a smaller payload and are easier to implement, while RPC exchanges are more complex to orchestrate. Another benefit of message-oriented exchange is the flexibility to add and modify content without breaking the service contract.

Because so many MISMO specifications relate to documents, the document-literal approach makes sense for most data sets. We will use it here as a Web service implementation for a request/response data set, such as a credit query. Many other tasks involve request/response data sets, such as mortgage insurance and flood insurance. Not surprisingly, many (DTDs) within the MISMO standards are related.

There are three basic steps to implementing a MISMO standard as a Web service: creating the XML data set according to the standard's definition, organizing the XML data set so that it will be created or consumed consistently, and then writing the WSDL.

The code snippets in Listings 1-3 (these listings are online at www.sys-con.com/webservices/sourcec.cfm) illustrate the steps to an implementation of the MISMO Credit Request v2.3 document-type definition. Listing 1 shows the basic format of the request/response data sets, which comprises REQUEST_GROUP, REQUEST, REQUEST_DATA, and SUBMITTING_PARTING elements and attributes.

Next, this DTD must be organized, that is, converted to an XML schema definition (XSD). For this exercise, we converted the code using a Java-based utility called Trang, and Listing 2 shows the outcome. This conversion capability is slated to be built into the next major release of Microsoft Visual Studio .NET 2005. Nevertheless, the conversion from DTD to XSD takes just a few minutes.

The third and final step to creating a Web service is accomplished by using a current Java or Microsoft .NET development tool and this schema definition as the input parameter, to produce the document-literal Web service implementation (i.e., WSDL).

The Web service in Listing 3 contains one method, ProcessCreditRequest(), which accepts as an input parameter XML that adheres to the defined schema, and returns a code indicating success or failure.

From DTD to WSDL, the development process can be completed in just a few hours using the current generation of Java or Microsoft .NET development tools. In response to the credit request Web service, a trading partner would direct either their Java or Microsoft .NET development tool at the WSDL file and generate the code necessary to consume the Web service. This capability underscores the ease with which MISMO standards can be implemented as Web services.

Best Practices

Ease aside, certain best practices can enhance collaboration with trading partners on a Web services project, and implementing MISMO standards is no exception. Agreement on several fundamental aspects of interaction will streamline the effort. These include what data will be exchanged and the extent to which MISMO standards will be used for that data.

Trading partners would do well to inventory each others' skills and even certain aspects of their infrastructure. This will reveal ahead of time any differences in staff availability, technical understanding, and familiarity with the MISMO standards, not to mention past experience with Web services and related technology.

Arranging for access to trading partners' infrastructure will help smooth the way for the project as well. We find it's advantageous for each team to include a member with access to the infrastructure. Web services implementation runs counter to conventional development wisdom, in that it requires development work to be exposed to the outside world. Any number of unforeseen access-related issues can be eliminated quickly by the team member with infrastructure responsibility, from changing firewall settings to administering network and machine configuration for a stable connection to partners in the outside world.

Benefits

By implementing MISMO standards–based message exchange using Web services, banks can establish a platform for automation. And the advantages of automation are astounding – whether or not the method includes Web services. At an MBA conference in March, one financial institution estimated it now processes twice as many loans per underwriter at one-fifth the cost, is 30% more efficient in post-closing review, and cut processing costs by about 75%.

Web services make it easier to add incremental functions to automated processing as well. MISMO recognizes that banks may need some way to automate aspects of the mortgage process that do not yet have MISMO standards. Rather than develop a flavor for each process that involves a request/response transaction, such as credit, mortgage insurance, and homeowners' insurance, MISMO has developed the "enveloping" standard. It stipulates rules for the beginning, middle, and end of a data set, and makes it easy to develop WSDL that can be reused. We find most organizations choose to start with a credit request – a fundamental aspect of any lending transaction – and repurpose the WSDL for another type of request/response function, such as mortgage insurance.

Additional Resources

Organizations and tools abound to help companies build Web services to execute standards-based tasks. These are just a few:

- *Mortgage Industry Standards Maintenance Organization:* www.mismo.org. For a complete definition of the credit request/response standard, see www.mismo.org for the credit request/response action.
- *Mortgage Bankers Association:* www.mbaa.org
- Association for Cooperative Operations Research and Development (ACORD): www.acord.org
- *Web Services-Interoperability Organization:* www.ws-i.org
- *More information about Trang:* www.thaiopensource.com/relaxng/trang.html

About the Authors

J. Scott Bushey and Tyson Hartman are consultants with Avanade Inc., a technology integrator for Microsoft solutions in the enterprise. Bushey, an Avanade Principal Solutions Developer, is responsible for architecting and developing .NET solutions, focusing on Web services. Hartman is an Avanade Fellow, one of Avanade's senior-most technologists responsible for guiding Avanade's .NET vision and go-to-market solutions.

■■■ info@avanade.com

Secrets of Designing a Service

Applying basic principles

Software design has always been a focus for developers, but as we cycled through different approaches, standards, and architectures over the years, I think we've had a tendency not to pay enough attention to the fundamentals of software engineering. Clearly I've seen a decline in software quality because of this, not from a lack of programming talent, but a lack of upfront architecture and design. Skip this step and your service will cost much more to build and deploy, as you find yourself in an interactive death spiral that's difficult to recover from.

WRITTEN BY
DAVID S. LINTHICUM

With the movement towards SOA, and the use of services to assemble and integrate software, we have to pay particular attention to design. Indeed, services have many unique design patterns, a bit different than traditional software systems, these include:

- Reuse
- Heterogeneity
- Polymorphism
- Aggregation
- Limited scope
- Standards based

First and foremost, services should be designed for reuse. Services become apart of any number of other applications, and thus must be designed to provide behavior and information, but not be application specific.

Services have to be designed for *heterogeneity*. Web services should be built so that there are no calls to native interfaces or platforms. This is due to the fact that a Web service, say one built on Linux, may be leveraged by applications on Windows, Macs, even mainframes. Those that leverage your service should do so without regard for how it was created, and should be completely platform independent.

Polymorphism, although an object-oriented programming term, means that we need to design a service with the ability to process services differently depending on their data or context. This facilitates reuse.

Also, when we build or design services we need to account for *aggregation*. Many services will become parts of composite services leveraged by an application, and thus you must consider that in their design.

Services are not applications and should have *limited scope*. In other words, they do simple things such as checking inventory or calculating reorder points. If your needs are more complex you simply write more services and don't overload a single service with too much functionality. Services with too much functionality are considered heavy and are difficult to reuse since you may deploy a service where you're only leveraging 10% of its functions. Lighter, or more granular, services are much easier to reuse.

Finally, services need to be designed as *standards based*. While in the world of Web services this seems like a no-brainer, many developers and architects ignore compliance with standards and thus limit interoperability.

Checklist for Designing a Service

Now that we understand the common design patterns we must follow, the question is, how do you design a service? There are certain steps architects and developers can follow; here are some suggestions.

First, you need to define the purpose of the service. What the service will do, and who is the intended user.

Possible Artifact: Service definition document

Second, you need to determine the information to be bound to the service including both metadata and schemas. This means you need to understand how information is leveraged by the service, and what functions require what data. Typically services are storing information outside of themselves, thus you also need to design the mechanisms for accessing outside data sources. Moreover, you need to define data policies here, including data validation constraints and dependencies.

Possible Artifact: Metadata and Schema Document

Third, you need to determine the functions (methods) encapsulated inside the service, in other words the behaviors you would like to expose. For instance, if our service checks inventory it may expose:

```
CheckInvenotoryOnHand(product_ID)
CheckInventoryReorderPoint(product_ID)
CheckInventoryOnOrder(product_ID)
```

It's also at this step that we define each function, including how the function breaks down using a traditional functional decomposition chart. This means that we define the higher-level function by defining all lower-level functions. Structure charts or decomposition diagrams are very helpful here.

Possible Artifact: Structure chart

Fourth, we need to define any interfaces into the service as both machine and human. This means we need to determine how the service will interact with the calling applications, and through what mechanisms. While Web services define the mechanisms for both interface discovery and communications (e.g., WSDL and SOAP), we need to determine what those interfaces are and what they do. In many instances they map directly back to the functions, but not always. Moreover, with the use of rich client interfaces in many instances you may be interacting with a human through a portal-type interface; you need to define that here as well, including design of the user interface.

Possible artifact: API design, user interface design

Finally, we need to define how the service is to be tested. This is an important but often neglected step where you define how those leveraging the service will test the service within the context of their usage pattern. You need to define test information, service invocation, and validity of results. Even performance profiling should be included in this step.

Possible Artifact: Test plan

Other Design Considerations

Of course there are other things you need to design into a service, including management and security.

You need to design a service with points of management to allow those that leverage the service to include it into their management infrastructure. Designing for management means designing and building points of management into the service, management APIs really, and defining their usage to those that leverage the service.

Security is a bit more of a systemic design issue, meaning you need to define how your service is accessible and how you intend on authenticating users. Typically this means identity management facilities, but what's most important is that you define the security parameters to those who are looking to leverage the service and that security is designed in.

Of course I can't discuss all of the detailed design procedures one would employ in designing a service, but I think I've hit on the more obvious things here. What's important is that you actually apply some fundamental software engineering and design principles here so you create a service that's able to meet your needs the first time. Those who skip that step eventually have to return to it to fix or add to the service later. This interactive approach gets costly quickly and only delays the delivery of the working service. So, always focus on the fundamentals.

About the Author

Dave Linthicum is the CTO of Grand Central Communications (www.grandcentral.com) and has held key technology management roles with a number of organizations including CTO of both Mercator and SAGA Software. David has authored or co-authored 10 books, including the groundbreaking and best-selling *Enterprise Application Integration* released in 1998. His latest book, *Next Generation Application Integration, From Simple Information to Web Services* was just released.

■■■ e-mail: linthicum@att.net

The Best
.NET
Coverage
Guaranteed!
*Offer subject to change without notice
Only $69.99
1 year (12 issues)*
* Newsstand price $83.88 for 1 year
Subscribe
ONLINE
www.wsj2.com or
CALL
888 303-5252
SYS-CON MEDIA
SYS-CON Media, the world's leading i-technology publisher of developer magazines and journals, brings you the most comprehensive coverage of Web services.
WebServices JOURNAL

Web Services Security Hype

If we're going downhill, that means we're gaining momentum, right?

According to the latest Web services "hype cycle" from Gartner, both Web services security standards and the deployment of Web services with security are rushing headlong into the dreaded "Trough of Disillusionment." This means that the greatest levels of hype in these areas are supposedly behind us and the reality of just what can and cannot be done is collectively dawning on us.

WRITTEN BY
MICHAEL MOSHER

Taken at face value, this news could be either good or bad. The good news could be that now that the hype is over and we have passed the lofty "Peak of Inflated Expectations," we can all get down to the serious work of putting together workable security solutions and solid security standards to help bring Web services to where they deserve to be. The bad news could be the security components of Web services getting mired in the "Trough of Disillusionment" for too long and losing their appeal for the enterprise.

Rightful Place?

One question we should ask ourselves is, do the Web services security categories belong where Gartner has placed them on the hype curve? There are a number of ways that we can look at it. One way is to examine the position of the security elements on the hype curve relative to their peers. The security pieces still have a long way to go to catch up with established components of Web services, such as SOAP and WSDL, which are already on the "Plateau of Productivity" and are on the verge of exiting the hype cycle as they approach full mainstream adoption.

Another way to look at it is to consider how these components are progressing over time. If you look at where they were placed on the curve at the same time last year, the security elements have been big movers – especially when compared to some other areas, like UDDI, which have been essentially frozen in place. In last year's hype cycle, Web services security standards had not even made it to the top of the "Peak of Inflated Expectations." In just a year's time, by Gartner's estimation, the security standards have made respectable advances toward broad acceptance and implementation; secure Web services made a roughly equal advance along the curve.

Unfortunately, the path to productivity must inevitably pass through disillusionment, which is where Gartner sees the current state of Web services security. To really make a judgment about whether Gartner has made the right call about where we are with security, and where we might be heading, it is useful to understand how we got here.

Keeping the Momentum

Gartner's hype cycle assumes that all new technologies will eventually hit some rough spots in their life cycle, especially if they fail to meet the lofty expectations that are so often set for them early on. This is certainly true of Web services security. As Web services took off, there seemed to be no shortage of efforts to answer the need for securing this new paradigm. Creativity, and even unprecedented cooperation, appeared to be the order of the day: rival authentication standards banding together to create SAML; Microsoft and IBM joining hands to chart out a whole family of standards; promises of quick action from standards bodies to "fast track" Web services security standards; dozens of companies responding to the call to create technologies for implementing the standards. Plus, the newly conceived security standards showed bright promise for applications far beyond the world of Web services. It all felt so good, we should have known it would have to end. Competition, old rivalries, "standards bloat," and many other factors have served to pull us collectively back to reality. For example, Microsoft and IBM started to see different directions for their WS-* roadmap. And, the notion that standards would make security products interoperable right out of the box remains a dream for most. So, perhaps the assessment that we are in a state of disillusionment around Web services security, if it is off the mark at all, is not off by much.

But that doesn't mean this is the end of the story. The descent into disillusionment could

> "do the Web services security categories belong where Gartner has placed them on the hype curve?"

