

WebServices

JOURNAL

.NET J2EE XML

WSJ2.COM

pg. 35

International
Conference & Expo

Edge 2004
EAST

February 24-26, 2004

Hynes Convention Center, Boston, MA



From the Editor
Final Answer

by Sean Rhody pg. 3

Industry Commentary
Mining Customer Gold with
Web Services for CRM

by John Wookey pg. 7

Product Review
Glue 4.1 from
The Mind Electric

by Paul Maurer pg. 30

RETAILERS PLEASE DISPLAY
UNTIL DECEMBER 31, 2003

\$6.99US \$7.99CAN



SYS-CON
MEDIA

Using Web Services for Business

Delivering the message

PAGE
36

FOCUS ON VERTICAL INDUSTRIES

Take It to the Bank

Implementing FpML



Andrew Parry

14

Web Services in the Insurance Industry

A dynamic industry takes the lead



Aziz Hussein

32

WSJ Feature: SOAP's Two Messaging Styles

Balancing their abilities against your needs

Rickland Hollar

8

Axis: Axis-izing Legacy Applications

Two approaches that work



Adelene NG

16

WSJ Feature: The Critical Need for Monitoring & Analysis

Remember why you chose Web services in the first place

Jothy Rosenberg

24

SOA: Demystifying Service-Oriented Architecture

Take the right approach



Jim Webber &

Savas Parastatidis 40

Security: Overcoming the Web Services Insecurity Complex

New standards and solutions address security concerns of companies 46

Gene Thurston

Standards: Transacting Business with Web Services, Part 2

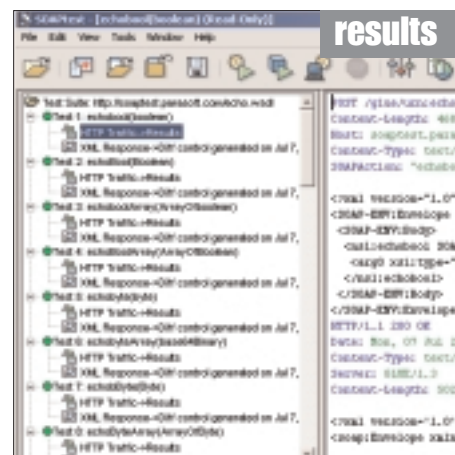
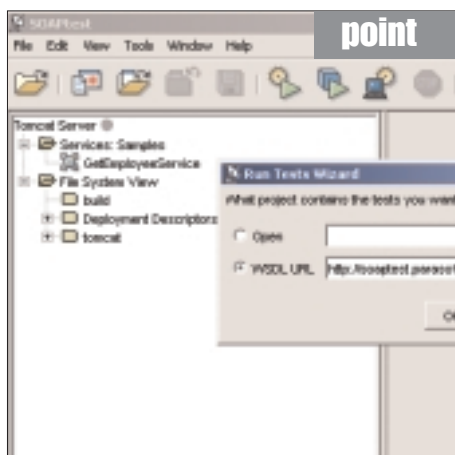
The coming fusion of BTM & BPM



Alastair Green &

Peter Furniss 50

Verify Web services instantly with Parasoft SOAPtest.



(It's as easy as 1-2-3.)

Parasoft **SOAPtest** is the first and only testing tool that enables you to instantly verify that your Web services are working properly.

Just point SOAPtest to your Web services WSDL files. It automatically creates test suites and runs comprehensive tests of the Web service's functionality and load capacities.

Server emulation for SOAP client development.

If you're developing a client-side application and need to verify functionalities such as credit card processing, oftentimes the Web services server is in use or otherwise unavailable for testing. Parasoft SOAPtest solves this problem by enabling you to emulate server responses and verify client functionality in just seconds.

SOAPtest also lets you quickly create templates written in Java™, JavaScript, and Python.

•----- For Downloads go to www.parasoft.com/ws11. Or call 888-305-0041.

Copyright ©2003 Parasoft Corporation. All rights reserved. All Parasoft product names are trademarks or registered trademarks of Parasoft Corporation in the United States and other countries. All other marks are the property of their respective owners.

Now supports WS-I Basic Profile 1.0

Compatibility:

Works with any SOAP-compliant implementation, including Microsoft and IBM Windows 2000/XP
Linux
Solaris

**A part of Parasoft Automated Error
Prevention (AEP) Solutions and Services**

PARASOFT
soaptest

Final Answer

INTERNATIONAL ADVISORY BOARD

Andrew Astor, David Chappell, Graham Glass, Tyson Hartman,
Paul Lipton, Anne Thomas Manes, Norbert Mikula,
Frank Moss, George Paolini, James Phillips, Simon Phipps

TECHNICAL ADVISORY BOARD

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

EDITORIAL EDITOR-IN-CHIEF

Sean Rhody sean@sys-con.com

INDUSTRY EDITOR

Norbert Mikula norbert@sys-con.com

PRODUCT REVIEW EDITOR

Brian Barbash bbarbash@sys-con.com

.NET EDITOR

Dave Rader davidr@fusiontech.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 EDITORS

Jamie Matusow jamie@sys-con.com

Jean Cassidy jean@sys-con.com

ASSISTANT EDITOR

Jennifer Van Winckel jennifer@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 DIRECTOR

Louis F. Cuffari louis@sys-con.com

ASSISTANT ART DIRECTOR

Tami Beatty tami@sys-con.com

CONTRIBUTORS TO THIS ISSUE

David Burdett, Alastair Green, Rickland Hollar, Aziz Hussein,
Paul Maurer, Adeline Ng, Savas Parastatidis, Andrew Parry,
Sean Rhody, Jonathan Rosenberg, Gene Thurston,
Jim Webber, John Wooley

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

©COPYRIGHT

Copyright © 2003 by SYS-CON Publications, Inc. All rights reserved. No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopy or any information storage and retrieval system without written permission. For promotional reprints, contact reprint coordinator, SYS-CON Publications, Inc., reserves the right to revise, republish, and authorize its readers to use the articles submitted for publication.

All brand and product names used on these pages are trade names, service marks, or trademarks of their respective companies.

SYS-CON Publications, Inc., is not affiliated with the companies or products covered in Web Services Journal.



Recently, I've been seeing some chatter around adding a programmatic aspect to Web services that is currently not part of the specifications – namely, adding object orientation (in particular inheritance, although I'm sure polymorphism is implied). I've thought about this, and I think it's a bad idea.

I have a lot of experience in object-oriented coding and design, from C++ and Java as well as in some languages where it was bolted on, like PowerBuilder (yes folks, at one point Powerbuilder was not object oriented). And although I can see advantages to the concepts of inheritance and polymorphism in a general purpose programming language, I don't see the same advantages in a non-programmatic, descriptive system where the main reason for existence is to create an abstracted, easily callable API regardless of platform, language, or software.

From my point of view, Web services is primarily about defining business processes. Web services are the business API of the programming world, an API that reflects the idea of a services-oriented architecture. And after years in the industry there's one thing I'm certain of – almost no business process needs to have inheritance.

Business processes are the computerized implementation of ways of doing business. Sometimes it's completely automated, sometimes it's partially automated, and sometimes it's almost completely manual, with just one or two systems for input and tracking. But regardless, people don't think in terms of inheritance. They think in terms of inventory processes and the like. What is important is how they do their job, not decomposing the job for reuse.

Not that reuse and its companions aren't important to business – they are. It's just that services, which are truly at the edge of an enterprise, are as much a part of the business world as they are the technical world. Or at least they should be. Given all the marshalling and other out-of-process work that has to be done to invoke a Web service, we shouldn't see them exposing granular methods of an object. There should be no “setCustomer-



WRITTEN BY

SEAN RHODY

FirstName” as part of a Web service – there should only be a “Customer” Web service. And because of the coarse-grained nature of Web services, they should not be part of some inheritance structure.

Java has the keyword *final* for a reason. In some circumstances it just makes sense not to let someone further down the line extend a class. Whether it's because of innate complexity or just

to keep certain methods from being misused, there are times when you don't want to allow inheritance. Nowhere is this more appropriate than at the system interface, where use of an API is turned over to coders who haven't the slightest clue what is behind the curtain. Nor should they have to know anything. But that's the point. You want the service to work as advertised. As a flat hierarchy.

I don't want to have to have a base inventory service from which my particular inventory service (such as a magazine inventory) is derived. Because as a user of the magazine service I don't want to have to know things about the base inventory service, I just want to check to see if a magazine is in stock.

Even further, one of the advantages of Web services is that both the consumer and the provider can be written in any language that supports Web services paradigms. Large companies that have invested decades of programming into CICS/COBOL can leverage that code just as easily as the most recent J2EE work or .NET assemblies. And languages and paradigms that have no object-oriented constructs, and there are many of them, can use Web services as long as they can connect. And it needs to stay that way. Leave inheritance for behind the edge, not at the edge. And that's my Final Answer. ☺

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



Why You Need to Know Your WSDL

The consumption of nearly every Web service begins with a WSDL file. With momentum building behind the "WSDL-first" methodology, the WSDL will be designed before the service is implemented. Unfortunately, the WSDL specification is ambiguous, and in some places contradictory. SOAP toolkits are not guaranteed to be uniform in their interpretation of WSDL, which creates potential for serious interoperability issues. With so much depending on this one document, it is critically important to be able to thoroughly understand a WSDL to avoid serious headaches later. Mindreef is pleased to introduce SOAPscope™ 2.0 with "Four Ways" to know WSDL. SOAPscope 2.0 increases your WSDL understanding and saves time whether you are developing a Web service, or diagnosing a problem.

Where Does It Start?

There are three common WSDL scenarios: Web service consumption (i.e., client development), traditional Web service development, and WSDL-first Web service development. Each scenario implies WSDL exposure at different stages of development. For clients, the WSDL already exists and has been finalized (hopefully). In traditional Web service development, the WSDL is automatically generated from the service. In WSDL-first development, the WSDL is designed first and the service is implemented in conformance. Whether using traditional or WSDL-first methodology, it's likely that the service implementation and WSDL will iterate together over several cycles. Choosing WSDL-first development indicates an XML Schema-oriented mind set, with a focus on Web service interoperability.

How Can You Know Your WSDL?

See it:

SOAPscope provides you with the familiar tools for visualizing WSDL documents (raw XML, and formatted XML with syntax coloring), but additionally offers Tree view (a structured view that allows you to collapse and expand the XML to focus on important details), and Pseudocode View™ (a concise summary of the WSDL that shows available methods and datatypes). Potential WSDL clients can use Pseudocode View when they assess a

service API and want a no-nonsense run down, whereas Web service developers can use the other views to quickly navigate and diagnose WSDL-related issues.

*SOAPscope 2.0
increases
your WSDL
understanding
and saves time*

Check it:

Web service consumers are as concerned with service quality as the provider. However, easily generated client-side stubs can provide a false sense of security. The moment an interoperability error occurs, clients will forward the problem to the provider. It is critical, then, to reduce the number of such problems before they occur. The Web Services Interoperability Organization (WS-I) has made an important contribution to interoperability with their Basic Profile 1.0. Mindreef goes even further with the SOAPscope 2.0 WSDL Analyzer which quickly checks a WSDL against WS-I Basic Profile 1.0, as well as the WSDL 1.1 specification and Mindreef's own knowledge base of "best practices." The Analyzer diagnoses, highlights, and presents real and potential problems along with the information you need to understand and resolve them. Running the Analyzer often during development allows iterative refinement of the WSDL and helps to maintain a high level of quality as the WSDL evolves.