> “So, how can we ensure that security stays on track to help Web services deliver on their promises?”

mean that real productivity and value from Web services security is just over the next rise. The danger is that if momentum is lost, these key components of the Web services world could suffer the same fate that intrusion detection technologies have suffered in the larger security space – a permanent place in the “trough.” Losing momentum at this critical juncture could have dire consequences for security in Web services and the usefulness of Web services as a whole.

The Next Big Step

So, how can we ensure that security stays on track to help Web services deliver on their promises? I see three things that we can do for a start:

- ***Keep it real:*** If we can properly manage our expectations and not fall back on the overblown hype of the past, then our disillusionment will likely be short-lived;
- ***Close the gap:*** Right now, Gartner shows deployment of Web services with security as being a good deal further along in the cycle than the Web services security standards. This is a dangerous gap since it could indicate that many Web services security deployments are not using standards. I hope it is actually more of a matter of definitions, since Gartner considers the use of Secure Sockets Layer (SSL) encryption to constitute a Web service deployed with security (most robust applications Web services require much more to be secure). Whether the gap is real or just a gap in understanding, we must work to close it;
- ***Maintain the focus***: If those who are experimenting with or adopting Web services for use in their environments keep a strong focus on the importance of security to Web services, then the momentum should be able to carry these components over the hump.

It will be exciting to see if this next year turns out to be one in which the security pieces of the Web services puzzle at last snap firmly into place.

About the Author

Michael Mosher is the technology director of the CSC Consulting Business and Technology Risk Management practice. He specializes in security architecture and security strategy, and has designed security solutions for Fortune 500 clients in financial services, manufacturing, energy, and health care. Michael has a broad background in government and commercial security, including six years as a special agent with the U.S. government investigating computer and white-collar crimes.

wsjsecurity@sys-con.com

IN THE NEXT ISSUE OF WSJ...

Focus: .NET/J2EE

Evolving JMX to Support Web Services Management Technologies

Recently, XML, SOAP, and Web services have emerged as the dominant distributed interaction mechanism across heterogeneous platforms and over the Internet due to their platform neutrality, extensibility, ease of use and wide industry acceptance. The same factors also make them an excellent choice of integration technology for exchanging management information. This article looks at how to simplify JMX and increase its usefulness in a future revision, and define a JMX connector for WSDM through a separate JSR.

Web Services + the Grid = Prime Time

Security and the “ilities” are two major hurdles Web services must address before they can be considered “Prime Time”. This article looks at how the recent Grid-inspired standards proposals and Grid experiences address these missing elements.

Web Services: Monitoring and Management for Reliability

Applications designed around messaging have built-in reliability, scalability, and monitoring (depending on the vendor). Web services on the other hand, have to be designed for reliability, scalability and monitoring as the protocol intrinisically does not support these features. These are extremely important considerations that need to be evaluated in the architecture stage to prevent the solution itself from becoming a problem.

Plus...

Web Services in Action: Integrating with the eBay Marketplace

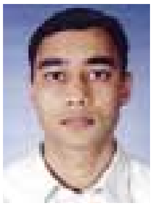
Web services have served as a topic of interest for IT professionals since at least 2001, and yet there have been few examples of significant, successful deployment. This article looks at how eBay has used Web services to dramatically grow both acquisition and third party development efforts, transforming a Web site into a Web platform. It will examine the business benefits of Web services for eBay, its partners, and independent developers, and describe their efforts to support this growing community.

Web Service Local Reference

Solving significant concerns

There is a need for container-managed support for local invocations among colocated Web services. This feature would be similar to EJB local invocations in the J2EE world.

WRITTEN BY



PANKAJ KOTHARI &

VENKAT RAGUNATHAN

This need stems from the fact that a lot of enterprise applications are being Web service enabled and wrapped for the twin purposes of easier EAI and external access across firewalls in a standards-based way. Given the increasingly complex nature of business transactions, often these Web services that need to invoke each other due to functional dependencies. Also, some facade Web services need to invoke back-end applications, which themselves are Web services that in turn wrap other legacy applications running on different platforms. In these scenarios, and many others, it would be an overhead to use HTTP/SOAP for invocations among Web services running within the same Web service container.

There needs to be a way for the container to identify potential efficiencies in such scenarios and make use of them. One such way is to use local references for calls among colocated Web services. Also, optionally, Web service clients can be granted an ability to designate some invocations as "local" Web service invocations.

In this article we look at possible scenarios that emphasize the need for such features. Further, we also identify various architectural possibilities for how this can be achieved.

Scenarios

There are various circumstances in which local references to Web services would be useful:

- Functional dependencies may exist across colocated Web services. For example, a location-based Web service deployed in an MNO (Mobile Network Operator) environment like "local map service" needs location information, which could be provided by a location Web service. Since this location service is possibly in the same Web service container, it would make sense to invoke the location service as a language method call rather than through HTTP/SOAP.
- Third-party Web services may be needed to wrap around and provide new standards-based interfaces and in turn expose them as new Web services. This can happen, for example, in the case of third-party telecom Web services obeying an MM7 interface to be exposed as Parlay-X Web services. In this scenario, the wrapper Web service can call the underlying Web service more efficiently than by the traditional way of parameter marshalling/ unmarshalling and network invocation.
- There may be a need to compose existing Web services into more complex and higher value-added Web services, which are branched sequences of simpler Web service invocations. For example, a content portal can compose a profile Web service and a news Web service to give a user a personalized news Web service. This composed service is colocated with the constituent simple Web services and can interact with them using a "local" invocation.
- There may be a need to provide a facade Web service to multiple other Web services as a common invocation point or common interface. This could serve the dual purpose of simplifying the interfaces to the end user as well as abstracting them from actual service locations. This Web service pattern can have a runtime advantage because the Web services encapsulating the actual business logic are located in the same container.
- Non-Web service components may exist that invoke Web services within the same container; for example, a servlet or EJB invoking a Web service. In such cases, local invocations can enhance the performance of the applications. Ideally, this must be done transparently by the Web services container.

In these and many similar scenarios, more efficient invocations to Web services would be possible than those supported by the current implementations based on HTTP/SOAP. The Web service containers need to exploit these possibilities and optimize these calls.

> “Performance and scalability of server components are significant concerns in the Web services implementations”

Container Architectural Possibilities

Presented here are some possibilities that container providers can explore to achieve local invocations among Web services. The Web service containers can provide Web service client libraries (e.g., JAX-RPC factories) that can internally optimize “local” Web service access. This can be done by first identifying the fact that the invocation is local. This identification can be done in many ways:

1. By the client application marking the invocation local, or
2. By the container intelligently deducing this.

In the J2EE world, for example, JAX-RPC client libraries are used for Web service invocations. The “Call” object is configured with invocation parameters and in this case is used as the client-side proxy. By having the Call class determine through the invoked endpoint that the invocation is local, the implementation can intelligently and transparently optimize the invocation.

It should be possible to achieve similar efficiencies in the .NET world.

Using the above mechanisms, the container is able to identify the invocation as local. Further, this information should be used to optimize the invocation itself.

The optimizations in the invocations are container dependent, although there are, again, numerous potential approaches. Containers can choose to convert SOAP/HTTP calls to direct object invocations. This will obviate the overheads in the above protocols and the network latencies.

On another level, the protocols can be optimized to gain execution time economies. Bypassing serialization and deserialization of parameters is another option to achieve similar goals of performance and scalability. One more means of optimization could be to avoid type-mapping lookups for the colocated calls. Another way is to bypass security alone – in case authentication and authorization are already performed in the calling Web service.

Remember that one objective is that existing clients should be unaware of migration. The changes to client libraries must not render existing deployments incompatible.

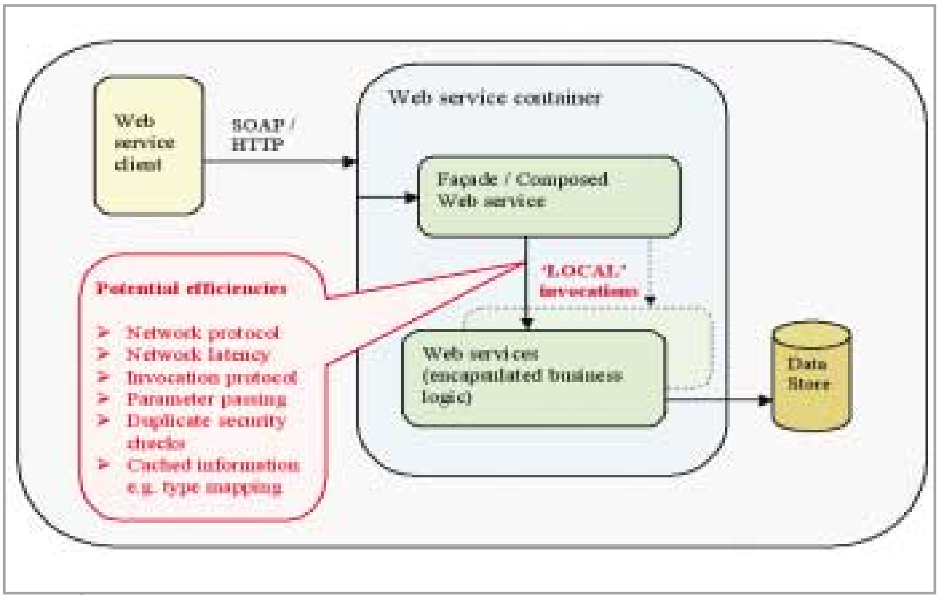


FIGURE 1 Web Service Local Reference invocation efficiencies

Heterogeneous Invocations

Since the concept of local invocations is an intracontainer affair, the optimizations can be completely abstracted by container implementations and can be made transparent to the invoking client (see Figure 1).

An added advantage from the above is that the existing clients can enjoy the added performance and scalability without having to undergo any change themselves. This means that heterogeneous invocations (e.g., .NET client to J2EE Web service) will continue to be as much of a possibility as they are today.

Conclusion

Performance and scalability of server components are significant concerns in the Web services implementations. The proposal here aims to mitigate these concerns and help optimize application execution in turn reducing underlying hardware costs. It exploits a priori information available to the container with respect to Web service deployment. This information is utilized for achieving better performance and scalability for invocations between colocated Web services.

About the Authors

Pankaj Kothari is a technology consultant at Hewlett-Packard. He has seven years of experience in the areas of Web services, J2EE, and workflow, and has been involved in various product development efforts at HP.

■■■pankaj.kothari@hp.com

Venkat Ragunathan is a software architect at Hewlett-Packard. He has 14 years of experience in the areas of Web services, mobility, and Internet applications and has been involved in various product and solution development efforts at HP.

■■■venkat.ragunathan@hp.com

Federated Identity Standards

Confused? You bet you are

■ Business is becoming increasingly virtual and decentralized, while real-time management of relationships with employees, contractors, partners, suppliers, and customers is becoming ever more crucial. Even within a single company, applications may reside on different platforms, in separate departmental security domains, in legacy databases derived from prior acquisitions, or (thanks to outsourcing) in separate companies. As gaining access to distributed resources becomes increasingly vital, the ability to manage identity effectively becomes a paramount concern.

WRITTEN BY
ERIC NORLIN
&
DARREN PLATT

Federated Identity

"Federated Identity" is a set of mechanisms through which companies can share identity information between security domains. As a result of federation, companies are now able to create identity-based applications (such as federated single sign-on) that enable increased access to cross-boundary information.

Federated identity delivers several compelling benefits to organizations. Federation means that local identities and their associated data stay in place, but they are linked together through higher-level mechanisms. In addition, federated identity organizes controlled linkages among the distributed identities of a user. This allows for efficient management, control, and movement in a radically distributed world. As organizations integrate more tightly with trading partners and outsourcers, federated identity provides a flexible mechanism that authenticates users from partner organizations and provides them with seamless access to protected online resources.

But federated identity and the standards surrounding it can be very confusing. From Liberty to WS-* to SAML and sea to shining sea, federation has become a bit of a tangle. This article will sort through some of the acronym jungle.

The Beginning of Federated Identity Standards

The move for federated identity standards began because customers of enterprise identity management companies started to ask how they could obtain single sign-on to and from their partner organizations. The fact that vendors could not ensure their own systems would achieve ubiquity of deployment led vendors to quickly realize that the ultimate solution to this problem was not one-off integration, but rather a standard mechanism for single sign-on.

The quest for developing one standard began with a company named Securant. (Securant and their identity management platform, "ClearTrust," was later sold to RSA Security). Securant worked for several months – with a few dozen of its customers, partners, and other vendors – to create a standard called "AuthXML." A few days after AuthXML was publicly announced, Netegrity and VeriSign announced their own standardization effort for the same problem – "S2ML." Through the encouragement of customers and analysts, it was decided that it was best for all involved if the efforts were combined.

During this time, a couple of meetings were held with representatives of all of the leading Web access control vendors. Ultimately, it was decided to merge the two standards efforts at the OASIS standards organization. All of the members of that group joined OASIS to work on the new standard, which took its name from the two standards it was based on, and was named SAML (the "Security Assertion Markup Language"). Their work resulted in the SAML specification, released on January 9, 2001.