Try it:

SOAPscope 2.0 makes it easy for you invoke any Web service without writing a single line of code. Potential clients of a WSDL can test-drive any service with an easy to use, intuitive, form-based interface. SOAPscope incorporates a thorough understanding of XML Schema, allowing you to

invoke services with flexibility and type safety. You can optionally exercise control over HTTP headers and the SOAP envelope. As with all SOAP traffic on a SOAPscope-enabled machine, SOAPscope logs your invocations for later review or replay. Web service developers can immediately test and debug their services at all points during the Web service development life-cycle. For WSDL-first development, this enables a developer to ensure their service keeps in lock-step with the WSDL that describes it.

Diff it:

Web services are living APIs; WSDLs might be iterated frequently, maybe even after deployment. If you are writing a Web service client, WSDL changes can lead to version mismatches and unexpected errors. SOAPscope's XML-aware diffing highlights every change for quick diagnosis. For the Web service developer, diffing provides the ability to keep on top of frequent changes. Diffing can even be applied to individual SOAP messages that SOAPscope has logged, to make it easy to discover what separates a good message from a bad one.



SOAPscope provides four ways to know your WSDL

Knowing your WSDL

SOAPscope 2.0 provides the crucial insight into the world of WSDL that every Web service producer or consumer should have. With the ability to see it, check it, try it, and compare it, you can quickly diagnose WSDL-related problems and maintain a high level of quality in your published WSDLs. SOAPscope 2.0 includes all the great SOAP sniffing and logging features that made version 1.0 the best Web service diagnostic tool available. See your SOAP, Know your WSDL: give SOAPscope 2.0 a try today by visiting us at <http://www.mindreef.com/nl0309>.

© Copyright 2003, Mindreef, Inc.

Mindreef, SOAPscope, Debugging the Data, and Pseudocode View are trademarks of Mindreef, Inc.

The names of companies and products mentioned herein may be the trademarks of their respective owners. This product uses Hypersonic SQL. This product includes software developed by the Politecnico di Torino, and its contributors.

"My purchase of SOAPscope has already more than paid for itself"

Jim Abers



NEW
SOAPscope 2.0
Still just \$99

4 ways to...KNOW YOUR WSDL

No matter how robust the system, one small WSDL mistake can break everything. That's why Mindreef integrated 4 powerful WSDL diagnostics into SOAPscope. SOAPscope 2.0 makes it easy to keep your WSDL clean.

See it: To be productive with WSDL you need different views for different tasks. Only SOAPscope gives you pseudocode, tree, syntax-colored XML & raw views to quickly understand any WSDL.

Try it: Invoke Web services without writing any code. "Test drive" a web service through an easy-to-use form. Easily trying different conditions is essential for productive troubleshooting, testing or debugging.

Diff it: A small change in a WSDL can cause a hard-to-find problem. Diff pinpoints the change and is fully integrated with the other SOAPscope diagnostics so it's there when you need it.

Check it: Check for validity, compliance and conformance against the W3C WSDL 1.1 standard, the WS-I Basic Profile and Mindreef's best practices learned in the Web Services trenches. If your WSDL doesn't pass cleanly through SOAPscope, it's not fully interoperable!

SOAPscope combines the industry's best Web services diagnostic tools into a powerful, comprehensive, easy-to-use package. SOAPscope is the only Web services diagnostic tool you will need whether you're creating your first WSDL or debugging production services.

"Finally there's a product that makes it easy for developers to tackle WSDL: SOAPscope 2.0!"

*Aaron Skonnard,
Instructor, DevelopMentor.*



**Let Mr. SOAPscope
check YOUR WSDL!**

FREE evaluation at
www.mindreef.com

"The Web Services Diagnostic Experts" **Mindreef**



Drowning
in a sea of WSDL
descriptions?

***The Strikelron™
Web Services Analyzer
to the Rescue!***

***Discover, invoke and understand
any Web service in just a few clicks.***

The Strikelron Web Services Analyzer provides software developers, website builders and business analysts the ability to locate and understand the data structure, data requirements, behavior and results of any Web service anywhere in the world. All the details you need to make use of a Web service and exploit it per a given business scenario are available at your fingertips.

The StrikeIron Web Services Analyzer allows you to:

- * Visualize any Web service in a few clicks
- * Invoke any Web service and fully understand its behavior
- * Cut development time from hours to minutes

The Web Services Analyzer is the first in a suite of Strikelron products that will make Web services the first choice of development shops and business users who are implementing Service-Oriented Architectures or are taking advantage of Web services with traditional applications such as Web sites and business intelligence solutions.



Delivering the World of Web ServicesSM

Download the Analyzer today at www.strikeiron.com

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

TECHNICAL DIRECTOR

Alan Williamson alan@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

DIRECTOR, SALES & MARKETING

Megan Ring-Mussa megan@sys-con.com

ADVERTISING SALES MANAGER

Alisa Catalano alisa@sys-con.com

ASSOCIATE SALES MANAGERS

Carrie Gebert carrie@sys-con.com

Kristin Kuhnle kristin@sys-con.com

SYS-CON EVENTS

PRESIDENT, SYS-CON EVENTS

Grisha Davida grisha@sys-con.com

CONFERENCE MANAGER

Michael Lynch mike@sys-con.com

NATIONAL SALES MANAGER

Sean Raman raman@sys-con.com

CUSTOMER RELATIONS/JDJ STORE

CIRCULATION SERVICE COORDINATORS

Niki Panagopoulos niki@sys-con.com

Shelia Dickerson shelia@sys-con.com

Edna Earle Russell edna@sys-con.com

JDJ STORE MANAGER

Rachel McGouran rachel@sys-con.com

SYS-CON.COM

VP, INFORMATION SYSTEMS

Robert Diamond robert@sys-con.com

WEB DESIGNERS

Stephen Kilmurray stephen@sys-con.com

Christopher Croce chris@sys-con.com

ONLINE EDITOR

Lin Goetz lin@sys-con.com

ACCOUNTING

FINANCIAL ANALYST

Joan LaRose joan@sys-con.com

ACCOUNTS RECEIVABLE

Kerri Von Achen kerri@sys-con.com

ACCOUNTS PAYABLE

Betty White betty@sys-con.com

SUBSCRIPTIONS

SUBSCRIBE@SYS-CON.COM

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

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.



Mining Customer Gold with Web Services for CRM

We all know that the ultimate goal of CRM is to achieve a consolidated, 360-degree view of the customer at any touch point. We

know that when companies know more about their customers, they can provide better service and personalized marketing and selling – which translates into more profitable customer relationships. But the reality for most organizations is that customer data is spread far and wide across departmental and organizational boundaries. Even when customer data is gathered into a centralized location, customer information is often not useful because customer-facing applications do not have access to it. Traditional efforts to provide access to customer information from all applications have required protracted custom integration projects and have met with limited success. However, the advent of Web services offers faster, standards-based integration for low cost, rapid ROI – and along with it, renewed hope for the success of CRM implementations.

Consider the challenge faced by telecommunication companies that generate customer information from a variety of customer touch points. Billing, order processing, and field service departments interact with customers through various channels – monthly bills, telesales representatives, and onsite service calls. To ensure a high-quality customer experience, they all need a complete view of customer data. But with data scattered across multiple systems, displaying accurate, current customer information is difficult.

Similarly, manufacturing enterprises that work with partners to install and service their products often run into trouble when customers change their minds about an order. While the customer may cancel the order with the manufacturer, the service partner is often not notified in time to cancel delivery and installation services because the partners' systems is not linked in to the manufacturer's systems. Or perhaps the technician brings the wrong parts or tools to complete a repair because he does not have access to the customer's current install base. Those of us who have spent countless hours on the phone with customer



WRITTEN BY
JOHN WOOKEY

service representatives know of this issue first-hand. With these frustrations, the customer will go elsewhere to place their next order.

The two situations outlined above illustrate two central challenges plaguing CRM implementations:

1. Consolidation of transactional customer data
2. The ability for customer-facing applications to access customer data

Companies can consolidate operational customer data using prebuilt repositories available from select enterprise applications vendors, or they can build one internally. Once cleansed and aggregated data from all data sources is loaded into a single repository, all systems would need to read from and write to that single repository in real time, or at the very least operate locally but synchronize with it in real time.

For a "single source of customer truth" to be of use, enterprises have to integrate all potential customer-facing applications to the customer data repository. However, companies have found that proprietary integration is not only time intensive and costly, but also is inordinately expensive to maintain. Worse, integration is inflexible; changes in operations or acquisitions and divestitures suddenly render the systems useless. Often, companies decide to forgo all but the essential integration points. The result? The customer information lies untouched.

Enter Web services. Web services excel at providing access to your customer data. They are based on open standards, which means that you don't need access to specialized proprietary skills to deploy or subscribe to Web services. They are quick to deploy because all systems use the same protocol. Many different systems can call the same Web service for updates, i.e. fewer Web services replace a large number of integrations. They are simple – one system merely calls another using the predefined protocols and triggers a process, or obtains information. Even partners and other parties can subscribe to the very same Web services, greatly lowering the cost of collaboration as well as

–Continued on page 57

SOAP's

Two

Messaging Styles

Balancing their abilities against your needs

■ To RPC, or not to RPC: that is the question. Whether 'tis nobler in the mind to suffer the control and dependency of coupling, or to take arms against a sea of troubles, and by opposing, end them?

The Simple Object Access Protocol (SOAP) offers two messaging styles: RPC (Remote Procedure Call) and document style. One is for creating tightly coupled, inter-object style interfaces for Web services components; the other is for developing loosely coupled, application-to-application and system-to-system interfaces. Some of you may have questions about the differences in the styles or the problems they are designed to solve. My goal here is to answer those questions. I'll first present the two styles in enough detail for you to gain an appreciation of their relative strengths and weaknesses; I'll then look at guidelines for their use.

The first question you may have is what is an RPC? An RPC is a way for an application running in one execution thread on a system to call a procedure belonging to another application running in a different execution thread on the same or a different system. RPC interfaces are based on a request-response model where one program calls, or requests a service of, another across a tightly coupled interface. In Web services applications, one service acts as a client, requesting a service;



WRITTEN BY
RICKLAND HOLLAR

the other as a server, responding to that request. RPC interfaces have two parts: the call-level interface seen by the two applications, and the underlying protocol for moving data from one application to the other.

The call-level interface to an RPC procedure looks just like any other method call in the programming language being used. It consists of a method name and a parameter list. The parameter list is made up of the variables passed to the called procedure and those returned as part of its response. This is true on both sides of the interface. Both sides believe they are calling, or are being called by, a locally running procedure. Wiring in between hides the complexity of moving data between the two applications.

For Web services, SOAP defines the wiring between the calling and called procedures. At the SOAP level, the RPC interface appears as a series of highly structured XML messages moving between the client and the server where the <Body> of each SOAP message contains an XML representation of the call or return stack.