From SAML to Liberty to WS Federation

The Liberty Alliance, a consortium of technology vendors and end-user companies, was formed to provide an open standard for federated identity. Liberty's work sought to build upon SAML to provide, in essence, an extension that more tightly defined the broad-based work that SAML had started. In so doing, Liberty also incorporated the WS-Security specification that Microsoft, IBM (and others) had submitted to the OASIS body. And, of course, the story doesn't end there.

Microsoft, IBM, VeriSign, and others united to write a broad set of specifications under the "WS" label. This body of work is intended to be a more modular architecture than other federated identity specifications – primarily because the WS work is meant to include other Web services needs. Accordingly, one can find WS-Security, Policy, Trust, Secure Conversation, and WS-Federation – where federation is one member of a larger family.

This tangle of standards and dependencies leads us to a brief explanation of the federated identity standards universe.

The Universe of Federated Identity

The universe of federated identity has come to encompass much more than simply single sign-on. Groups like the Liberty Alliance and the vendors focused on the WS-Federation specifications have begun to work on attribute exchange and associated services.

Figure 1 outlines the standards and dependencies that exist today. A brief explanation of each follows.

SAML 1.0, 1.1, and 2.0

SAML pronounced "Sam –el" is an extensible language for securely exchanging user information between security domains. SAML 1.0 defines a security token format (called an assertion), as well as "profiles" that define methods of using these assertions to provide Web single sign-on. SAML 1.1 incorporates feedback and errata from the 1.0 specification. SAML 2.0 is currently in the solution proposal phase – a state meant to address the requirements outlined in the market requirements document. SAML 2.0's primary objective is an

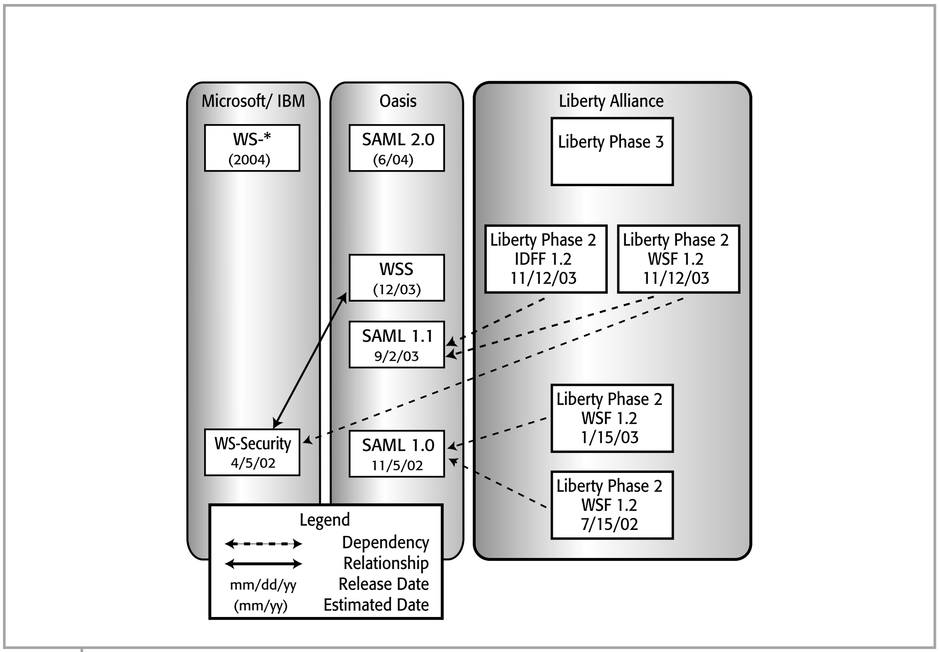


FIGURE 1 **Today's standards and dependencies (Graph courtesy of Ping Identity Corporation's technology brief, "Federation Standards Overview")**

incorporation of Liberty ID-FF 1.2, which I will discuss later.

Liberty Phase 1 (ID-FF 1.0)

Liberty Phase 1 extends SAML 1.0 by adding its own profiles for how to wield SAML assertions. These additional profiles add support for account linking, identity provider introduction, and global logout. The Liberty Alliance model defines roles within a federation – an Identity Provider (IdP) and a Service Provider (SP). Liberty ID-FF1.1 incorporates feedback and errata from the 1.0 specification.

Liberty Phase 2 (ID-FF 1.2)

This set of standards extends ID-FF with new functionality, such as one-time assertions of identity (for anonymity), metadata exchange, and affiliate relationships. Liberty Phase 2 (ID-WSF 1.0) is a set of standards that extend the existing Liberty framework with functionality for discovering and offering identity-related services. Profile access mechanisms are specified as an initial service, allowing for access to user attributes. Liberty Phase 2 defines many of its messages and protocol bindings in terms of SAML 1.1, and uses WS-Security for securing SOAP messages.

Liberty Phase 3

This set of standards still in the elaboration stage, but it is expected that ID-WSF will be extended with new services built on top of attribute exchange, such as a digital wallet and calendaring/address book services.

WS-Security

This specification defines mechanisms for providing security token–based integrity and confidentiality on Web Service (SOAP) messages. Several security tokens are defined, as well as a mechanism for associating them with messages.

WS-Security Extensions (WS-Trust, WS-Policy, WS-Federation)

This collection of specifications is an evolving set of Web service-oriented mechanisms for layering authentication, authorization, and policy across both a single and multiple security domains. WS-Federation defines a framework for federation. Profiles will be developed subsequently to specify the details for implementation.

OASIS WSS

An Oasis technical committee focused on the finishing stages of standardizing WS-Security and several security token types.

Practically Speaking

Confused? Of course you are – that's why this article exists. To sort this out, let's deal with the universe of federation standards as they exist today. That leaves us with SAML 1.1 (which is not backward compatible with SAML 1.0), Liberty 1.1, and Liberty Phase 2.

Given all of the above, the distinguishing factor is simple:

If you are an enterprise that is seeking to link two pre-existing accounts, and you must do so before the end of Q1 2005, then your easiest path is to use the Liberty Alliance specifications.

The reason for this is simple. While SAML 1.1 will allow you to perform the same operations, it will require you to extend the profiles to do so. The Liberty Alliance – in its current form – is built to accomplish this operation. That said, all of this will change as SAML 2.0 is released (realistically it should ship in products in the Q2 2005 timeframe), as SAML 2.0 will incorporate Liberty 1.2.

The Future of Federated Identity Standards

The one question that surfaces time and time again is around convergence – will there be a convergence of the federated identity standards? Hopefully, this article has shone some light on that conundrum. SAML and Liberty are becoming increasingly intertwined. WS-Federation (and the supporting specifications) stand apart. Broadly speaking, then, federated identity sits in two camps: the SAML/Liberty camp and the WS-Federation camp. Of course, some vendors are planning to incorporate all standards.

In the meantime, the practical implementation of federated identity becomes a question of business drivers. If there is a business imperative to more tightly integrate and manage distributed systems of identity, and that business imperative demands some near-term action, then the enterprise needs to make some hard choices. The safe bet is a vendor that has stated support for all three standards (nearly all do). The implementation choices will most likely come down to the use case – and if that use case is about linking two pre-existing accounts, then the Liberty Alliance presents the easiest path. ⓔ

About the Authors

Eric Norlin is SVP of Strategic Marketing for Ping Identity Corporation (www.pingidentity.com) and contributing editor for *Digital ID World* (www.digitalidworld.com).

Darren Platt is director of Solutions Architecture for Ping Identity Corporation (www.pingidentity.com). His previous positions include VP of Engineering for Securant and founding member of the SAML technical committee.

A LIMITED TIME SAVINGS OFFER FROM *SYS-CON MEDIA*

SUBSCRIBE TODAY TO MULTIPLE MAGAZINES

AND SAVE UP TO $340 AND RECEIVE UP TO 3 FREE CDs!*

RECEIVE YOUR DIGITAL EDITION ACCESS CODE INSTANTLY WITH YOUR PAID SUBSCRIPTIONS

CALL TODAY! 888-303-5282

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

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

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

□ LinuxWorld Magazine

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

□ JDJ

U.S. - Two Years (24) Cover: $144	You Pay: $99.99 /	Save: $45 + FREE $198 CD
U.S. - One Year (12) Cover: $72	You Pay: $69.99 /	Save: $12
Can/Mex - Two Years (24) $168	You Pay: $119.99 /	Save: $48 + FREE $198 CD
Can/Mex - One Year (12) $120	You Pay: $89.99 /	Save: $40
Int'l - Two Years (24) $216	You Pay: $176 /	Save: $40 + FREE $198 CD
Int'l - One Year (12) $108	You Pay: $99.99 /	Save: $8

□ Web Services Journal

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

□ .NET Developer's Journal

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

□ Information Storage + Security Journal

U.S. - Two Years (24) Cover: $143	You Pay: $49.99 /	Save: $93 + FREE $198 CD
U.S. - One Year (12) Cover: $72	You Pay: $39.99 /	Save: $39
Can/Mex - Two Years (24) $168	You Pay: $79.99 /	Save: $88 + FREE $198 CD
Can/Mex - One Year (12) $84	You Pay: $49.99 /	Save: $34
Int'l - Two Years (24) $216	You Pay: $89.99 /	Save: $126 + FREE $198 CD
Int'l - One Year (12) $108	You Pay: $59.99 /	Save: $48

□ Wireless Business & Technology

U.S. - Two Years (12) Cover: $120	You Pay: $49.00 /	Save: $71 + FREE $198 CD
U.S. - One Year (6) Cover: $60	You Pay: $29.99 /	Save: $30
Can/Mex - Two Years (12) $120	You Pay: $69.99 /	Save: $51 + FREE $198 CD
Can/Mex - One Year (6) $60	You Pay: $49.99 /	Save: $10
Int'l - Two Years (12) $120	You Pay: $99.99 /	Save: $20 + FREE $198 CD
Int'l - One Year (6) $72	You Pay: $69.99 /	Save: $2

□ MX Developer's Journal

U.S. - Two Years (24) Cover: $143	You Pay: $49.99 /	Save: $93 + FREE $198 CD
U.S. - One Year (12) Cover: $72	You Pay: $39.99 /	Save: $32
Can/Mex - Two Years (24) $168	You Pay: $79.99 /	Save: $88 + FREE $198 CD
Can/Mex - One Year (12) $84	You Pay: $49.99 /	Save: $34
Int'l - Two Years (24) $216	You Pay: $89.99 /	Save: $126 + FREE $198 CD
Int'l - One Year (12) $108	You Pay: $59.99 /	Save: $48

□ ColdFusion Developer's Journal

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

□ WebSphere Journal

U.S. - Two Years (24) Cover: $216	You Pay: $129.00 /	Save: $87 + FREE $198 CD
U.S. - One Year (12) Cover: $108	You Pay: $89.99 /	Save: $18
Can/Mex - Two Years (24) $240	You Pay: $159.99 /	Save: $80 + FREE $198 CD
Can/Mex - One Year (12) $120	You Pay: $99.99 /	Save: $20
Int'l - Two Years (24) $264	You Pay: $189.00 /	Save: $75
Int'l - One Year (12) $132	You Pay: $129.99 /	Save: $2

□ PowerBuilder Developer's Journal

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

□ WLDJ

U.S. - Four Years (24) Cover: $240	You Pay: $99.99 /	Save: $140 + FREE $198 CD
U.S. - Two Year (12) Cover: $120	You Pay: $49.99 /	Save: $70
Can/Mex - Four Years (24) $240	You Pay: $99.99 /	Save: $140 + FREE $198 CD
Can/Mex - Two Year (12) $120	You Pay: $69.99 /	Save: $50
Int'l - Four Years (24) $240	You Pay: $120 /	Save: $120 + FREE $198 CD
Int'l - Two Year (12) $120	You Pay: $79.99 /	Save: $40

*WHILE SUPPILES LAST. OFFER SUBJECT TO CHANGE WITHOUT NOTICE

3-Pack

Pick any 3 of our magazines and save up to $210⁰⁰

Pay only $99 for a 1 year subscription plus a **FREE CD**

- 2 Year – $179.00
- Canada/Mexico – $189.00
- International – $199.00

6-Pack

Pick any 6 of our magazines and save up to $340⁰⁰

Pay only $199 for a 1 year subscription plus **2 FREE CDs**

- 2 Year – $379.00
- Canada/Mexico – $399.00
- International – $449.00

9-Pack

Pick 9 of our magazines and save up to $270⁰⁰

Pay only $399 for a 1 year subscription plus **3 FREE CDs**

- 2 Year – $699.00
- Canada/Mexico – $749.00
- International – $849.00

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

SYS-CON MEDIA

New Opportunities for Web Services Technology

New laws create new needs

Mention the word "compliance" and it is likely to conjure up images of scandalous performance by companies such as Worldcom, Enron, and Tyco. But beyond corporate governance and government regulations such as Sarbanes-Oxley, HIPAA, and the National Do Not Call Registry, compliance is creating a new need for technology in less obvious areas.

WRITTEN BY
HAL STEGER

Perhaps the largest of these relates to the rise of outsourcing, whereby companies are moving non-core functions to outside vendors. Along with the rise of outsourcing, there is an attendant increase in the use of service-level agreements (SLAs). An SLA is a contract between a provider and recipient to deliver one or more services according to an agreed upon set of performance standards. It contains a description of the service or deliverable to be provided; it sets performance expectations in terms of cost, volume of work, responsiveness, and quality; and it defines metrics for evaluating whether or not the performance requirements have been met.