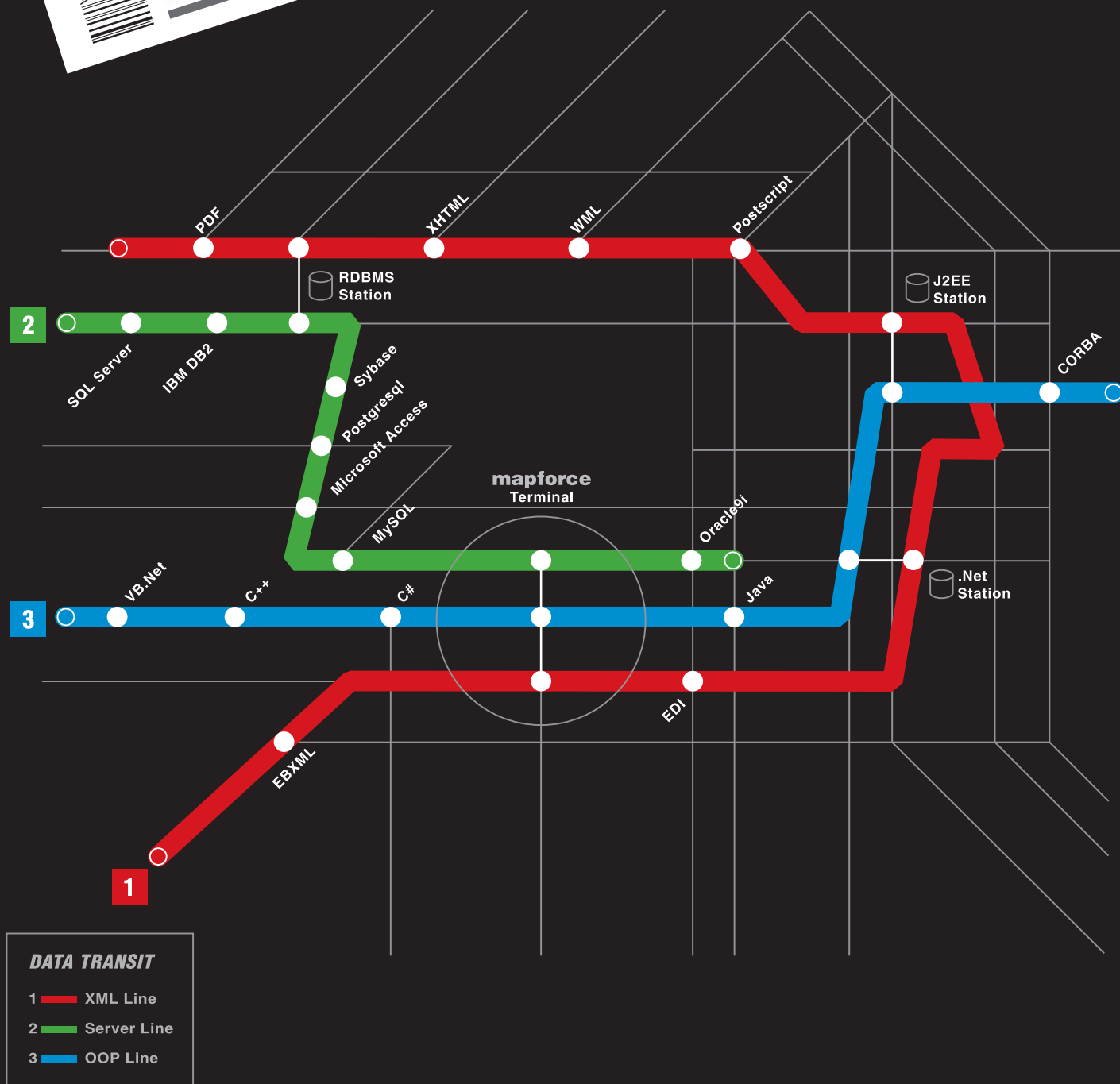
The transformation from call-level inter-

face to XML and back occurs through the magic of two processes – marshaling and serialization. Figure 1 illustrates the major components and steps involved in this process.

- The process begins with the client calling a method implemented as a remote procedure. The client actually calls a proxy stub that acts as a surrogate for the real procedure. The proxy stub presents the same external interface to the caller as would the real procedure, but instead of implementing the procedure's functionality, implements the processes necessary for preparing and transporting data across the interface.
- The proxy stub gathers the parameters it receives through its parameter list into a standard form, in this case, into a SOAP message, through a process called marshaling.
- The proxy stub encodes the parameters as appropriate during the marshaling process to ensure the recipient can correctly interpret their values. Encoding may be as simple as identifying the correct structure and data type as attributes on the XML tag enclosing the parameter's value or as complex as converting the content to a standard format such as Base64. The final product of the marshaling process is a SOAP message representation of the call stack.
- The proxy stub serializes the SOAP message across the transport layer to the server. Serialization involves converting the SOAP message into a TCP/IP buffer stream and



Next Stop, Mapforce 2004!



Introducing Mapforce 2004! A powerful new XML mapping tool from Altova, producer of the industry standard XML Development Environment, XMLSPY. Mapforce is a visual data mapping tool, which auto-generates custom mapping code in multiple output languages such as XSLT and Java, to enable programmatic XML-to-XML or XML-to-Database data transformations. Mapforce 2004 is the hassle-free transit for moving data from one format to another, enabling efficient information reuse. **Download a free trial now!**

ALTOVA®

www.altova.com

All other trademarks are the property of their respective owners.

transporting that buffer stream between the client and the server.

The server goes through the reverse process to extract the information it needs. A listener service on the server deserializes the transport stream and calls a proxy stub on the server that unmarshals the parameters, decodes and binds them to internal variables and data structures, and invokes the called procedure. The listener process may be, for example, a J2EE servlet, JSP (JavaServer Page), or Microsoft ASP (Active Server Page). The

messages, but doesn't specify how data is actually serialized across the interface. SOAP can bind to any protocol (usually either HTTP or Simple Mail Transport Protocol [SMTP]) for serialization, which means the specifications for those protocols actually define the serialization rules.

Section 7 of the SOAP specification defines the rules for marshaling RPC calls into XML messages (the most recent version of the SOAP 1.2 specification moves this information to the Adjuncts section, but the rules remain the same). Section 7 says to encode RPC

SOAP's built-in rules for encoding data values. Encoding is necessary any time the recipient needs to interpret an element's value as something other than a literal string, i.e. as an integer, floating point number, or MIME type. XML Schema offers an increasingly popular alternative that has all but obsoleted Section 5 encoding. Listings 1 and 2 illustrate the two options for a skeletal RPC method call; the `encodingStyle` attribute tells the recipient which scheme is being used.

With this background on RPC style in place, the next question is how does document-style messaging differ? The difference is primarily in the control you have over the marshaling process. With RPC-style messaging, standards govern that process. With document-style messaging, you make the decisions: you convert data from internal variables into XML; you place the XML into the `<Body>` element of the encapsulating SOAP document; you determine the schema(s), if any, for validating the document's structure; and you determine the encoding scheme, if any, for interpreting data item values. The SOAP document simply becomes a wrapper containing whatever content you decide. For example, the SOAP document shown in Listing 3 contains an XML namespace reference, `http://www.xyz.com/genealogy`, that presumably includes all the information a receiving program needs for validating the message's structure and content, and for correctly interpreting data values.

Figure 2 illustrates the steps in a typical document-style message exchange. If you compare the steps involved in this process with those involved in processing an RPC-style message from Figure 1, you will notice they are essentially parallel processes.

- The SOAP client uses an Extensible Stylesheet Language Transformation (XSLT) and the DOM parser, or some other means, to create an XML document.
- The SOAP client places this XML document into the `<Body>` of a SOAP message.
- The SOAP client optionally includes a namespace reference in the message that other applications can use for validating the encapsulated document's format and content. The namespace reference may be included as an attribute either on one of the SOAP elements or on the XML document's root element. If the document does not include a namespace reference, the client and server must agree on some other scheme for validating and interpreting the

“For Web services, SOAP defines the wiring between the calling and called procedures”

client and server reverse roles and the inverse process occurs to return the server's response to the client.

You may be curious about the distinction I make between marshaling and serialization, having seen the terms used interchangeably. I distinguish between them because with Web services different standards define the rules for the two processes. SOAP defines the rules for marshaling and encoding data into XML

method calls and responses as hierarchical XML elements, or structures, where the root-level element name is the method name in the case of the request and an arbitrary value in the case of the response, the structure's child elements are the method's parameters or return values; and each parameter or return value's elements are the data value or values it represents.

Section 5 of the SOAP specification defines

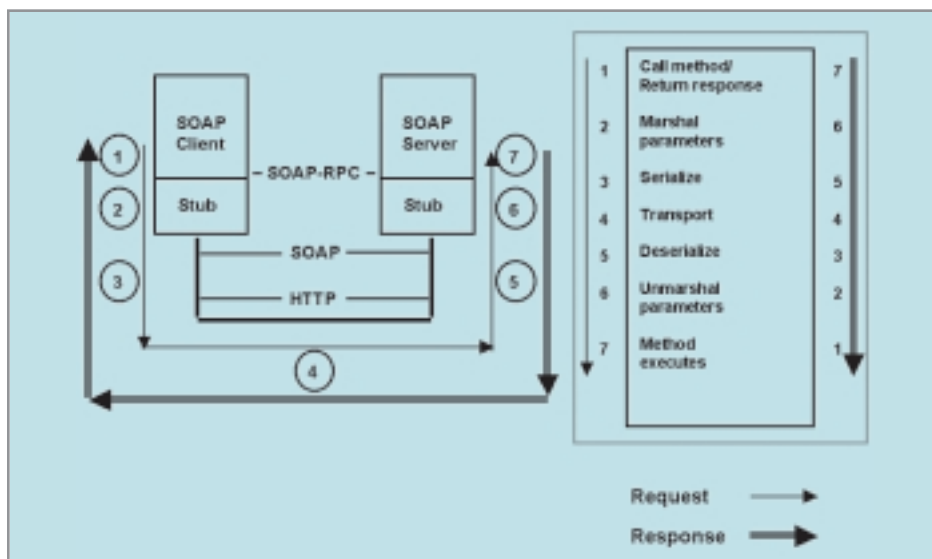


FIGURE 1 Major components and steps in marshaling and serialization

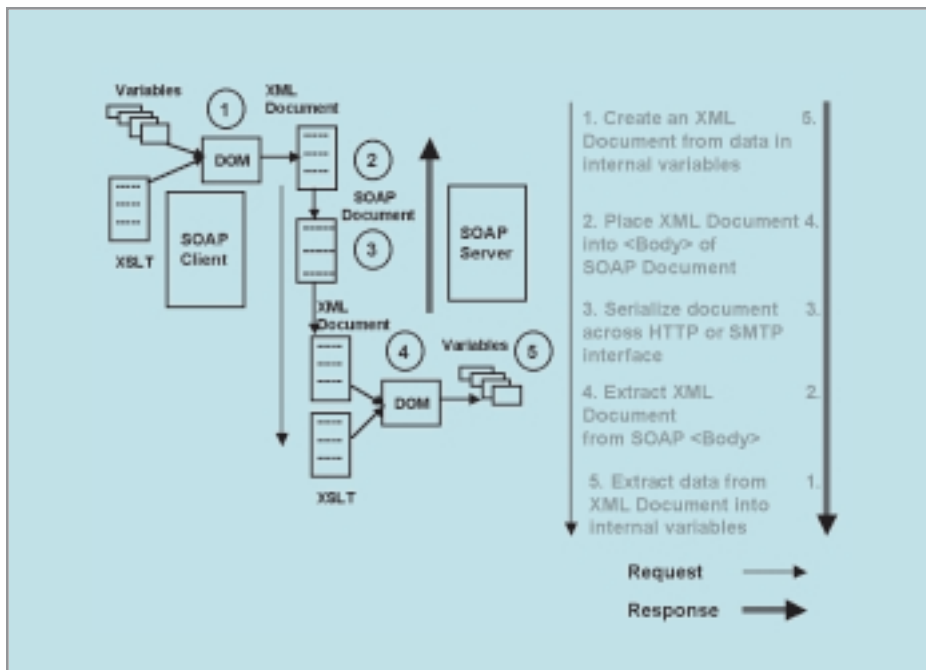


FIGURE 2 Steps in a document-style message exchange

document's contents.