As more companies outsource their IT infrastructure and business functions, they rely increasingly on contractual obligations and SLAs to ensure their needs are met and they are getting their money's worth. A June 2004 survey of 320 IT professionals conducted by Oblicore found that outsourcing has become important to 76% of companies. About half of the companies had 10 or more SLAs, 28% had more than 50, and 7% had more than 1,000. Forty-two percent of companies reported they had more SLAs than a year ago, while 56% predicted more SLAs in the year ahead. Interestingly, 64% of respondents said their SLAs had major or moderate financial consequences for not reaching SLA targets. Perhaps most importantly, 75% of companies said that it was important to improve SLA management, which is an important type of compliance.

The survey also found that 49% of companies have a mix of internal, customer, and supplier SLAs. This shows that many companies now participate in a "service chain," whereby the performance of suppliers can directly affect a company's ability to satisfy its own customers. This was most apparent in industries with the word "service" in their name, such as financial services, telecommunication services, and healthcare services. While it is clear that SLAs are on the rise and are becoming more important and difficult to monitor, 43% of companies do not report on contracts at all, while another 16% only report quarterly or even less frequently. At the other end of the spectrum, in terms of "best practices," 13% of companies reported on contracts in real time, 11% did so daily, and 21% weekly. Companies indicated that the primary benefits of more frequent SLA monitoring and management were increased customer satisfaction, improved operational efficiency, and increased performance visibility.

Balanced against the increased importance of outsourcing and the general lack of reporting are numerous industry studies that show that as many as 75% of major outsourcing projects fail to "comply" with their original objectives. What's wrong with this picture?

Companies are finding that compliance is not easy or cheap. Business "regulations," often in the form of SLAs and other legal agreements, are intended to help companies specify, monitor, and measure internal performance as well as their relationships with customers and suppliers. Government regulations place their own compliance demands on companies. Yet compliance monitoring and reporting is hampered by the fact that many large companies are geographically and functionally diverse, and the trends toward outsourcing and service chains make compliance even more challenging.

What are the implications of this for technology and in particular for Web services, and what new opportunities are they creating? Consider the example of a health insurance provider and its relationship with external entities such as customers, doctors, hospitals, etc. HIPAA requires that the provider implement safeguards to protect against the misuse of individually identifiable health information. At the same time, the insurance provider may have signed IT outsourcing agreements with one or more vendors to manage and run its back office operations. So how does the insurance provider proactively monitor the performance of its outsourcing vendors to ensure that they are not inadvertently and illegally disclosing sensitive patient health information without the company's consent, thereby exposing the company to major legal liability? Most companies are now resorting to SLAs and active monitoring to ensure compliance.

Consider another example, from the world of financial services. Compliance is creating a need for companies to exchange different forms of performance data in a seamless and real-time manner. For instance, financial services firms are dependent on global providers of network services to provide brokerage services to customers around the globe. To gain a competitive advantage, financial services companies commit to providing high levels of service, during specific time periods, in different geographies, at low cost. To achieve this, they outsource major portions of their IT to best-of-breed network providers that offer high-quality bandwidth at low rates due to economies of scale. To facilitate this service chain, there needs to be a continuous flow of performance data between multiple parties. Financial service companies need to monitor the health of their networks and compare it to industry standard benchmarks. At the same time, they must constantly monitor the level of service that they are providing to customers, in the form of service availability, response time, transaction throughput, and call center responsiveness to customer issues. This requires gathering, aggregating, correlating, analyzing, and reporting reams of performance data from heterogeneous IT systems and business applications.

What has become apparent is that "compliance" is more than adhering to static government regulations by establishing high-level guidelines, training personnel, filling out forms, gathering quarterly signatures on financial documents, and filing paperwork that is rarely viewed. Compliance today requires the ongoing proactive monitoring, management, and reporting on a dynamic set of business commitments and standards. It is causing three separate disciplines that previously were performed independently to become intricately intertwined.

Prior to the enhanced litigiousness of our society and the related increase in the use of SLAs, it was commonplace for companies to separately manage contracts, measure financial results, and monitor IT service levels. However, with the onslaught of SLAs, and especially now that there are serious financial consequences for not meeting service targets, it is causing companies to carefully connect the dots between legal, financial, and IT performance. Failure to do so could have potentially disastrous effects to the tune of millions of dollars in assessed penalties, lost revenue, and even jail time in the event of fraud.

The challenge of compliance reporting is aggravated by performance data that is present in systems that have grown independently and that is brought together only by using manual methods. A supplier may provide a company with a spreadsheet via e-mail. This approach is error prone and does not lend itself to providing an adequate picture of compliance for a business. More often than not manual performance data is late and out of context, providing little value in the effort to satisfy compliance requirements. Lack of accurate and timely data due to manual collection and transfer processes is the bane of chief compliance officers everywhere.

The characteristics of today's systems, where silos of information do not readily communicate with each other, lead to a set of problems that make compliance very difficult to implement and manage, except at a most rudimentary level. These characteristics include:

- Inconsistent data management policies across systems
- Inconsistent data formats across systems
- Non-integrated systems that do not share information

- Poor data reconciliation across systems for compliance

The result is a new set of compliance requirements that is spurring the need for technology innovation:

- ***Capture all aspects of regulations, including legal, financial and technical***. Most regulations, whether internal or external, contain a combination of legal, financial, and technical terms. This invites the need for a Web service for updating and exchanging contractual and regulatory data between multiple parties on an ongoing basis.

- ***Combine data from disparate systems and manage a wide range of processes***. As noted, required compliance data exists in multiple systems that are geographically and functionally disperse. Compliance applications need to take required data from systems, transform it to a common denominator, correlate it to regulations, and generate reports, all in real time. For example, Section 409 of Sarbanes-Oxley requires real-time reporting on any event that materially impacts the financial health of the company. A bad debt should not only be captured by the compliance system, it should also be reported in real time to concerned executives. Complying with section 409 would require a compliance application to access bad debt data from the financial system, generate reports, and access the e-mail system to send the reports. Accessing and integrating these disparate systems and data is a call to arms for Web services.

- ***Provide relevant reports to regulatory bodies***. Different constituencies of a compliance system require different types of reports. If multiple external or internal customers are served with a compliance report, several report flavors may be required at the same time. Not only are the report contents different, they might also be viewed on a different frequency. The CFO might want a management dashboard that updates every time a compliance violation occurs, whereas an account receivables manager might look at a daily report and a monthly summary report on outstanding debts. A noncompliance indicator on an online report might prompt the user to drill down to the source of the problem. Reports must not only be generated on the fly, but also offer the ability to be highly flexible, creating a need for Web services that can provide this type of data access.

> “The result is a new set of compliance requirements that is spurring the need for technology innovation”

WSJ ADVERTISER INDEX

ADVERTISER	URL	PHONE	PAGE
Active-Endpoints	www.activeendpoints.com		15
Gartner	www.gartner.com/us/aiws	203-316-6603	27
Gilbane Conference	www.lighthouseseminars.com	781-821-6734	23
IBM	www.ibm.com/middleware/process		Cover IV
Information Storage+Security	www.ISSJournal.com	888-303-5282	57
IT Solutions Guide	www.sys-con.com/itsolutions	201-802-3021	41
Mindreef	www.mindreef.com		9, 11
OpenLink Software	www.openlinksw.com/virtuoso/	800-495-6322	Cover II
Parasoft	www.parasoft.com/AchieveQuality	888-305-0041	6
Prolifics	www.prolifics.com	800-675-5417	5
Secrets of the XML Masters	www.sys-con.com/freecd	888-303-5282	43
Service Integrity	www.serviceintegrity.com		19
StrikeIron	www.strikeiron.com	919-405-7010	3
SYS-CON e-newsletters	www.sys-con.com	888-303-5282	51
SYS-CON Publications	www.sys-con.com/2001/sub.cfm	888-303-5282	39
Web Services Edge East	www.sys-con.com/edge	201-802-3066	37
Web Services Journal	www.wsj2.com	888-303-5252	31
WebAppCabaret	www.webappcabaret.com/wsj.jsp	866-256-7973	Cover III

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. This index is provided as an additional service to our readers. The publisher does not assume any liability for errors or omissions.

- ***Monitor compliance continuously and reveal underlying causes of noncompliance***. A compliance system should be “always on,” recording transactions that affect compliance as they happen. Reporting after the fact will not meet business and government compliance requirements. Furthermore, a compliance system should be able to drill down to the root cause of non-compliance and allow for “what-if” analysis of the compliance event. This elevates the compliance system from a static reporting tool to a proactive business monitoring application. Web services that provide real-time event notification and access to relevant compliance data are needed.

In summary, compliance requires a diverse set of performance data to be evaluated in the context of contractual obligations. If ever there was an opportunity that requires the need to easily and securely exchange data between multiple companies as well as ease the interoperability of disparate and heterogeneous applications and data, compliance is a major driver that will encourage the development of new Web services.

About the Author

Hal Steger has worked in high tech for 22 years. He is senior vice president of worldwide marketing at Oblicore Inc., a leading provider of enterprise applications that help companies monitor, manage, and report on SLAs and outsourcing contracts. He was cofounder and vice president of marketing of Rubric, which pioneered the category of enterprise marketing automation applications. Hal holds a bachelor’s of science degree in computer science and economics from the University of Michigan, and an M.S.I.A. (MBA) from Carnegie-Mellon. He has spoken at many prominent industry conferences and written numerous articles for leading business and IT trade publications.

FREE*CD! ($198.00 VALUE!)

Secrets of the XML Masters

Every ***XML-J*** Article on One CD!

XML JOURNAL
— PRESENTS —
$198 VALUE
XML
Resource CD
Secrets of the XML Masters
Inside:
Every Issue of XML-J Ever Published on One CD!

— The Complete Works —

CD is organized into 33 chapters containing more than 450 exclusive ***XML-J*** articles!

All in an easy-to-navigate HTML format! **BONUS: Full source code included!**

ORDER AT WWW.SYS-CON.COM/FREECD

**PLUS $9.95 SHIPPING AND PROCESSING (U.S. ONLY)*

SYS-CON MEDIA

Only from the World's Leading i-Technology Publisher

Stretching UDDI

Develop your own Java apps to consume Web services

■ UDDI (Universal Description, Discovery, and Integration) is fast becoming a standard for storing business processes available on the Web. Although UDDI is capable of storing many different types of data, for the purposes of this article I'll focus on how UDDI can be used to register Web services, thereby making them available for application-level consumption.

WRITTEN BY
ANDREW BRADFIELD

What Is Covered?

This article provides Java developers with a fast and simple way to develop their own UDDI Java applications to consume Web services registered within a UDDI Registry. The sample code (the source code is online at www.sys-con.com/webservices/sourcec. cfm) contains a payroll Web service (deployed from an entity bean) and a UDDI session bean.

Background

This article is based heavily on the tutorial Discover Web services with the WSDK V5.1: UDDI cited in the Resources section. I will present an extension to the basic UDDI4J classes so you can quickly set up your own UDDI registry of Web services. While working through the Discover Web services with the WSDK V5.1: UDDI tutorial, I developed an extension API that provides a higher-level approach to the UDDI4J API. This article extends only a small subset of that functionality, since its purpose is to register, discover, and consume a Web service stored within a UDDI Registry. For a broader discussion of UDDI and its uses beyond storing Web services, please see *Understanding UDDI* by Tom Bellwood.

Payroll Web Service

The Payroll.ear file contains an EJB project named PayrollEjbPrj. Within this EJB project are two entity beans, Department and Employee, and one session bean, Payroll. The Department and Employee entity beans expose getter and setter methods for the sample database. The Payroll bean, which exposes 12 methods, uses the getter methods of the entity beans to query various pieces of information from the sample database.

The sample code can easily be extended to provide setter functionality from within the Payroll bean. Exposing all of the sample database fields is beyond the scope of this article.

UDDIClient Session EJB

The UDDIClient session bean makes almost exclusive use of the UDDI utility API discussed in the next section.

A UDDI Registry may contain Businesses, Services, and TModels. The UDDIClient bean provides a publish() and two delete() methods for each type of UDDI entry. As shown in the method signatures in Listing 1, these methods are as simple as possible, while shifting the work to the UDDI Utility API.

Configuration Assumptions

You are using IBM WebSphere Studio Application Developer 5.0 or another J2EE-compliant development environment such as the Eclipse IDE.

A DB2 application has been installed and a sample database has been created. (This sample uses the Employee and Department tables. To create the sample database, Open DB2 First Steps and click on the "Create Sample Database" link).

A UDDI Registry is installed and properly configured. Please see the InstallUDDI listing for an automated setup of a UDDI Registry to use the Cloudscape Database. Execute installuddilibs.bat from a command window to create the necessary directory structure.

Methods getService and executeService

The getService and executeService methods of the provided UDDIClient bean demonstrate how to retrieve our Payroll Web service and invoke some of its methods. The executeService method takes a string representing the name of a service stored in the registry. In the sample code, the service registered is the Payroll Web service. For this article, I've hard coded two methods that are called from the Payroll Web service found within the registry (see Listing 2).

UDDI Utility API

The UDDI Utility API has four main classes, three of which are shown in Listings 3–5. The fourth, Utilities.class, contains all of the setup information for the UDDI Registry. The values and structure are explained in detail in the Publishing your services on UDDI with the WSDK V5.1 tutorial (see Listing 3).

Summary

As you have seen, UDDI programming can be a very powerful way to share information on the Web via the UDDI Registry. This sample code, and more specifically the UDDI Utility API, are provided as a starting point from which you can develop a more customized solution.

Resources

Code

- ***UDDI Utilities API Javadoc:*** In the UDDI Proxy.zip's doc directory. There is only one index.html located in the doc directory. Use it to reference all of the Javadoc for the entire article.
- ***UDDI Utilities API source:*** In the UDDI Proxy.zip's Utility API Source directory
- ***Sample code:*** In UDDI Proxy.zip there are two .ear files, UddiClientEAR.ear and PayrollEAR.ear. Both are needed.
- ***UDDI Registry installer In the UDDI Proxy.zip's InstallUDDI directory:*** There are a number of libs and a batch file named installuddilibs.bat. The bat file installs the UDDI registry to use Cloudscape.

Specifications

- ***SOAP 1.1 Specification:*** www.w3.org/TR/soap
- ***UDDI Specification:*** www.oasis-open.org/committees/tc_home.php?wg_abbrev=uddi-spec
- ***WS-I Basic Profile:*** www.ws-i.org/Profiles/

BasicProfile-1.0-2004-04-16.html

- ***WSDL 1.1 Specification:*** www.w3.org/TR/wsdl.html

Tools

- ***IBM WebSphere Studio Application Developer:*** www-306.ibm.com/software/ awdtools/studioappdev
- ***IBM WSDL4J Project on developerWorks:*** www-124.ibm.com/developerworks/projects/wsdl4j/
- ***Eclipse:*** www.eclipse.org

Other Resources

- *Publishing your services on UDDI with the WSDK V5.1:* www-106.ibm.com/developerworks/edu/ws-dw-ws-psuddi51-i.html?S_TACT=104AHW04&S_CMP=EDU
- *Discover Web services with the WSDK V5.1: UDDI:* www-106.ibm.com/developerworks/webservices/edu/ws-dw-ws-dwsuddi51-i.html
- *Understanding UDDI:* www-106.ibm.com/ developerworks/webservices/library/ws-featuddi
- *UDDI4J Web site:* www-124.ibm.com/ developerworks/oss/uddi4j/
- *UDDI home page:* www.uddi.org

About the Author

Andrew J. Bradfield is an advisory software engineer working in IBM's Software Division at the IBM T.J. Watson Research Center in Hawthorne, New York. His projects include J2EE development ofcontent management and data management solutions. Andrew graduated with a degree in computer science from Hamilton College in Clinton, New York.

■■■abrad@us.ibm.com.

Listing 1

```
public String publishBusiness(String busName)
{return BusinessUtilities.publishBusiness(busName);}

public void deleteBusinessByName(String busName)
{BusinessUtilities.deleteBusinessByName(busName);}

public void deleteBusinessByKey(String busKey)
{BusinessUtilities.deleteBusinessByKey(busKey);}

public String publishService(String serviceName, String
busName, String tModelName, String tModelOverviewDoc, String
accessPointStr, String description)
{return
ServiceUtilities.publishService(serviceName,busName,tModelName,
tModelOverviewDoc,accessPointStr,description);}

public void deleteServiceByName(String serviceName)
{ServiceUtilities.deleteServiceByName(serviceName);}

public void deleteServiceByKey(String serviceKey)
{ServiceUtilities.deleteServiceByKey(serviceKey);}

public String publishTModel(String tModelName, String
overviewDocString)
{return
ModelUtilities.publishTModel(tModelName,overviewDocString);}

public void deleteTModelByName(String tModelName)
{ModelUtilities.deleteTModelByName(tModelName);}

public void deleteTModelByKey(String tModelKey)
{ModelUtilities.deleteTModelByKey(tModelKey);}
```

Listing 2

```
/**Execute a Service found in the UDDI Registry
public void executeService(String serviceName)
{//Obtain all access points for services providing this
  service
:
:
:
//For each access point, create an PayrollService object
String accessPoint = (String)it.next();
payroll = getService(accessPoint);

if(payroll != null)
{
   System.out.println("Employee 000010's bonus is " +
     payroll.getBonus("000010"));
   System.out.println("Employee 000010's phone number is " +
     payroll.getPhoneNo("000010"));
}
:
:
:
}

/**Create a Service object, to communicate with the Web
     service defined by the URL
private Payroll getService(String urlString)
{
:
:
:
   URL url = new URL(urlString);
   System.out.println("Payroll URL: " + url.toString());
   System.out.println("Getting Payroll object");
   payroll = payrollServiceLocator.getPayroll(url);
   System.out.println("Payroll object retrieved successful-
ly");
:
:
:
   return payroll;
}
```

Listing 3: BusinessUtilities

```
/**Display business information of a BusinessEntity
public static void showBusinessDetail(String businessKey)

/**Locate a Business by name
public static Vector findBusinessByName(String busName)

/**Locate a Business by key
public static BusinessList findBusinessByKey(String
businessKey)

/**Delete a Business by name
public static void deleteBusinessByName(String busName)

/**Delete a Business by key
public static void deleteBusinessByKey(String businessKey)

/**Publish a Business
public static String publishBusiness(String busName)
```

Listing 4: ModelUtilities

```
/**Locate a technical Model by name
public static Vector findTModelByName(String tModelName)

/**Locate a technical Model by key
public static TModelList findTModelByKey(String tModelKey)

/**Delete a technical Model by name
public static void deleteTModelByName(String tModelName)

/**Delete a technical Model by key
public static void deleteTModelByKey(String tModelKey)

/**Publish a technical Model
public static String publishTModel(String tModelName, String
overviewDocString)
```

Listing 5: ServiceUtilities

```
/**Locate a Service by name
public static Map findServiceByName(Vector serviceNames)

/**Locate a Service by key
public static ServiceList findServiceByKey(String serviceKey)

/**Delete a Service by name
public static void deleteServiceByName(String serviceName)

/**Delete a Service by key
public static void deleteServiceByKey(String serviceKey)

/**Publish a Service
public static String publishService(String serviceName, String
busName, String tModelName, String tModelOverviewDoc, String
accessPointStr, String description)
```

Taxi Cabs and Railroads

A new approach to building adaptive information systems

"When change within your organization is slower than that without, you're in real trouble. We can't predict the future, but we can learn to react a lot faster than our adversaries." – Jack Welch

WRITTEN BY
JONATHAN SAPIR

If we try to build software solutions for highly paid knowledge workers who work in a rapidly changing environment in the same way that we build solutions for back office workers who work in predictable circumstances, we are doomed to failure. Freezing specifications, building software to fit those specs, and ignoring that business is constantly changing is a sure recipe for disaster.

To achieve the levels of functionality, flexibility, and time-to-market required by business today, a radical shift is required in the way in which software is developed. This major shift is already well underway, with Web services and SOAs. But the technology alone will not make any serious impact on the speed and effectiveness with which we are able to build information systems. We need a completely fresh approach to our methodology.

A New Approach

Historically, information has been delivered to desktops in much the same fashion as railroads were built in the early 1900s. Building a railroad system required multiple stages of planning, agreed-upon destinations, predetermined stops at train stations, limited switching choices, the moving of businesses closer to the stations, and rigid schedules to maximize rail efficiency rather than user demand. The very nature of the railroad system leaves little room for flexibility and adaptability. This characteristic is critically important for railroads – and certain types of business applications such as accounting and manufacturing.

But this approach, with its fixed plans, fixed rails, stations, and predetermined schedules, doesn't work when events cannot be easily anticipated and responses need to be made up on-the-fly. To continue the transportation metaphor, the need for a dynamic business environment is more closely reflected in the process that taxi cab companies use to respond to demand. In a typical U.S. city, cabs cruise the streets with only flexible strategies, allowing response to demand to unfold as required. Decisions are made as closely as possible to the time when action must be taken. The driver makes decisions on the spot – consistent with passengers needs.

In the railroad "methodology," the organization plans in advance and passengers must adjust their plans accordingly. In the taxicab approach, the organization must adjust in real time to the passenger whose plans are unknown most of the time. This requires organizations to embrace uncertainty, dynamic demand, and some degree of chaos, and to learn to thrive on it.

When users are no longer constrained by the shackles of inflexible information systems and are instead empowered by them to act as independent agents pursuing their own solutions with minimal central control, new, highly competitive, and formidable business enterprises can emerge.

Why Now?

Technology and global political, economic, and social trends are intersecting, making this new approach both necessary and possible.

- The speed of change has made existing modes of building software obsolete. In an era when change arrives without warning and threatens to eradicate entire companies and industries overnight, organizations can survive only by engaging the eyes, ears, minds, and emotions of all individuals, providing them with tools and encouraging them to act on intellectual capital and initiative.
- For the first time ever, the software industry has a usable, universally agreed upon, open standard for creating and assembling building blocks of functionality in the form of Web services. The standards and wide support for Web services make possible a significant change in the way software is developed, deployed, and maintained.
- Users are becoming more and more IT savvy and can take on more responsibility for managing their own "personal information services" in ways similar to their ability to build their own spreadsheets, desktop databases, queries, reports, etc.

The Tipping Point

We are reaching the "tipping point" for this approach. The focus of IT is still on delivering solutions by the old railroad model, but this will change – soon. The new approach will make users responsible for automating their own jobs in ways that make sense to them; they will be able to "package" their expertise and make it available as a service over the Web; and they will be able to synchronize these services with other services to achieve larger, more complex business objectives.

Sources

- Rubenstein, Moshe; and Firstenberg, Iris. (1999). *The Minding Organization: Bring the future to the present and turn creative ideas into business solutions.* John Wiley & Sons, Inc.
- Sapir, Jonathan. (2004). *Igniting the Phoenix: A new vision for IT*. Xlibris. ⓔ

About the Author

Jonathan Sapir is the president of InfoPower Systems, Inc., a Chicago-based consulting firm; the author of *Igniting the Phoenix: A New Vision for IT*; and the architect of a breakthrough personal service builder product called SnapXT, which will be available later this year. You can read his blog at www.IgnitingThePhoenix.com

■■■ jasapir@infpwr.com

XML JOURNAL
SUPPLEMENT

HOME
ENTERPRISE SOLUTIONS
CONTENT MANAGEMENT
DATA MANAGEMENT
XML LABS

This Month

XML-Based Interop, Close Up

In addition to the strategy side of Web services, there is also the protocol-oriented side of things, the XML side. Embracing not only XML itself but also the full range of mainstream XML-based technologies like XPath, XSLT, XML Schema, and SOAP, *XML-Journal* has been delivering insightful articles to the world of developers and development managers since the year 2000.

It is our privilege to bring *XML-Journal* directly to readers of *Web Services Journal*, and vice versa. Anyone already familiar with the Web services world of SOAP, UDDI, and WSDL will find here articles and features each month that will interest them – about the cutting-edge technologies and latest products that are changing not only our industry, but the way the world exchanges information. To make it easy for you to find your way around, we have four distinct sections:

Content Management:
Organization, dissemination,
and presentation of information

Data Management:
Storage, transformation, representation,
and general use of structured and unstructured data

Enterprise Solutions:
Systems and applications that manage
mission-critical functions in the enterprise

Labs:
Product reviews, book reviews, tutorials,
and standards analysis

HOME
ENTERPRISE SOLUTIONS
CONTENT MANAGEMENT
DATA MANAGEMENT
XML LABS

Federal Government XML Implementation

Seeking middle ground

WRITTEN BY
MIKE CHAMPION

Should emergency personnel and law enforcement be called to the scene of a suspected chemical warfare attack, the last thing these frontline workers will want to do is wrestle with incompatible IT systems. Therefore, the federal government is in the throes of linking databases scattered throughout the 22 agencies that now make up the Department of Homeland Security (DHS), while tying in the efforts of state and local entities. Although a cohesive, intergovernmental network of antiterrorism systems is far from finished, XML is sure to become a key element in this orchestrated campaign.

XML is not only present in tactical initiatives, such as an antiterrorism response that mandates the tight integration of far-flung systems. it is also playing an increasing role in efforts to streamline basic, day-to-day federal operations. In fact, two years ago, the Office of Management and Budget (OMB) made open XML a cornerstone of the e-Government push, a concerted effort to develop a framework for the government to deliver services to citizens and businesses.

As DHS ramps up in the face of unprecedented national security threats, the new department is also facing huge administrative computing challenges. To unify internal operations scattered among 90+ disparate information resource management systems, DHS leadership is undertaking "eMerge^2," a major business process and system overhaul effort designed to integrate DHS' financial processes from. eMerge2 will reach across budget and cost management, funds control, general ledger, accounts payable and receivable, travel, and acquisition functions. Since most enterprise integration feats of this size now hinge on XML, the technology will almost certainly factor into eMerge2.

Numerous agencies are putting e-Government goals into action through the ongoing President's Management Agenda, which entails efforts such as an IRS initiative to stop the frustrating practice of having businesses resubmit information already delivered to the federal government. The White House estimates that over six years $6 billion will be saved in using XML to facilitate electronic transmission of various corporate tax forms.