- The SOAP client serializes the message to the SOAP server across either an HTTP or SMTP bound interface.

The SOAP server reverses the process, potentially using a different XSLT, to validate, extract, and bind the information it needs from the XML document to its own internal variables. The roles reverse and the two follow inverse processes for returning and accessing any response values. The rules guiding the marshaling process are the primary difference between this process and that for RPC-style messages. With document-style, you as the SOAP client's author create those rules.

Strengths and Weaknesses

Now that we've looked at both styles in some detail, we can discuss their relative strengths and weaknesses.

RPC-style messaging maps to the object-oriented, component-technology space. It is an alternative to other component technologies such as DCOM and CORBA where component models are built around programmable interfaces and languages such as Java and C#. RPC-style messaging's strength in this space lies in its platform independence. It offers a standards-based, platform-independent component technology, implemented over standard Internet protocols. One of the benefits of this style's XML layer is that clients and

servers can use different programming languages, or technologies, to implement their respective side of the interface, which means one side can choose one set of technologies, such as J2EE's JAX-RPC, while the other chooses a completely different set, such as .NET's C#. RPC-style messaging's standards heritage can be an important consideration in hybrid environments (one using multiple technologies such as J2EE and .NET) and can provide a transition path between different technologies.

RPC-style messaging's weaknesses include:

- Strong coupling:** If you change the number, order, or data types of the parameters to the call-level interface, you must make the change on both sides of the interface.
- Synchronicity:** Most programming languages assume synchronous method calls: the calling program normally waits for the called program to execute and return any results before continuing. Web services are asynchronous by nature and, in comparison to technologies such as DCOM and CORBA, long running. You may want to take advantage of Web services' asynchronous nature to avoid the user having to wait for calls to complete by developing asynchronous RPC calls, but that adds another level of complexity to your application. Some tools hide this complexity using callbacks,

or other techniques, to enable processing overlap between the request and the response. Check to see if the tools you're using let you choose between synchronous and asynchronous RPC calls.

- Marshaling and serialization overhead:**

Marshaling and serializing XML is more expensive than marshaling and serializing a binary data stream. With XML, at least one side of the interface, and possibly both, involves some parsing in order to move data between internal variables and the XML document. There is also the cost of moving encoded text, which can be larger in size than its binary equivalent, across the interface.

How do these drawbacks compare to those found in other component technologies? The coupling and synchronicity issues are common to RPC-based component technologies, so they are really not discriminators when making comparisons between these technologies. The marshaling and serialization overhead is greater for RPC-style messaging and places this messaging style at a relative disadvantage. However, with today's high-speed processors and networks, performance is generally not an issue.

Document-style messaging is clearly an option in any situation where an XML document is one of the interface parameters. It is ideal for passing complex business documents, such as invoices, receipts, customer orders, or shipping manifests. Document-style messaging uses an XML document and a stylesheet to specify the content and structure of the information exchanged across the interface, making it an obvious choice in situations where a document's workflow involves a series of services where each service processes a subset of the information within the document. Each service can use an XSLT to validate, extract, and transform only the elements it needs from the larger XML document; with the exception of those elements, the service is insensitive to changes in other parts of the document. The XSLT insulates the service from changes in the number, order, or type of data elements being exchanged. As long as the service creating the document maintains backwards compatibility, it can add or rearrange the elements it places into a document without affecting other services. Those services can simply ignore any addi-

tional data. Document-style messaging is also agnostic on the synchronicity of the interface; it works equally well for both synchronous and asynchronous interfaces.

Document-style messaging's weaknesses include:

- **No standard service identification mechanism:** With document-style messaging, the client and server must agree on a service identification mechanism: a way for a document's recipient to determine which service(s) need to process that document. SOAP header entries offer one option; you can include information in the document's header that helps identify the service(s) needed. WS-Routing makes just such a proposal. Another option is to name elements in the <Body> of the message for the services that need to process the payload the elements contain. You might ask how that differs from schema-based RPC-style messaging. You would be right in assuming there is little or no difference except possibly in terms of the number of "calls" that can be made per message. A third option is to perform either structure or content analysis as part of a service selection process in order to identify the services needed to process the document.
- **Marshaling and serialization overhead:** Document-style messaging suffers from the same drawbacks as RPC-style messaging in this area. However, the problem may be more

severe with document-style messaging. Document-style messaging incurs overhead in three areas: in using DOM, or another technique, to build XML documents; in using DOM, or SAX, to parse those documents in order to extract data values; and in mapping between extracted data values and internal program variables. Tools generating equivalent RPC-style interfaces optimize these transformations. You may have trouble achieving the same level of efficiency in your applications using standard tools.

Given these drawbacks, you may ask whether document-style messaging really is an alternative. The answer is yes. There are two compelling reasons to use document-style messaging. One is to gain the independence it provides. Its strength lies in decoupling interfaces between services to the point that they can change completely independently of one another. The other is that document-style messaging puts the full power of XML for structuring and encoding information at your disposal. The latter is one reason many consider document-style superior to RPC-style messaging.

Summary

Given their relative strengths and weaknesses, what guidelines should you use in choosing between the two messaging styles? RPC-style messaging's strength is as a bridging component

technology. It is a good option for creating new components and for creating interfaces between Web services and existing components – you simply wrap existing components with RPC-style Web services interfaces. RPC-style messaging is also an excellent component standard in situations where you are using multiple technologies, such as J2EE and .NET, and want to develop sharable components. So, there is clear justification for adopting an RPC style as a standard in these roles.

Document-style messaging's strengths are in situations where an XML document is part of the data being passed across the interface, where you want to leverage the full power of XML and XSL, and in instances where you want to minimize coupling between services forming an interface, such as in application-to-application and system-to-system interfaces. So, there is clear precedent here as well.

Neither style is a panacea. You must consider the relative strengths and weaknesses of each against your requirements. With these guidelines in mind, however, it is safe to adopt either based on your specific needs. ☺

About the Author

Rickland Hollar is a senior applications architect for the Central Intelligence Agency with over 30 years experience in the industry. Prior to joining the CIA, he was president of a Virginia-based software development firm.

■■■ rick_hollar@yahoo.com

Listing 1: Options for a skeletal RPC call

```
<?xml version="1.0"?>
<env:Envelope
  xmlns:env="http://schemas.xmlsoap.org/soap/envelope/"
  env:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:enc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-Instance">
  <env:Header>
    ...
  </env:Header>
  <env:Body>
    <SomeMethod>
      <param enc:arrayType="xsd:ur-type[2]">
        <item xsi:type="xsd:int">100</item>
        <item xsi:type="xsd:int">20</item>
      </param>
    </SomeMethod>
    ...
  </env:Body>
</env:Envelope>
```

Listing 2: Options for a skeletal RPC call

```
<?xml version="1.0"?>
<env:Envelope
  xmlns:env="http://schemas.xmlsoap.org/soap/envelope/"
  env:encodingStyle="http://www.xyz.com/sm">
  <env:Header>
    ...
  </env:Header>
```

```
<env:Body>
  <sm:SomeMethod xmlns:sm="http://www.xyz.com/sm">
    <param>
      <item>100</item>
      <item>20</item>
    </param>
  </sm:SomeMethod>
  ...
</env:Body>
</env:Envelope>
```

Listing 3: Sample XML namespace reference

```
<?XML version="1.0" ?>
<env:Envelope
  xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
  <env:Body>
    <xyz:family xmlns:xyz="http://www.xyz.com/genealogy">
      <parents>
        <father age="35">Richard</father>
        <mother>Kim</mother>
      </parents>
      <children>
        ...
      </children>
      <siblings>
        ...
      </siblings>
    </xyz:family>
  </env:Body>
</env:Envelope>
```

Download the code at
sys-con.com/webservices

Ektron Success Story #1096

Enterprise Web Content Management without the enterprise integration.



Beth knows that effective, timely communication is key in today's competitive market. By choosing Ektron's enterprise Web content management solution with Web Services and RSS Syndication, her partners and distributors are receiving accurate, up-to-date information. As a result the Web site has become a powerful, strategic, business tool.

Let us show you how an Ektron XML Web Content Management Solution can enhance your Web site.

Learn More at:
www.ektron.com/ws

Ektron - Redefining Web Content Management

Take It to the Bank

Implementing FpML

■ Global derivatives markets continue to enjoy strong growth. The Bank for International Settlements Quarterly Review for June 2003 notes "Data from the semiannual BIS survey on positions in global OTC derivatives markets at the end of December 2003 show there was an exceptionally sharp increase in gross market values, up by 43% to \$6.4 trillion."

The International Swaps and Derivatives Association (ISDA) is the global trade association representing participants in the privately negotiated derivatives industry, a business covering swaps and options across all asset classes. One of its most notable achievements has been the formation of a standardized document architecture that has greatly facilitated market evolution.

As a natural next step in market evolution, ISDA is working with market participants to express the hardcopy document architecture in softcopy form to allow increasing volumes in the derivatives markets to be handled with greater accuracy and lower cost, through extensive use of automation. As part of this process, ISDA formally adopted FpML.org, the organization that developed Financial products Mark-Up Language, a business information exchange standard for electronic dealing and processing of financial derivatives instruments. Based on XML, FpML establishes the industry protocol for sharing information and dealing in, financial swaps, derivatives, and structured products (see Figure 1).

FpML 4.0 provides XML Schema objects to describe the majority of derivative contracts by volume, derived directly from the legal framework established by ISDA, and is made freely available by them under public license. ISDA does not provide "off-the-shelf" support in the public standard; it is straightforward to create private extensions,



WRITTEN BY
ANDREW PARRY

for product description or workflow reasons.

These XML Schema objects are composed to form different distinct document prototypes, such as a Trade Confirmation. This document-centric approach allows us to form document instances that represent both the full economics of the deal (or optionally, a reference to it) and the workflow state the document instance is currently in.

Web services provide an ideal interface to services that operate on document instances given the support provided for structure, data typing, and platform neutrality. Support can be provided at all stages of process flow (see Figure 2).

Several services are outlined below, some of which focus on the theme of traversal, either locally within the FpML document instance, or remotely to related document instances, such as legal FpML document instances. Techniques for traversal are well established within the rich heritage of SGML, and by applications such as

Boeing Data Renaissance Suite 4, which are applied to critical systems. Such services allow all elements of the trade "package" to be traversed, most commonly for software automation, but also for the benefit of end users who may need to inspect legal definitions or applicable business rules.

Other services fit more directly into a conventional framework of interactive Web application technology, such as exposing services using JAX-RPC, to support interaction between components, both within and without the organization. These more typically operate on the document to perform a business action, resulting in a workflow state transition, as opposed to traversal services.