But while the federal government seems committed to harnessing the power of XML, it also seems to be taking two very distinct approaches to XML implementation. The first is a top-down approach, with government IT leaders pushing select XML standards across agencies. The second is a bottom-up approach, with individual agencies implementing XML projects at a grassroots level. Are these two approaches compatible?

Cross-Agency Standards Coming Down

The federal top-down approach to common XML standards has high-level government CIOs embarking on a concerted campaign to get all federal agencies to use a core set of XML standards through an effort spearheaded by the federal CIO Council (refer to the Data Reference Model [DRM] of the Federal Enterprise Architecture [FEA] www.cio.gov/archive/SRM_TRM_for_Agency_Review_jan_29_03.pdf). Tying all federal XML applications to select XML standards could help address a variety of vexing issues, from shoring up military data exchange and communication between federal repositories to providing tighter integration between databases scattered throughout all levels of government.

Developing cross-agency XML standards is very challenging; technical issues still exist, but more troublesome can be the political, bureaucratic, or cultural issues that crop up. Organizations differ in terms of their legacy systems and processes, missions, and perspectives on exactly what information is strategically critical. These factors easily lead to clashes when it comes time to integrate.

Consider, for instance, the often-exhausting exercise of developing schemas used to tag data from disparate organizations working together. For these schemas to be meaningful and for data to be exchanged effectively across organizational boundaries, information categories must be rigorously analyzed. Furthermore, XML data that is syntactically valid but semantically meaningless must be corrected – a laborious task involving reconciling disparate terminology used in different agencies.

Basic turf issues could then compound this seemingly relentless process and getting buy-in from entrenched personnel on common labels could prove next to impossible. Then, to make matters worse, once semantic differences are resolved, the end result may well be the creation of more abstract terms that leave ordinary people scratching their heads over exactly how to apply these confusing new terms to the concrete details of their jobs.

To offload some of this burden from individual agencies, the CIO Council is now working on the development of standard schemas and other labor-intensive chores surrounding XML standardization. Along with providing uniformity, the CIO Council also wants to free up individual program managers eager to pursue XML applications.

Grassroots XML Springing Up

Concurrent with the CIO Council's push for cross-agency XML standards, a bottom-up XML push from the grassroots is now underway throughout agencies. Military and intelligence communities, as well as civilian agencies, are at varying stages of implementing a number of IT integration projects that incorporate XML technology in a significant way. These projects build upon a fundamental strength of XML: it unifies diverse platforms, applications and communities without displacing infrastructures and processes familiar to participating users.

The leaders of these projects have decided not to hold back on XML-related forays until there is a shortlist of acceptable cross-agency standards, schemas, and repositories with the CIO Council's seal of approval. They see the need as too urgent and the opportunity as too significant (especially in a color-coded threat environment) to let the bureaucratic process run its winding course.

One department taking the lead in this regard is the US Navy – widely recognized as a pioneer in federal XML adoption. One Navy program, the Interactive Electronic Technical Manuals (IETMs) application, uses XML to deliver a standard set of manuals to all Navy vessels and may be used to serve material to Coast Guard cutters as well.

In devising the application, the Navy asked manufacturers supplying information that will make up the manuals to ship that content in XML format using their own schemas. The understanding was that the Navy would then receive that content and build it into manuals as a separate step. Using XML, the Navy was able to import and export the manufacturers' data without having to understand the content in detail – a basic XML process known as information aggregation.

The Navy also reportedly wants the capability to let individual ships share information on their own best practices by swapping data held in varying formats. This process requires moving the IETMs application toward a more advanced process called semantic integration – the ability to manipulate content, not just data structure. To do this, the Navy is using a number of open standards, including Web Ontology Language (OWL; see sidebar). Having enjoyed heavy investment from the World Wide Web Consortium (W3C), OWL is an evolving specification able to transform informal business conventions into rigorous ontologies.

Striking a Realistic Balance

IETMs are just one XML application well under way within the Navy. All of the Navy's independent XML ventures adhere to a central department-wide mandate that promotes interoperability and the use of select XML standards.

Specifically, the Navy has declared off-limits the use of any proprietary XML extensions, heeding a warning from the Government Accountability Office (GAO), which in 2002 advised federal agencies to steer clear of vendor add-ons aimed at capturing market share. In late July 2004, the Navy drove home that point when it issued a groundbreaking XML developer's guide.

The Navy's approach to XML presents a strong argument that it is possible to strike a realistic balance between promoting innovative, bottom-up implementation and establishing firm, top-down guidance. As evidence that the entire federal government may soon follow the Navy's lead – at least in the realm of prohibiting proprietary XML extensions – the CIO Council is reportedly discussing the possibility of penalties for agencies that use XML standards that are not entirely open.

Another example of giving the immediate green light on independent applications while insisting on strict adherence to existing industry standards, comes from the business community. Instead of trying to mandate uniform internal applications and data formats, industry associations and self-regulatory bodies have urged financial institutions to adopt the eXtensible Business Reporting Language (XBRL) standard, which describes a way to use XBRL tags to exchange financial statements.

The finance industry's wide use of XBRL is, in large part, an effort to avoid heavy-handed efforts on the part of government regulators to impose XML-related specifications. So far, that strategy has been overwhelmingly successful. In fact, government regulators such as the Securities and Exchange Commission (SEC) and the Federal Deposit Insurance Corporation (FDIC) seem loath to prescribe government standards and instead prefer to encourage industry standards.

A Technology for the Times

If XML is to make major improvements in the federal government – transforming the very relationship between government and U.S. businesses or transcending agency boundaries in the War on Terror – federal leaders will have to fully support both a top-down dedication to develop XML standards and a pragmatic bottom-up effort to build applications.

There is no need to wait for cross-agency authoritative data standards to apply powerful XML technologies – such as XPath, XSLT, and XQuery – to locate information in a message stream or database and seamlessly combine and swap this information among disparate sources and enterprise applications.

After all, XML's main attribute is that it is extensible, or able to break down traditional barriers to efficient data interchange without standardization of formats and vocabularies. If there was ever a time when the federal government needed such a capability, that time is now.

OWL

"Ontology" is a term that is becoming widely used to define a formal language that:

- Defines common words and concepts for a specific area of knowledge
- Allows applications and databases to share the meaning of information within that area of knowledge

The Web Ontology Language (OWL) is an XML-based language, defined by the W3C, that can formally describe the meaning of terminology used in a variety of documents. It defines vocabulary by which to describe properties and classes of meaningful concepts, as well as relationships among them. OWL should facilitate greater machine interpretability of Web content than that supported by XML alone. For more details, see www.w3.org/TR/owl-guide/

Author Bio

Michael Champion is a senior technologist at Software AG, Inc., working in the company's Enterprise Architects group. He has had extensive involvement with the World Wide Web Consortium (W3C), including cochairing the Web Services Architecture Working Group. His participation on the W3C's Document Object Model (DOM) Working Group included work as an editor of the core XML portion of the DOM Level 1 Recommendation. Michael has authored numerous articles and is a frequent speaker at industry events. He holds a bachelor's degree from the University of Michigan, and did graduate study specializing in data analysis and computer simulation of international conflict.

MIKE.CHAMPION@SOFTWAREAGUSA.COM

WRITTEN BY **KIRSTAN VANDERSLUIS**

Integration Using XML

From Chapter 7

XML-based Integration Use Cases

Organizations have found value in using XML for integration tasks within the enterprise and across businesses in information exchange projects. In this section, we describe several common cases where XML has proven valuable for integration.

Application Integration

The concept of application integration has been around for a couple of decades. A common integration strategy has been to transfer messages between systems to move information and report events within the enterprise. While these messages generally had custom formats in the past, companies are increasingly adopting XML as the message format. Because of the ease of adoption of XML, an increasing number of applications in the enterprise, whether packaged or custom written, are able to participate in integration projects.

XML-based APIs

Many packaged applications, such as Siebel, SAP, and PeopleSoft, are exposing their functionality through XML-based application programming interfaces. This allows other applications within the enterprise to integrate with the packaged application. For example, an SAP system might provide order provisioning capabilities to a custom-developed order entry Web application.

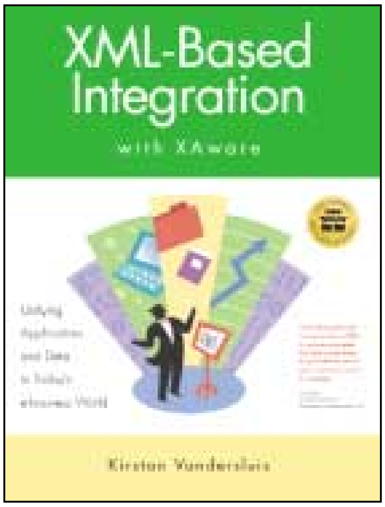
While the calls into the packaged application still require significant analysis and development work to function in an integrated manner with the rest of the enterprise, the use of XML removes one obstacle to integration – namely, the format of the information – and makes it straightforward for applications to exchange information.

Aggregation

XML introduces an opportunity within an organization to create logical views of information within the enterprise. The rich hierarchical structure enabled by XML lets an organization express business objects as XML structures. A complex structure such as a customer can be specified by an XML Schema, then software can be written to populate that structure from one or more data sources within the enterprise. The benefit is that applications can then be designed to use the defined XML structure of the business object, without detailed knowledge of where the actual data comes from. This ability to create an abstraction layer for applications in this manner is a core feature of EII systems.

Business to Business Integration

Business to Business Integration is the activity of automating business processes between companies, such as ordering parts, paying invoices, and reporting inventory needs. By performing these functions electronically using messages, many processes can be automated that previously required human intervention. Electronic Data Interchange (EDI) has been in use for over 25 years, and is especially prevalent in established industries such as the automotive industry. Unfortunately, EDI is expensive to implement, and trading communities have largely centered around a few dominant manufacturers who have exerted their will on smaller trading partners to improve the efficiency of the large company.



Reprinted with permission from the book *XML Based Integration with XAware* by Kirstan Vandersluis (www.maxpress.com/catalog/xaware.html).

While EDI is still used in many industries, the simplicity and relative ease of implementation of XML is attractive to new entrants, and most newer trading communities have adopted XML instead of EDI due to its overall ease of adoption. Organizations such as RosettaNet (www.RosettaNet.org) and ebXML (www.ebXML.org) have brought together constituents from around the world interested in increased efficiency of business processes across companies. The result in each case is a set of defined processes that use XML as the format for passing messages between companies in business transactions.

Of course, the format used for messages is just one technology decision for moving information between businesses. Businesses must operate under trusted relationships, and activities within any given process, such as order management, must be sequenced in an agreed order. Nevertheless, XML has played a key role in removing barriers to electronic business by providing an inexpensive infrastructure to all participants.

Information Exchange

Many industries and government agencies need to exchange information for reporting, monitoring, and other purposes. XML has become the standard format in which to exchange this information. Many of the recent electronic government initiatives use XML as the core technology when information exchange is required. For example, the Environmental Protection Agency (EPA) requires state-level agencies to provide information on companies that potentially create pollutants or other environmental hazards. The EPA defined XML Schemas that the states now conform to when reporting such information to the federal EPA. This is a common scenario among both government and industry groups that require the sharing of information. A central authority creates an XML Schema or format that participants then use to move information in an agreed format.

Information Distribution

XML has proven to be an ideal format for distributing information to interested parties. For example, one of the early XML success stories was Rich Site Summary (RSS), an XML format for content publishers of any size to describe the content of their Web site. RSS allows a site operator to distribute summary information about articles housed at the site to interested parties. Any Web site can then display headlines with pointers to the actual content. The benefit is that sites become more dynamic and interesting by displaying summaries of articles of interest to their visitors. Publishers of content benefit because their articles are widely advertised to targeted audiences, who then visit the home site to view the actual article. RSS is implemented by many of the top news-producing Web sites, including CNN, CNET, BBC, and ZDNet.

Web Services

One of the most popular uses of XML is in Web services. As discussed in Chapter 4, Web services is a distributed programming model built on XML standards. It allows a service to be built and described using XML, and a requestor to initiate an operation for the service using SOAP, another XML standard. Again, XML is the key enabling technology that has made it feasible for a large number of participants to engage in a coordinated effort, distributed programming.

WRITTEN BY **BOGDAN BLASZCZAK**

What Color is Your Schema?

Add your information to the standard

There is no doubt that the World Wide Web Consortium (W3C) XML Schemas enjoy quite a success. They are used in many major software environments and are applied to many domains, from finance to the Search for Extra Terrestrial Intelligence (SETI).

But what if, despite all of that, the standard schema facilities aren't enough to solve your problems? Fortunately, the W3C working group left the door open for schema extensions. You can use <annotation> elements to add your own information to a schema. Also, all schema elements allow additional attributes qualified with a namespace other than the XML Schema namespace itself.

Listing 1 shows a sample schema with an annotation. An annotation can be the first child of any other schema element (see www.w3.org/TR/xmlschema-1). An <annotation> can have any number of <documentation> or <appinfo> children.

The <documentation> element is expected to provide human-readable information for documentation purposes. The xml:lang attribute can be used to identify the language of the information. Hence, you can use multiple <documentation> components to provide information in different languages. You can even use HTML or any other XML-based notation (it needs to be well formed), if a plain, unstructured text is not enough.