- **FpML for legal documents:** Allen & Overy submitted an initial proposal to ISDA to represent legal data in electronic form. This will make it possible to describe the complete nature of the relationship (legal, credit, collateral, and commercial) between participants. Currently the electronic text contained within FpML documents is illustrative, and dispute resolution occurs by reference to definitive hardcopy documents. Our objective is to reduce the need for human intervention to resolve definitions and reduce legal risk.
- **Business rule validation:** XML Schema provides support for syntactical validation of a document instance against the associated document prototype, but cannot ensure correct business sense is enforced for the large number of legal syntax combinations that now exist in FpML. The FpML Validation Working group has formed Business Validation Rules, which have been implemented using both XLinkIt and Schematron.
- **Workflow:** Earlier versions of FpML concentrated on describing the economics of a trade in a neutral context, which gave no sense of where the trade was in the workflow process. ISDA has now formed distinct

“Web services provide an ideal interface to services that operate on document instances”

document types, such as Trade Confirmation, which arise as part of a well-defined workflow and allow for the ready automation of workflow. ISDA/FpML is collaborating with FIX, with the focus from our side on performing a gap analysis of FIX business processes that are applicable to over-the-counter derivatives markets.

- **Confirmation matching:** The ISDA 2003 Operations Benchmarking Survey clearly highlights confirmations processing as one of the most important issues facing the derivative market today. It finds that "84% of equity derivative confirmations (are) sent out by T+5 ... reasons most often cited for confirmations not meeting their normal dispatch times are nonstandard language; new or nonstandard products; and delays in obtaining data or approval from front office, legal or compliance ... average number of confirmations that are outstanding ... increases generally with the sophistication of the product."

This highlights both the delay in sending the confirmation request out, and the subsequent problem of confirms being outstanding. FpML-based confirmation-matching services resolve these issues by providing a standard language, and allowing for either policy-based confirmation matching by each party, or provision of a central confirmation-matching service.

- **Valuations:** Vendors Gemsoup and Integrasoft have proposed the addition of valuation support to FpML. This proposal distinguishes between the market environment, the model environment, and the valuation result, and is an excellent example of the "package traversal" theme outlined above, where all involved parties are able to traverse to the part of the package of interest to them. The client of the Valuation service will wish to see both the valuation and how it was arrived at. To remove the problem of traversal to a "remote" part of the package, such as a yield curve the provider of the valuation is using, it is most likely that all these parts will be made "local" through inlining them into the valuation method. A fragment of a valuation is shown in Listing 1.

Summary

Going forward, implementers will exploit the natural fit between the migra-

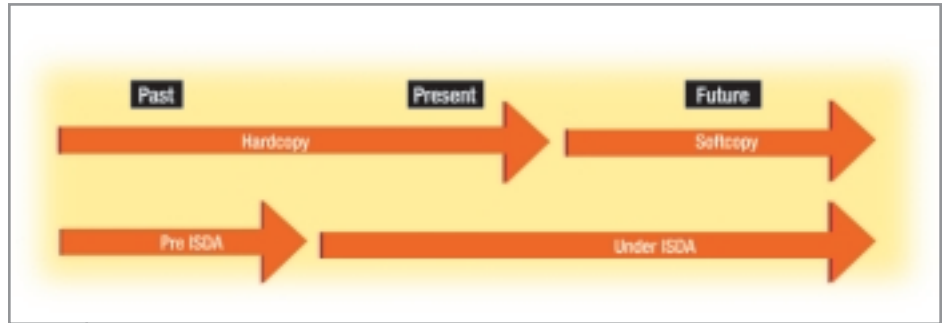


FIGURE 1 Adoption of FpML

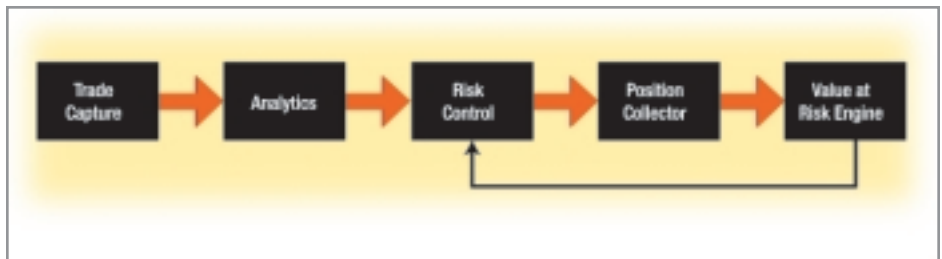


FIGURE 2 Web service support of document instances

tion to electronic documentation and Web services, with utility and value added as the main focus. Implementations will vary in scale between integration of legacy systems to an external service such as confirmation matching, through to full front-to-back support, providing seamless support for the full life cycle, regardless of where the services providing this support are located.

Given the wide range of network, software, and consultancy services already in the marketplace, it is possible to implement FpML-based systems at moderate cost, and without imposing a steep learning curve. The ISDA 2003 Operations Benchmarking Survey finds "large firms indicated that they had increased FpML use ... 33% of medium firms in this year's survey expect to increase the use of FpML during 2003." ISDA/FpML will continue to work closely with market participants to ensure that they capture the maximum benefit from automation.

References

- *Bank for International Statements Quarterly Review for June 2003*
www.bis.org/publ/r_qt0306.htm
- *International Swaps and Derivatives Association: www.isda.org/index.html*
- *Financial products Markup Language: www.fpml.org*
- *Boeing Quill 21: www.boeing.com/defense/space/aerospace/techdata/quill.htm*

- *FIX Protocol Organization: www.fixprotocol.org/cgi-bin/Workgroup.cgi?id=1057070654&menu=1057070654*
- *ISDA 2003 Operations Benchmarking Survey: www.isda.org/c_and_a/pdf/ISDA_Operations-Survey2003.pdf* @

About the Author

Andrew Parry is a project manager at Deutsche Bank, and chair of the FpML Equity Derivative Working Group. His current area of specialty is risk control and P&L attribution systems for business area controllers in equity derivative, convertible bond, and cash equity business lines. He holds a BA in economics from Manchester University and a masters degree in computer science from Sheffield Hallam University.

■■■ andrew.parry@db.com

Listing 1

```
<valuationReport id = "val-report
-01">
<portfolioRef>abc-port
-01</portfolioRef>
<valuation>
<receiverPartyReference href
= "abc"/>
<currency>GBP</currency>
<npv>1203490.00</npv>
<valDate>2003-06-04</valDate>
<npvSide>mid</npvSide>
</valuation>
<marketEnvironmentReference href
= "usdlabor"/>
<marketEnvironmentReference href
= "euribor"/>
</valuationReport>
```

Download the code at
sys-con.com/webservices

Axis-izing Legacy Applications

Two approaches that work

■ This article is a follow-up to an earlier article "Exposing Legacy Applications" (WSJ, Vol. 3, issue 5). It demonstrates how to integrate legacy applications with Web services using Apache Axis. Axis, which is a complete rewrite of Apache SOAP, promises to be faster and more flexible than Apache SOAP.

Two approaches will be illustrated here. The first uses the Axis API. In the second approach, the Axis tools (Java2WSDL, WSDL2Java) will be used to generate stubs. We then write a client that uses the generated stubs to access the Web service.

The legacy application used in this article captures and outputs system accounting information in an ASCII format.

Application Overview

The details of the architecture and description of the various components can be found in the first article. I had a legacy application in which I wanted to expose its commonly called operations as Web services. One of the uses of the application was to capture and track print job information for customer accounting purposes. It was based on a two-tier client/server model and the underlying communication protocol used was ONC-RPC. No changes were made to the existing legacy application. Converting the legacy application into a Web services application would allow a user to query these operations residing on a Web Server via a Web browser and have the results displayed to the user through the Web page.

To achieve this, a three-tier architecture was adopted. The legacy server resides on the third tier. The second tier contains both the Web server and the legacy client. A wrapper class was created to isolate the legacy client code, making it easier to



WRITTEN BY
ADELENE NG

maintain. The SOAP client lives on the first tier. It accesses the legacy client on the Web server through the wrapper class. SOAP is used to communicate between the SOAP client and the services exposed on the Web server. The services communicate with the legacy client through the wrapper class. ONC-RPC is the protocol used between the legacy client and server. Figure 1 shows the overall system architecture.

Figures 2 and 3 illustrate the call sequence between the SOAP client, Web server (containing the wrapper class and legacy client), and legacy server. The sequence diagram summarizes the interactions between the different tiers and clearly shows which platform each of the classes reside on.

Tools Used

The entire system was written in Java running on Windows 2000 Professional. The tools used to build this system are:

- *Java 2 Platform, Standard Edition (J2SE):* <http://java.sun.com/j2se/>
- *ONC-RPC for Java, a commercial ONC-RPC package:* www.distinct.com/
- *Apache Axis Release 1.1:* <http://ws.apache.org/axis/>
- *Apache Tomcat 4.1.24 (Servlet Engine):* <http://jakarta.apache.org/tomcat/tomcat-4.1-doc/index.html>

What Is Apache Axis?

Axis is an implementation of an XML-

SOAP-based protocol used for exchanging data in a distributed environment.

Several implementations of SOAP are available, for example, Apache SOAP (<http://ws.apache.org/soap/>), JWS DP (<http://java.sun.com/webservices/webserver-vicespack.html>), GLUE (www.themindetec-tric.com), WASP (www.systinet.com) – the best known of which is Apache SOAP. Think of Axis as Apache SOAP 3.0. It is a complete rewrite of Apache SOAP 2.2.

Differences Between Axis and Apache SOAP

Apache Axis supports SOAP 1.1 and 1.2, Web Services Description Language 1.1, XML Schema 1.0, and JAX-RPC whereas Apache SOAP supports only SOAP 1.1, with limited support for XML Schemas. It does not support WSDL and uses a proprietary API. For those of you who are unfamiliar with WSDL, it is an XML document that describes Web services. It specifies the location of the service and the operations the service exposes.

In addition, Axis supports both RPC/encoded and Doc/literal Message formats whereas Apache SOAP supports only the former. RPC/encoded messages follow both SOAP RPC and encoding rules. This means that the SOAP body contains an element with the name of the remote procedure to be invoked. This element in turn contains an element for each parameter for that remote procedure. Both the parameters and return results are encoded in the SOAP body. The "encoding" portion refers to a set of serialization rules defined in the SOAP 1.2 Specification. These specify how objects, structures, and arrays should be serialized.

Doc/literal does not use the SOAP encoding rules for data. Instead, data is serialized according to some schema usually expressed using the W3C XML Schema. Table 1 summarizes the differences between Axis and Apache SOAP.

Advantages over Apache SOAP

Axis offers the following advantages over Apache SOAP. These are:

- **Speed:** Axis is faster compared to Apache SOAP. This is achieved by using SAX internally rather than DOM.
- **Flexibility:** Developers can easily add extensions to the SOAP engine to allow

Web Services - Beyond Integration

by Bob Brauer

BOB.BRAUER@STRIKEIRON.COM



All new technologies claim to be “better, faster, cheaper”, but only those that provide value far beyond their original intent tend to make a real difference in our lives. Truly great technologies grow to create an exciting new future far beyond the initial vision of the inventors.

For example, James Watt first built the steam engine to pump water out of mines. Next it was used to power factories. Then came steam locomotives, which made it possible to move large quantities of raw materials from the mines to the factories. The end result was mass-production and the Industrial Age, all from a simple technology that started out as a water pump.

Then there's the Internet. It started as a way for researchers to share information among universities. Who could have guessed thirty years ago that this simple innovation would not only change the world but also give name to the age itself, the Internet Age.

Web services is another technology that will have far greater impact than the simple purposes for which it was invented. The first Web services were all about B2B integration. Application and systems integration currently represents the largest expenditure in most I.T. organizations. Reducing integration costs is a big benefit of Web services, but what happens when executives start to ask, “Now that we've invested in Web services to solve our integration problems, what else can we do with them?” With staff cuts and cost constraints holding I.T. spending at a standstill, it's time to really take advantage of the investment in Web services and move to the next level.

Companies are starting to turn to Web services to leverage the unprecedented amounts of information that are available across the Internet today. This will result in increased business collaboration that will revolutionize how we use the Internet. Soon we'll be able to call a Web service to do things that require entire software applications today.

Operations like authorizing a credit card, validating an address, booking a trip, or even calculating sales tax will be available as simple but powerful Web services.

Of course, there will be skeptics. There always are. Some will argue that there aren't enough external Web services available to make all of this a reality. Don't worry. They said the same thing about the telephone when it was invented and they were wrong then too.

So what do we need to make this transition? Several things, but first we simply have to reduce the complexity of Web services. The simpler Web services get the more people will use them for new applications. Soon people will stop talking about SOAP, WSDL, JAX, and DIME the same way that HTTP, SMTP, MIME, and DOS are no longer discussed but are instead taken for granted as layers on the stack.

Web services are already getting easier. Products like the Strikelron™ Web Services Analyzer are a major step forward in reducing the complexity and increasing the ease of use of Web services.

The Analyzer makes it much faster to create a new Web service and far easier to figure out what an existing Web service does. Instead of worrying about the complexities of SOAP notation and XML grammar, you can immediately visualize a Web service and start to use it, which for most of us is a far better use of our time.

What will the Web services world look like in five years? I don't know, but I certainly subscribe to the old adage that the best way to predict the future is to help create it.

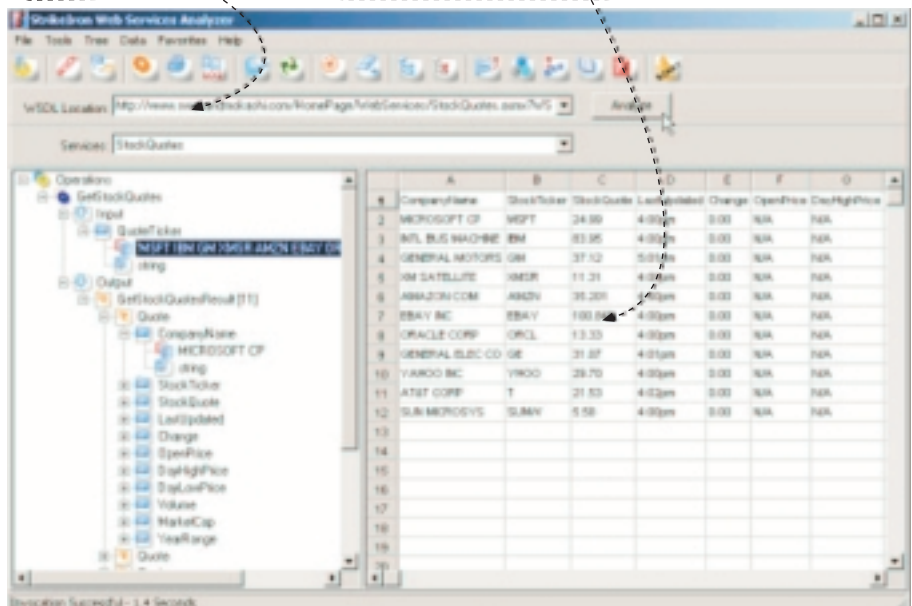
Bob Brauer is a Co-Founder of Strikelron and the visionary behind its Web services software products.

www.strikeiron.com
Delivering the World
of Web ServicesSM

Simply enter the WSDL to quickly execute and analyze any Web service.

View and save the resulting data from invoking a Web service directly to other applications.

Execute any Web service dynamically.



custom header processing or system management.

- **Instant deployment through the Java Web Service:** Note that this is intended for simple Web services only.

Points to Note When Migrating from Apache SOAP to Axis

Following are some of the items you need to be aware of when you attempt to migrate from Apache SOAP to Axis.

1. Axis does not return a generic Parameter type. The return type has to be explicitly set via the setReturnType() call.
2. The SOAPException class in Apache SOAP maps to AxisFault in Axis.
3. Although not used here, the MessageContext class is the Axis implementation of the Apache SOAP SOAPMessageContext class, and is core to message processing in handlers.

Building Web Services with Axis

Two approaches to building Web services with Axis are illustrated here. The first approach shows us how we can use the Axis API to build Web services. The second utilizes the tools that come with Axis to generate stubs. The client code is then written to utilize these stubs to access the Web service.

Using the Axis API

Create and Implement the "MyService" class

These are the Web service methods that will be called from a client. For our purposes, two methods will be exposed as Web services and are defined in the class MyService. The methods are:

- **public int GetNumberOfRecords_W(String filename):** Returns the number of records in the file specified as the argument
- **public String[] GetFileData_W(String filename):** Returns all the records in the file specified in the argument as an array of strings

Create the Web Service

Deployment Descriptor File

The WSDD is an XML file containing the deployment descriptor for statically configuring the Axis engine. This is proprietary to Axis. Axis uses the information contained in this deployment descriptor to track operations and type mappings. Note, this is different

from WSDL, which defines the custom XML types and methods exposed as Web services, the request and response pairs associated with each method, and the URL bound to this service. To create the file deploy.wsdd:

1. Specify the WSDD deployment and define the "java" namespace

```
<deployment
xmlns="http://xml.apache.org/axis/wsdd/"
xmlns:java="http://xml.apache.org/axis/wsdd/providers/java">
```

2. Define the service and provider name. The latter indicates it is a Java RPC service that is built into Axis.

```
<service name="MyService"
provider="java:RPC">
```

3. Specify the name of the Java class

```
<parameter name="className"
value="Accounting.MyService"/>
```

4. List all the methods callable from the client

```
<parameter name="allowedMethods"
value="*" />
```

5. The "*" allows all methods under the MyService class to be called. This can be further constrained by replacing the "*" with a space or comma-separated list of available method names:

```
<parameter name="allowedMethods"
value="GetNumberOfRecords_W
GetFileData_W"/>
```

Apache SOAP	Apache Axis
DOM	SAX Event Handler for better performance
SOAP 1.1	SOAP 1.1 and 1.2
Limited XML Schema Support	XML Schema 1.0
No WSDL support	WSDL 1.1
Proprietary API	JAX-RPC API
RPC/encoded only	RPC/encoded and Doc/literal
Low level API for headers	Easy handler support for headers

TABLE 1 Axis vs Apache SOAP

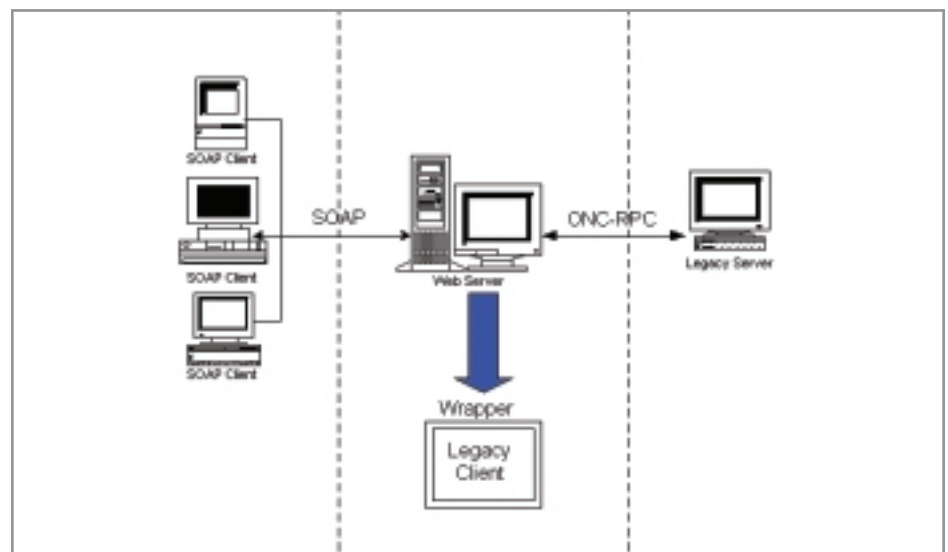


FIGURE 1 Overall system architecture

Axis can only send user-defined Java objects over the wire if there is a registered Axis serializer for it. The arbitrary Java classes have to be defined following the standard JavaBean convention. The BeanSerializer class is then used to serialize this class. The registration calls are made in your client code. Next, we need to tell Axis which Java classes map to which XML Schema type. This is done in the deployment descriptor file by adding the following line:

```
<beanMapping qName="ns:MyJavaClass"
xmlns:ns="urn:SomeService"
languageSpecificType="java:my.java.MyJavaClass" />
```

If the default bean serialization model is

insufficient to meet your application needs, Axis provides the ability to write your own custom serializers/deserializers. Once these are built, Axis needs to be told which types they should be mapped to. This is done by adding the typeMapping tag into the deployment descriptor file as follows:

```
<typeMapping qname="ns:MyJavaClass"
xmlns:ns="urn:SomeService"

languageSpecificType="java:my.java.MyJavaClass"

serializer="my.java.Serializer"

deserializer="my.java.DeserializerFactory"

encodingStyle="http://schemas.xmlsoap.org/
```

```
soap/encoding/" />
```

Writing the Client Code to Test the Web Service

The following steps are required to create the client code. The client accesses the Web services defined in the MyService class.

1. JAX-RPC Setup
 - Create a Service object

```
Service service = new Service();
```

- Create a Call object

```
Call call = (Call)service.createCall();
```

2. Set the URL for the Web service

```
String endpoint = "http://localhost:8080/axis/services/MyService";
call.setTargetEndpointAddress(new java.net.URL(endpoint));
```

3. Define the serializers/deserializers (if required) for the classes. Since the application does not need to transmit any application-defined data classes as part of the SOAP Requests, no serializers and deserializers are needed.
4. Set the name of the method that you will be calling. To associate the GetNumberOfRecords_W() method with the "call" instance, we invoke

```
call.setOperationName(new QName("MyService", "GetNumberOfRecords_W"));
```

Likewise, to associate the GetFileData_W() method with the "call" instance, we invoke

```
call.setOperationName(new QName("MyService", "GetFileData_W"));
```

5. Set the method arguments

```
call.addParameter("arg1", XMLType.XSD_STRING, ParameterMode.IN);
```

6. Set the return type of the method. To set the return type associated with the method GetNumberOfRecords_W(), we make the following call

```
call.setReturnType(org.apache.axis.encoding.XMLType.XSD_INT);
```