The <appinfo> element is what we are really looking for. Its content is intended to be processed by applications. The <appinfo> content does not interfere with the XML document validation. The only restriction on the content is that it needs to be a well-formed XML.

In Listing 1, the <appinfo> content could be intended for a document management system. The additional information could be used for document classification (a blue book or a red book) and for access control (public or restricted). Since a schema defines a class of documents, the annotations that you provide in the schema can be considered applicable to all compliant documents. However, it really is up to you and your application to assign any specific semantics to the annotation content. The W3C specification does not venture into that territory. Remembering to reference the correct schemas in all documents is imperative.

<appinfo> Examples

Previously, I was involved in the development of a configuration management server for distributed voice processing systems. The problem included generation and validation of software configuration sets. We decided to use XML Schema to define the configuration data options and constraints. As you can imagine, we discovered quite a few interdependencies between elements of the configuration sets. Since the W3C Schema language lacks a notation for general constraints between elements, we ended up applying our own <appinfo> extensions to correctly describe those special cases. Our GUI editor used both the schema and extension information to present the user with valid options and to validate any free input. It also used additional 'grouping' hints to intelligently manage the screen layouts. Similar techniques will be used in the next generation of Intervoice OAM&P solutions, Control Center, to efficiently provide system configuration and provisioning functions.

> "But what if... the standard schema facilities aren't enough to solve your problems?"

If what you need are powerful validation capabilities, you can extend your schema with Schematron rules. Schematron is an alternative schema language defined by Rick Jelliffe at Academia Sinica (see http://xml.ascc.net/resource/schematron/). The Schematron rule notation uses tree path references to specify assertions on XML document content (XPath with XSLT extensions). Schematron rules are quite useful for defining complex dependencies among arbitrary document elements. They nicely complement the grammar-based approach of the W3C schema language.

The following Schematron annotation sample came from www.topologi.com/public/Schtrn_XSD/Paper.html. Notice the use of a predicate in the <sch:rule> element.

```
<xsd:appinfo>
  <sch:pattern name=
    "Constraint of the Title element"
xmlns:sch=
"http://www.ascc.net/xml/schematron">
<sch:rule
context="Person[@Title='Mr']">
```

AUTHOR BIO

Bogdan Blaszczak is director of software research & development for Dallas-based Intervoice.

```
<sch:assert test="Sex = 'Male'">
If Title is "Mr" then Sex must be
"Male".
</sch:assert>
</sch:rule>
</sch:pattern>
</xsd:appinfo>
```

Since the Schematron rules are written using XPath and XSLT notation, a standard XSLT processor can be used to process them, requiring two steps for completion. You first need to convert the Schematron notation into a validating XSLT stylesheet using one of the 'metastylesheets' provided at http://xml.ascc.net/schematron/1.5. In the second step, the new validating stylesheet is used on a XML instance document. The processor output will contain any failed assertions thus pointing out problems in the document.

The examples discussed up until now focused on XML validation capabilities. Another good reason for extending a schema is to define data mappings from XML to a different data space.

If your database does not support XML natively, you will usually need to define how the information is to be stored in tables. For example, in the Microsoft SQL Server 2000 environment, you can use XML Bulk Load (a stand-alone COM object) to efficiently load XML data into tables. XML Bulk Load interprets mappings provided in an annotated schema to identify the target tables. Listing 2 is a sample mapping.

> "... you can add your own attributes to the standard schema elements as long as you qualify them with your own namespace."

The <sql:relationship> element identifies CustomerID in the Orders table as a foreign key that refers to the CustomerID primary key in the Customers table. The values of attributes of <sql:relationship> refer to XML and SQL elements declared within the schema body.

Attributes from Outer Space

As mentioned earlier, you can add your own attributes to the standard schema elements as long as you qualify them with your own namespace. Such attributes become yet another annotation of your schema. It is a more concise way of extending a schema than with an <annotation>, but it is also more suited for simple values than independently structured information.

If we dip again into the XML mappings of Microsoft SQL Server 2000, a good example of extending schema through custom attributes can be found in XML views (http://msdn.microsoft.com/library/default.asp?url=/library/enus/sqlxml3/htm/ssxsdannotations_0gqb.asp). XML views allow you to store and retrieve data in the XML format. You can also use XPath queries against an XML view.

An annotated schema that defines a XML view may look like Listing 3. In this sample, there is a direct mapping of XML elements (declared using the "xsd" namespace) to database tables (<sql:relation>) and fields (<sql:field>).

Summary

It isn't often that you design your own extensions to XML schemas. However, it is comforting to know that you can do so by adding additional information to the standard notation and by adding incremental processing to your environment.

BOGDAN.BLASZCZAK@INTERVOICE.COM

LISTING 1 • Sample schema with an <annotation

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <!--    *** The XML schema annotation ***    -->
  <xsd:annotation>
    <xsd:documentation> The Corporate Blue Books
       </xsd:documentation>
    <xsd:appinfo>
      <ivi:docs
   xmlns:ivi="http://www.intervoice.com/xml/meta" >
        <ivi:blue/>
        <ivi:public/>
      </ivi:docs>
    </xsd:appinfo>
  </xsd:annotation>
  <!--   *** The XML document structure: a book of
     rules ***    -->
  <xsd:element name="book">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="rule"
           maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

LISTING 2 • Sample data mapping

```
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:sql="urn:schemas-microsoft-
               com:mapping-schema">
  <xsd:annotation>
    <xsd:appinfo>
      <sql:relationship name="CustOrder"
                 parent="Customers"
                 parent-key="CustomerID"
                 child="Orders"
                 child-key="CustomerID" />
    </xsd:appinfo>
  </xsd:annotation>
<!--  *** The schema with declarations of customer
      and order elements ***    -->
</xsd:schema>
```

LISTING 3 • Annotated schema that defines an XML view

```
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:sql="urn:schemas-microsoft-
               com:mapping-schema">
  <xsd:element name="employee"
sql:relation="Employees">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="name" type="xsd:string"
           sql:field="Name"/>
      </xsd:sequence>
    <xsd:attribute name="id" type="xsd:integer"
       sql:field="EmpID"/>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

WRITTEN BY **DEEPAK VOHRA & AJAY VOHRA**

SimpleType and ComplexType in a Schema

Designing your own

This tutorial discusses the simpleType and complexType XML Schema structures and their corresponding representations in an XML document. XML Schema is used as the basis of an XML document structure, and some of the XML technologies, such as JAXB, are based on XML Schema.

Overview

An XML Schema is an XML-based representation of the structure of an XML document. XML Schema supports data types and namespaces; a DTD does not. In this tutorial, the simpleType and complexType structures used to represent a XML document will be discussed.

SimpleType Declaration

SimpleTypes are custom data types used with element and attribute declarations. A simpleType is represented with a <xs:simple Type/> declaration. An <xs:simpleType/> element is used to constrain character data in a element or a attribute declaration. A simpleType is used to constrain data types only; a simpleType does not have attributes or element nodes. With an element declaration the simpleType is used as:

```
<xs:schema>
<xs:element name="A"
type="simpleTypeA"/>
<xs:simpleType name="simpleTypeA">
</xs:simpleType>
</xs:schema>
```

or

```
<xs:element name="A">
<xs:simpleType>
</xs:simpleType>
</xs:element>
```

With an attribute declaration, the simpleType is used as

```
<complexType>
<xs:attribute name="AttributeA"
type="simpleTypeA"/>
</complexType>
```

or

```
<xs:attribute name="AttributeA">
<xs:simpleType>
</xs:simpleType>
<xs:attribute>
```

The <xs:simpleType/> declaration may be present in an <xs:element/> element, <xs:attribute/> element, <xs:restriction/> element, <xs:list/> element, <xs:union/> element, or the <xs:schema/> element.

The <xs:simpleType/> element may contain one of the <xs:restriction/>, <xs:list/>, <xs:union/> elements.

SimpleType with Restriction Element

A simpleType used to constrain a decimal data type would be

```
<xs:element name="A">
<xs:simpleType>
<xs:restriction base="xs:decimal">
<xs:minExclusive value="1.0"/>
<xs:maxInclusive value="5.0"/>
<xs:totalDigits value="3"/>
<xs:fractionDigits value="2"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
```

A corresponding representation of element Element "A" in an XML document would be

```
<A>5.0</A>
```

Element "A" in an XML document may have values between 2.00 and 5.00 (inclusive). Declaration <A>1.0</A> or <A>25.0</A> would not validate with the Schema.

A simpleType to constrain the length of a string data type would be

```
<xs:element name="A">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:minLength value="5"/>
<xs:maxLength value="10"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
```

A corresponding representation of "A" in an XML document would be

```
<A>string</A>
Declaration <A>str</A>
```

or

```
<A>stringelement</A>
```

would not validate with the Schema.

An element with a fixed length would be represented with simpleType as

```
<xs:element name="A">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:length value="5"/>
```

AUTHOR BIOS

Deepak Vohra is a Web developer and Sun Certified Java 1.4 Programmer. Ajay Vohra is a senior software engineer with Compuware

Ajay Vohra is a senior software engineer with Compuware..

```
</xs:restriction>
</xs:simpleType>
</xs:element>
```

An example of a corresponding representation of 'A' in an XML document would be

```
<A>strin</A>
```

An attribute with an enumeration of values would be represented with simpleType as

```
<xs:element name="A" >
  <xs:complexType>
    <xs:attribute
     name="AttributeA">
      <xs:simpleType>
        <xs:restriction
         base="xs:string">
          <xs:enumeration
           value="a"/>
          <xs:enumeration
           value="b"/>
         </xs:restriction>
       </xs:simpleType>
     </xs:attribute>
   </xs:complexType>
 </xs:element>
```

A corresponding representation in an XML document is

```
<A AttributeA="a"/>
```

"AttributeA" may have one of the values, "a" or "b."

The <xs:restriction/> element in a <xs:simpleType/> may be used to constrain the value space of data types derived from the string data type with the <xs:whitespace/> element.

The whiteSpace element is represented as <xs:whiteSpace value="preserve|collapse|replace"/>. The value of the attribute "value" of element <xs:whitespace/> may be "preserve," "collapse," or "replace."

If the value of an attribute "value" is "preserve," the whitepaces in the string data type are not changed. If the value of the attribue "value" is "replace," all occurrences of #x9(tab), #xA(linefeed), and #xD(carriage return) are replaced with a #x20(space).

An element with a whiteSpace restriction on its string value is represented in a schema as

```
<xs:element name="A">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:whiteSpace value="replace"/>
<xs:restriction>
</xs:simpleType>
</xs:elementA>
```

If the value of attribute "value" is "replace," the string in "A" in an XML document represented as

```
<A>    Trim
    String
WhiteSpace</A>
```

would be parsed as "Trim String White Space."

If the value of the attribute '"value" is "collapse," the leading and trailing #x20(space) are removed, and contiguous occurances of #x20(space) are merged into a single #x20(space).

"XML Schema is used as the basis of an XML document structure"

If the value of the attribute "value" is "collapse," the string in "A" in an XML document represented as

```
<A>  Trim              String
WhiteSpace        </A>
```

would be parsed as "Trim String White Space".

<xs:restriction/> element <xs:pattern/> is used to constrain the value space of data types to literals which match a regular expression.

An example of a simpleType with pattern restriction would be

```
<xs:element name="A">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:pattern value="[1-5]{2}(-[5-
10]{3})"/>
</xs:simpleType>
</xs:element>
```

An example of a corresponding representation in a XML document is

```
<A>25-567</A>
```

SimpleType with List Element

A simpleType may be used to represent a list of values in a XML document element with

```
<xs:list/> element.
<xs:element name="A">
<xs:simpleType>
<xs:list itemType="xs:decimal"/>
</xs:simpleType>
</xs:element>
```

An element with a list of values would be

```
<A>1.0 2  3</A>
```

SimpleType with Union Element

A simpleType may be used to represent a union of simpleTypes with the <xs:union/> element.

```
<xs:element name="A">
<xs:simpleType>
<xs:union>
<xs:simpleType>
<xs:restriction base="xs:integer"/>
</xs:simpleType>
<xs:simpleType>
<xs:restriction base="xs:string"/>
</xs:simpleType>
</xs:union>
</xs:simpleType>
</xs:element>
```

An element with a union of simpleTypes in an XML document would be

```
<A>15</A>
```

or

```
<A>element A</A>
```

"A" may be either an integer or a string.

ComplexType Declaration

A complexType is used to constrain elements and attributes in an XML document. It is represented with <xs:complex Type/> .

A complexType declaration may occur in the <xs:schema/> declaration or a <xs:element/> declaration.

If a complexType is declared in the <xs:schema/> element, the complexType may be referred to in the 'type' attribute of an element declaration.

HOME
ENTERPRISE SOLUTIONS
CONTENT MANAGEMENT
DATA MANAGEMENT
XML LABS

```
 <xs:schema>
<complexType name="complexTypeA">
</complexType>
<xs:element name="A"
type="complexTypeA"/>
</xs:schema>
```

A text(PCDATA) element is represented with a complexType as

```
     <xsd:element name="A">
      <xsd:complexType mixed="true">
        <xsd:complexContent>
          <xsd:restriction
           base="xsd:anyType">
          </xsd:restriction>
        </xsd:complexContent>
      </xsd:complexType>
    </xsd:element>
```

The corresponding representation of a text element in an XML document is

```
<A>Text Element</A>
```

An empty element is represented with a complexType as

```
    <xsd:element name="A">
      <xsd:complexType mixed="false">
        <xsd:complexContent>
          <xsd:restriction
           base="xsd:anyType">
          </xsd:restriction>
        </xsd:complexContent>
      </xsd:complexType>
    </xsd:element>
```

The corresponding representation of an empty element in an XML document is <A/>.

An unordered element set is represented with a complexType element as

```
    <xs:element name="A" >
      <xs:complexType>
        <xs:all>
          <xs:element name="B"
           type="xs:string"/>
          <xs:element name="C"
           type="xs:string"/>
          <xs:element name="D"
           type="xs:string"/>
        </xs:all>
      </xs:complexType>
    </xs:element>
```

A corresponding representation of an unordered element set in an XML document is

```
<A>
<D>Element D</D>
<B>Element B</B>
<C>Element C</C>
</A>
```

An element with a choice and a sequence is represented with a complex Type as

```
    <xs:element name="A" >
      <xs:complexType>
        <xs:sequence>
          <xs:choice>
          <xs:element name="B"
           type="xs:string"/>
          <xs:element name="C"
           type="xs:string"/>
          </xs:choice>
          <xs:element name="D"
           type="xs:string"/>
         </xs:sequence>
        </xs:choice>
      </xs:complexType>
    </xs:element>
```

A corresponding representation of an element with a choice and sequence in a XML document is

```
<A>
<B>Element B</B>
<D>Element D</D>
</A>
```

ComplexType with Mixed Content

An element with a mixed content is represented in a complexType as

```
    <xs:element name="A" >
      <xs:complexType mixed="true">
        <xs:element name="B"
         type="xs:string"/>
    </xs:complexType>
   </xs:element>
```

The corresponding representation of a mixed content element in an XML document is

```
<A>Element Text
<B>Element B</B>
</A>
```

ComplexType with SimpleContent Element

Elements with character data and attributes are represented with the <xs:simpleContent/> element.

```
<xs:element name="A">
<xs:complexType>
 <xs:simpleContent>
  <xs:extension base="xs:string">
   <xs:attribute name="attr"
    type="xs:integer"/>
  </xs:extension>
 </xs:simpleContent>
</xs:complexType>
</element>
```

The corresponding representation of a simpleContent element in a XML document is

```
<A attr="15">simpleContent Type
Element</A>
```

The simpleContent element may also be used with the <xs:restriction/> element.

```
<xs:element name="A">
<xs:complexType>
 <xs:simpleContent>
  <xs:restriction base="xs:string">
  <xs:minLength value="5"/>
   <xs:attribute name="a"
    type="xs:integer"/>
  </xs:restriction>
 </xs:simpleContent>
</xs:complexType>
</element>
```

The corresponding representation of a simpleContent element with a restriction in a XML document is

```
<A a="15">Element A</A>
```

ComplexType with ComplexContent Element

An element with constraint on its elements and attributes may be represented with the <xs:complexContent/> element. An example of this element would be

```
<xs:element name="A">
<xs:complexType>
<xs:complexContent>
<xs:restriction base="anyType">
<xs:sequence>
<xs:element name="B" type="xs:string"/>
```

```
<xs:element name="C" type="xs:inte
  ger"/>
</xs:sequence>
</xs:restriction>
</xs:complexContent>
</xs:complexType>
</xs:element>
```

The corresponding representation of element 'A' with complexContent in a XML document is

```
<A><B>Element B</B><C>25</C></A>
```

The complexContent element may be used to extend another complexType.

```
<xs:complexType name="complexTypeA">
<xs:sequence>
<xs:element name="B"/>
<xs:element name="C"/>
</xs:sequence>
</xs:complexType>
<xs:element name="A">
<xs:complexType>
<xs:complexContent>
<xs:extension base="complexTypeA">
<xs:sequence>
<xs:element name="D"/>
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
```

The corresponding representation of a complexContent element with an extension on another complexType is

```
<A>
<B>Element B</B>
<C>Element C</C>
<D>Element D</D>
</A>
```

Conclusion

This tutorial discussed the simpleType and complexType XML Schema structures, which may be used to design custom simpleType and complexType elements.

Resources:

- *XML Schema:* Structures Specification.

DVOHRA09@YAHOO.COM

AJAY_VOHRA@YAHOO.COM

Mindreef SOAPscope 4.0 First to Connect Development, Testing and Support Teams

(Hollis, NH) – Mindreef, Inc., has released Mindreef SOAPscope 4.0, the company's flagship Web services diagnostics system. SOAPscope allows for the complete capture and assembly of all evidence related to a Web services problem. With Version 4.0, SOAPscope is the first to enable seamless sharing of these artifacts between developers, testers, operations, and support teams – expediting the problem-solving process and overcoming "finger pointing" delays. SOAPscope 4.0 also offers significant new features, including support for secure Web services using SSL certificates, and the industry's first diagnostic support for the newly released WS-I Basic Profile 1.1 and WS-I Attachments Profile 1.0.

SOAPscope 4.0 introduces two concepts that advance the role of diagnostics for SOAs: "Workspace" creates a place to easily collect and organize problem data, while "Packaging" enables clear and consistent sharing of the problem data with the appropriate specialist. SOAPscope's diagnostic tools (debugging, testing, tuning) then allow the specialist to operate on the problem data and quickly solve the problem.

www.mindreef.com

webMethods Portal Provides Charter Communications with Single Access Point

(Fairfax, VA) – Charter Communications, Inc., a broadband communications company, has chosen webMethods, Inc., the industry's first Web services infrastructure company, to deploy webMethods Portal to aggregate business-critical internal Web-based applications, tools, and Web sites into a single unified Web interface as well as serve as an internal document repository.

webMethods Portal will streamline Charter employee access in the monitoring of network status, services and products. It also will serve as Charter's single point of access for providing secure, role-based system access on the basis of employee location and responsibilities. Implementing webMethods Portal will improve employee efficiency, as employees will no longer need to access several systems to perform duties and only those employees with specifically assigned access will utilize the systems.

www.charter.com, www.webMethods.com

Entrust and Vordel to Deliver a Comprehensive Security Solution

(Dublin & Dallas) – Vordel, an XML security company, and Entrust, Inc., a provider of identity and access management solutions, have signed a global distribution agreement that licenses Entrust to resell and support Vordel products worldwide. Incorporating VordelSecure, the XML gateway, and VordelDirector, the XML security server, into the Entrust Secure Identity Management Solution portfolio will help enable Entrust to expand its solution, which secures both Web site and Web service access to its enterprise customers.

The combined solution from Entrust and Vordel means that enterprises will not require separate security architectures for their business services and Web services applications. It offers a unified, consistent access policy infrastructure to enforce security across existing and new Web and Web services applications, thereby helping to enable greater efficiencies and cost savings.

This is a key extension to Entrust's Secure Identity Management Solution that provides authentication, authorization and provisioning capabilities. Organizations will have the capability to define authentication and access policy centrally, and have it enforced for both Web and Web services transactions. Vordel's XML security capabilities can be readily added to Entrust GetAccess) deployments due to the tight integration between the two product families. This agreement also provides customers with a single point of purchase and global support for their Web and Web services security needs.

www.entrust.com, www.vordel.com

IBM and Sun Join Microsoft on Web Services Event Spec

IBM, Sun Microsystems, and Computer Associates International now support the WS-Eventing specification for subscribing to Web services–based events, joining original developers Microsoft, BEA Systems, and Tibco Software.

The specification will define a baseline set of operations that allow Web services to provide asynchronous notifications to interested parties. For example, an event notification could pertain to shipping of an order or e-mail arriving. Microsoft officials cited desires for interoperability between different specifications as reasons why rivals such as Sun and IBM are climbing aboard.

Xamlon, Inc. Releases Xamlon v0.9 with Visual Studio.NET Integration

(La Jolla, CA) – Xamlon, Inc., an XAML resource for developers, has announced a new beta release of its development tool, Xamlon. The Xamlon engine enables developers to use XAML to rapidly build and deploy applications for current versions of Windows with the confidence that their applications will easily port to future Windows releases. A key new feature of Xamlon v0.9 is the integration of Visual Studio.NET 2003, which allows developers to harness the power of Visual Studio to design rich application interfaces and automatically generate the XAML code for those interfaces. Other new features of Xamlon v0.9 include basic 3-D support, new samples and an Adobe Illustrator SVG to XAML converter. The beta is available for free public download at www.xamlon.com.

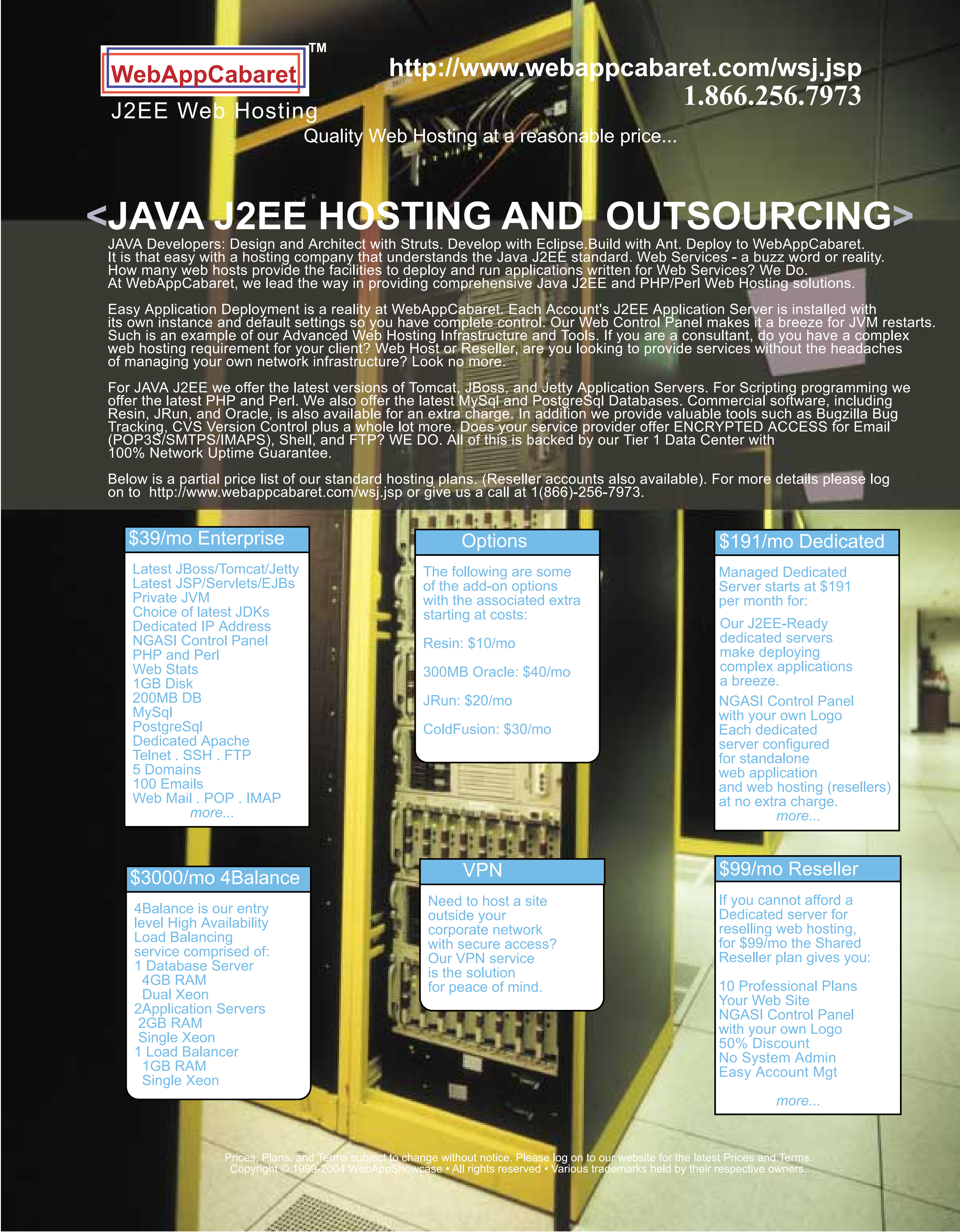
www.xamlon.com

StrikeIron Partners with DataFlux

(Research Triangle Park, NC) – August 30, 2004 — StrikeIron, Inc., a pioneer of online services and software to simplify working with Web services, and DataFlux Corporation, a provider of data management solutions, are partnering to provide components of the DataFlux Web-based technology as hosted StrikeIron Premium Web services available through the StrikeIron Web Services Business Network (WSBizNet).

Available now, the first component is the StrikeIron Address Verification Premium Web Service. This Premium Web Service is available on a real-time, on-demand basis with a "pay-per-use" subscription, providing customers with a lower cost and lower risk option to take advantage of DataFlux's data management capabilities. Small- to medium-size businesses and departments in larger companies can now take advantage of this capability without having to invest in additional infrastructure and tools.

www.strikeiron.com, www.dataflux.com



Middleware is Everywhere.
Can you see it?
4
5
3
2
1
IBM®