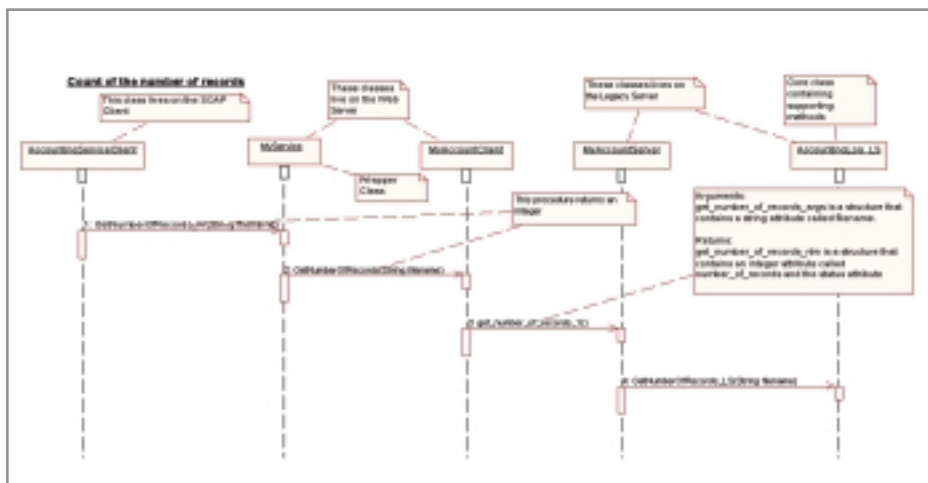


FIGURE 2 "Counting the number of Records" sequence diagram

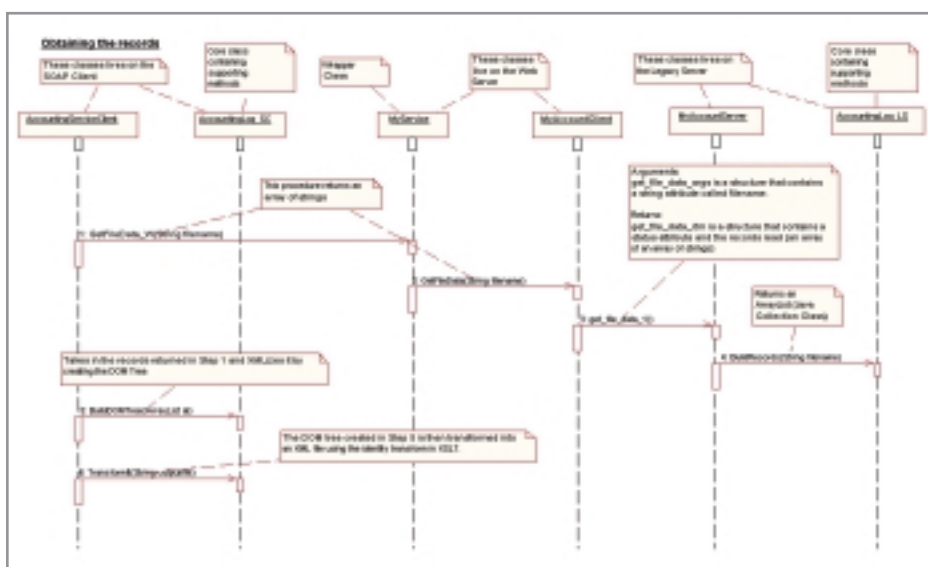


FIGURE 3 "Obtaining the Records" Sequence Diagram

Likewise, to set the return type associated with the method, `GetFileData_W()`, we make the following call,

```
call.setReturnType(new  
QName("ArrayOf_xsd_string"), java.lang.String[].class);
```

7. If the method name has been set to "GetNumberOfRecords_W", then it is invoked via

```
Integer ret = (Integer) call.invoke( new Object[] { s } );
```

"s" here refers to the argument that is being passed to the method that is being called. In this case, "s" represents the input filename.

8. Handle the errors

Deploying the Application

1. Compile all the classes
2. JAR the resulting class files and move the JAR file to the C:\jakarta-tomcat-4.1.24\webapps\axis\WEB-INF\lib directory.
3. Make sure that the distinct ONC-RPC JAR files are placed in the C:\jakarta-tomcat-4.1.24\webapps\axis\WEB-INF\lib directory.
4. Before deploying the service, make sure that your Web server is up and running. Here we are using the Apache Tomcat Server. It has several scripts under the C:\jakarta-tomcat-4.1.24\bin directory. To start the Apache Tomcat Server, go to the directory, C:\jakarta-tomcat-4.1.24\bin and type startup.
5. Deployment is done via the AdminClient tool,

```
java org.apache.axis.client.AdminClient deploy.wsdd
```

6. Next, run the legacy server application (if not already running).
7. Finally, run the SOAPClient.

To make things simple, I have created batch files for steps 1, 2, 4, 5, 6, and 7 to facilitate the execution of these commands (the source code for this article is online at www.sys-con.com/webservices/sourcecode.cfm).

Testing Whether the Web Service Is Correctly Deployed

There are two ways that you can check if your Web service is properly deployed. The first is to go directly to the Axis home page, <http://localhost:8080/axis/> (see Figure 4) and click on "View the list of deployed Web services". The page will show all the methods associated with MyService (see Figure 5). If the page only shows the heading, "And now...Some Services", then there is most likely an error in the `deploy.wsdd` file. To get a good idea of the cause of the error, look at the server log files (Apache Tomcat) located under C:\jakarta-tomcat-4.1.24\logs\localhost_log.<date>.txt

The second way is to directly invoke the service and the name of the method to test from the browser, for example,

```
http://localhost:8080/axis/services/MyService?method=GetNumberOf  
Records_W&s=C:\Experimental\Axis\src\ACCOUNT.LOG
```

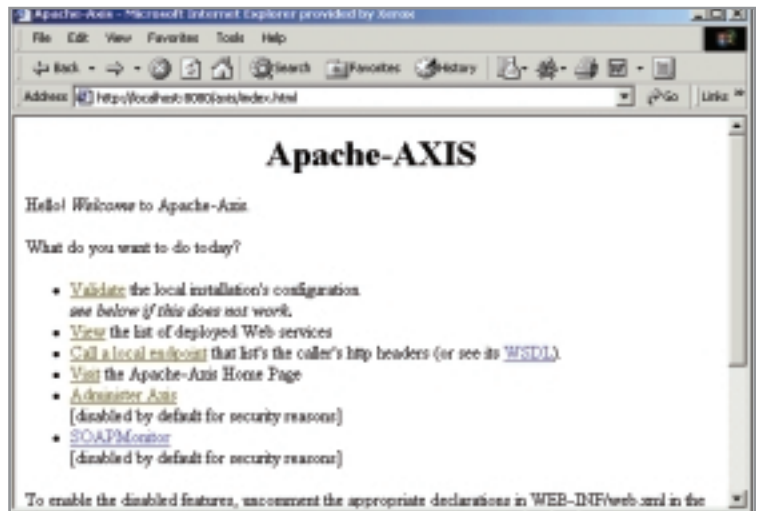


FIGURE 4 | Axis home page

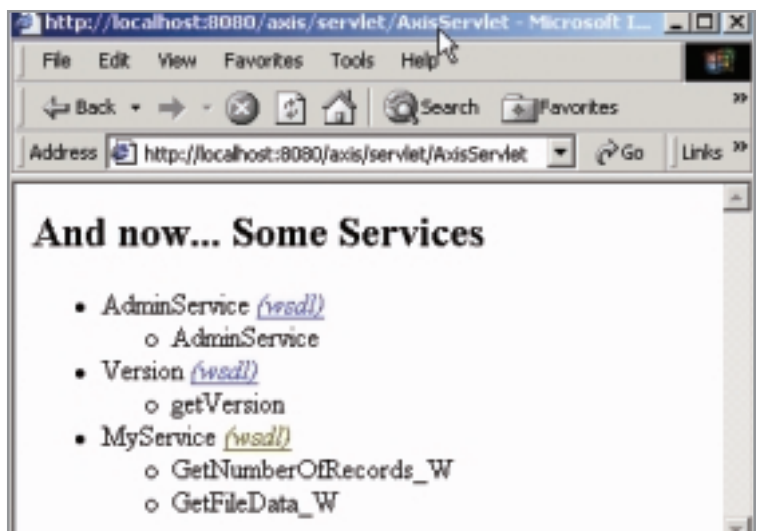


FIGURE 5 | List of deployed services

Figure 6 shows the result of invoking the service through the browser.

Using the Axis Tools

The second approach utilizes the tools that come bundled with Axis to generate the stubs.

Define the MyService and MyServiceImpl Classes

MyService is an interface class containing two methods. These are:

- **public int GetNumberOfRecords_W(String filename):** Returns the number of records in the file specified as the argument
- **public String[] GetFileData_W(String filename):** Returns all the records in the file specified in the argument as an array of strings

The MyServiceImpl class implements the methods defined in MyService.

Java2WSDL

The first step is to generate the WSDL file for the given MyService Interface. Use the Java2WSDL command to generate the WSDL file

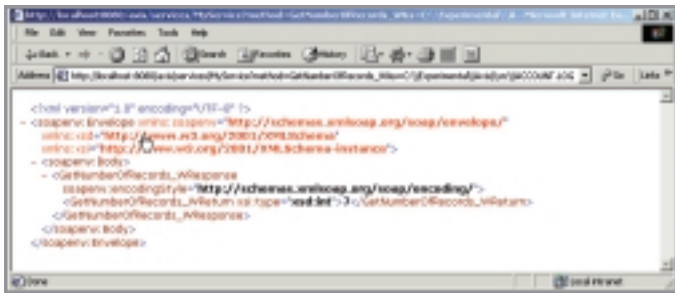


FIGURE 6 | Direct invocation of a service

```
java org.apache.axis.wsdl.Java2WSDL
-o Accounting.wsdl
-l"http://localhost:8080/axis/services/Accounting"
-n urn:Accounting
-p"Accounting" urn:Accounting
Accounting.MyService
```

Where

- o: Name of the output WSDL file, Accounting.wsdl
 - l: URL of the Web Service, http://localhost:8080/axis/services/Accounting
 - n: Target Namespace for the WSDL, urn:Accounting
 - p: Package to Namespace name,value pairs (Accounting = urn:Accounting)
- class-of-portType**: Fully qualified class (Accounting.MyService)

WSDL2Java

Next, generate the server-side wrapper code and stubs. The Axis tool, WSDL2Java, does all of this automatically when we run the following command.

```
java org.apache.axis.wsdl.WSDL2Java -o . -d Session -s -p
Accounting.ws Accounting.wsdl
```

Where

- o: Output directory for generated files
- d: Deployment Scope. This can be one of: Application, Request or Session
- s: Generate the server side bindings for Web Service
- p: Package – override all namespace to package mappings and use this package name instead

Name of the WSDL file: Accounting.wsdl (this was generated in the previous step).

The files shown in Table 2 are generated as a result of running the WSDL2Java command.

Modifying the AccountingSoapBindingImpl Class

Modify the AccountingSoapBindingImpl class as follows:

1. Add the following import statement after the package Accounting.ws statement:

Go to www.SYS-CON.com

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

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

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

Java Industry Newsletter

WebServices Industry Newsletter

XML Industry Newsletter

Wireless Industry Newsletter

WebLogic Industry Newsletter

WebSphere Industry Newsletter

GoldFusion Industry Newsletter

.NET Journal Industry Newsletter

FREE

E-Newsletters

SIGN UP

TODAY!

Generated Files	Description
AccountingSoapBindingImpl.java	Contains the implementation for the Web service. We need to edit this file to tie this to the existing MyServiceImpl.
AccountingSoapBindingStub.java	Client-side stub code encapsulating client access.
MyService.java	This defines the remote interface to our service that we wish to make available.
MyServiceService.java	This defines the Service interface of the Web service. Implemented by the MyService ServiceLocator class.
MyServiceServiceLocator.java	Helper class for retrieving the Service handle.
deploy.wsdd	Deployment descriptor to deploy the service.
undeploy.wsdd	Deployment descriptor to undeploy the service.

TABLE 2 | Generated files and their associated descriptions

```
import Accounting.MyServiceImpl;
```

2. Add the MyServiceImpl attribute and create an instance of the class:

```
MyServiceImpl ms = new MyServiceImpl();
```

3. Replace the generated return statement, "return -3;" in the method getNumberOfRecords_W() by:

```
return ms.GetNumberOfRecords_W(in0);
```

The argument, in0, is of type String and is the name of the file to be processed. Note that the return value of -3 is a value arbitrarily generated by the WSDL2Java tool.

4. Replace the generated return statement, "return null;" in the method

```
getFileData_W() by:  
return ms.GetNumberOfRecords_W(in0);
```

The argument, in0, is of type String and is the name of the file to be processed. Note that the return value of null is a value arbitrarily generated by the WSDL2Java tool.

Writing the Client Code to Test the Web Service

1. Create an MyServiceService object via the MyServiceServiceLocator:

```
Accounting.ws.MyServiceService  
service = new Accounting.ws.MyService  
ServiceLocator();
```

2. We use the MyServiceService object to get a remote handle to the service,

```
Accounting.ws.MyService acct =  
service.getAccounting();
```

3. Finally, we make a call to the method

```
acct.getNumberOfRecords_W("C:\\Experimenta  
l\\Axis\\src2\\ACCOUNT.LOG");
```

Deploying the Service

1. Compile all the classes
2. JAR the resulting class files and move the JAR file to the C:\jakarta-tomcat-4.1.24\webapps\axis\WEB-INF\lib\ directory.
3. Make sure that the distinct ONC-RPC JAR files are placed in the C:\jakarta-tomcat-4.1.24\webapps\axis\WEB-INF\lib directory.

4. Before deploying the service, make sure that your Web server is up and running. Here we are using the Apache Tomcat Server. It has several scripts under the C:\jakarta-tomcat-4.1.24\bin directory. To start the Apache Tomcat Server, go to the directory, C:\jakarta-tomcat-4.1.24\bin and type startup.

5. Deployment is done via the AdminClient tool

```
java org.apache.axis.client.AdminClient  
deploy.wsdd
```

6. Run the legacy server application (if not already running).

7. Run the client.

Conclusion

I have presented two approaches that show how we can integrate legacy applications with Axis. The first utilizes the Axis API. The second uses the tools that come bundled with Apache Axis. I have also summarized the differences between Axis and Apache SOAP, listed the advantages of using Axis, and brought up some salient points to note when migrating from Apache SOAP to Axis.

Acknowledgments

I would like to thank Tzu-Khiau Koh and Chui-Ching Low for their comments on the initial drafts of this paper. ☺

References

- Ng, Adelene. "Exposing Legacy Applications." *Web Services Journal*, Vol. 3, Issue 5. www.sys-con.com/webservices
- *RPC: Remote Procedure Call Protocol Specification Version 2*, RFC 1831, R. Srinivasan, August 1995: www.ietf.org/rfc/rfc1831.txt
- *XDR: External Data Representation Standard*, RFC 1832, R. Srinivasan, August 1995: www.ietf.org/rfc/rfc1832.txt
- *SOAP 1.2 Specification*: www.w3.org/TR/SOAP
- *Web Services Description Language (WSDL) 1.1 Specification*: <http://ws.apache.org/axis>
- *Java API for XML-Based RPC (JAX-RPC)*: <http://java.sun.com/xml/jaxrpc>
- *XML Schema*: www.w3.org/XML/Schema

About the Author

Adelene Ng is a software architect with Xerox Corporation based in Rochester, NY. She holds a Ph.D. in computer science from the University of London, and an M.Sc. from the University of Manchester.

■ ■ ■ adelene.ng@xssc.sgp.xerox.com



how many people does it take to change the web?

macromedia®
COLDFUSION®
MX

See for yourself. With the new ColdFusion MX, you can quickly build the new wave of Rich Internet Applications. Increase productivity with rapid server scripting and the new Macromedia MX development tools. Incorporate XML and web services with ease. Scale with a new Java™ architecture and deliver rich user interfaces with native connectivity to Macromedia Flash.™ You can even run your applications on leading application servers like IBM® WebSphere® and Sun® ONE. Try ColdFusion MX today and see what you can do now.

Download the free 30-day trial at
www.macromedia.com/go/cfmxad



Copyright © 2002 Macromedia, Inc. All rights reserved. Macromedia, the Macromedia logo, ColdFusion, Flash, and Macromedia Flash are trademarks or registered trademarks of Macromedia, Inc. in the United States and/or other countries. Other marks are the properties of their respective owners.

The Critical Need for Monitoring & Analysis

Remember why you chose
Web services in the
first place



■ The Web services paradigm is poised to become the dominant form of distributed computing this decade and beyond. Indeed, A. T. Kearney, an EDS global consultancy, found that 75% of companies ranging from less than \$50 million to more than \$1 billion in revenues and across 20 vertical industries have already deployed one or more Web service.

The Benefits of Web Services

Enterprise integration is consistently one of the top three priorities of IT organizations and accounts for \$50B in spending annually. Web services are becoming the technology of choice for solving pressing point-to-point integration needs because of the inherent benefits of the approach. Foremost among these benefits are cost reductions, business agility, and component reuse.

Cost Reductions

Reducing costs of an integration project as dramatically as 80% have gotten Web services a lot of attention. Merrill Lynch, DuPont, General Motors, Wachovia, NASDAQ, Continental Airlines, Dollar Rent-a-Car, JP Morgan Chase, and Zagat, to name a few, have used Web services for their recent application integration projects to drastically reduce their costs. In particular, Merrill Lynch was able to complete a project originally slated to cost



WRITTEN BY
JOTHY ROSENBERG

\$800,000 for \$30,000; that gets a CIO's attention. At Motorola, in one project involving the reuse of an existing credit-card validation function by "wrapping" it as a Web service, the company was able to save over \$100,000 in software development costs. The large cost benefit of Web services follows from how simply and rapidly they can be developed. In the past, enterprise integration required specialized – and sometimes downright esoteric – expertise. The programming skills necessary to use Web services for application integration are within the reach of the broad cadre of enterprise developers.

Agility – New Revenue

Cisco's John Chambers has said "the greatest contributions to corporate productivity, and consequently revenue, are likely to come out of dynamic changes to business processes." Business processes are hard to change when they are codified into inflexible monolithic applications. The component-based Web

services model offers the promise of business agility by enabling the rapid and dynamic configuration of information assets for the maximization of revenue.

The advantage of business agility is apparent in the case of MCI, which was able to transform its business several years ago with its MCI "Friends and Family" plan. The ability for MCI to use the lists of a particular customer's friends and family to provide pricing incentives gave it a valuable competitive edge. All the other long-distance carriers were caught flat-footed, unable to respond to MCI for 18 months because their assets were locked up in ossified systems that froze their business processes.

Reuse – Maximizing Leveraged Investments

When you build a Web service interface between two internal systems today, you can easily justify the cost savings for that point-to-point integration alone. But the future value is magnified each time the service-oriented architecture (SOA) created by your Web service is used by other Web service projects in other parts of the organization. Once the service is created it is likely you can reuse the same interface, or a slightly modified version of it, for different purposes later. This versatility is essential to a business's ability to deliver different combinations of products or services to new markets, win market share from competitors, and create new shareholder value.

Dollar Rent-a-Car needs to provide accurate, to-the-minute, location-specific car inventory data to multiple consumers: travel agent systems, airline systems, travel Web sites, and all the Dollar desks in airports. Each of these consumers leverages the same Web service connection to Dollar's legacy systems.

Web Services Evolution

Web services, like any other new technology, will go through several stages of evolution and adoption. In this section we'll examine how Web services usage will evolve over the next half-decade (see Figure 1).

First Phase: Tactical Deployment

The first phase of Web service adoption is characterized by widespread, but tactical, deployments. These typically involve exposing legacy data as a Web service for presentation in a portal or the creation of a point-to-point bridge between two different applications. Portals and Web services share a common goal: enabling a company's previous software investments to be combined and exploited easily in unanticipated, value-added ways. The second strong push for Web services is when companies are doing application integrations. Projects that are relatively low risk and represent "low hanging fruit" are typically chosen as initial Web service application integration projects.

Second Phase: Strategic Enterprise Middleware

For application integration, Web services are, without a doubt, the most compelling middleware technology we've ever seen. As first-phase deployments prove out the benefits, the complexity of Web services integrations will increase. Web services will start to become the middleware technology of choice for enterprises as they use it to integrate CRM, ERP, credit verification, and other customer-centric applications thereby lowering costs and providing better customer service.

As security standards mature in this second phase, enterprises will begin to more widely deploy Web services and attack high-value integrations with their trading partners. Indeed, even today a few innovative companies such as Dell, Dollar, Amazon, and Orbitz have aggressively deployed Web services to integrate supply chains or provide a richly integrated back end to an e-commerce application.

The Vision: Transformational Service-Oriented Architectures

In the third phase, Web services will become

ubiquitous and will form the fabric across which distributed enterprise application components communicate. All new application development and all packaged applications will expose their functionality via Web services. Applications will be formed by composing their functionality out of Web service components distributed across network and administrative domains. Services will be discovered and consumed in a dynamic fashion. Unlike previous attempts to deliver on

check or execution of a market sell order are vital components of a business process whose failure brings the entire business process to a halt. Others such as those that provide supplemental information like product information or weather details may be merely "nice to have."

Performance

The measurement of Web service performance has similar granularity requirements. The

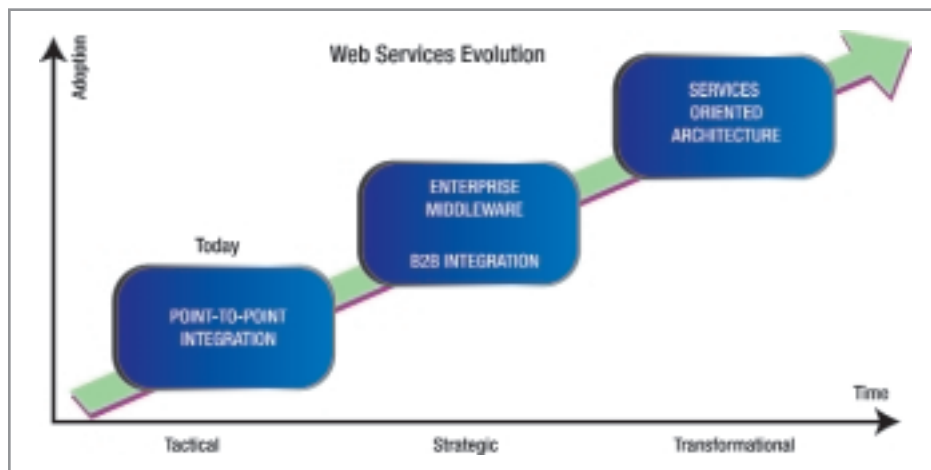


FIGURE 1 | Web services evolution

this vision, such as CORBA and DCOM, this time the "vision" is already showing some reality. It is already reality because four key areas never sufficiently addressed before are already solid and stable: an interface language (XML and WSDL), an information bus (the Internet), a flexible interface (XML documents), and a complete separation of interface from how applications are programmed (SOAP does not care if it talks to Java, C++, C#, or Cobol).

Operational Challenges Understanding Production Web Services

Production Web services operations have special requirements related to but extending beyond traditional operational metrics. These new operational challenges, which are also essential requirements, include reliability, performance, and security.

Reliability

The measurement of Web service reliability requires a finer level of granularity than the determination of application, system, or network reliability. All Web services are not created equal, and some are more critical to the operation of a business than others. For instance, key Web services that perform such functions as a credit

overall performance of a Web service is a relatively useless metric – you need visibility into the specific methods defined in a Web services WSDL interface description to understand whether observed delays might have a business impact. Method-level information is also necessary in order to isolate and optimize performance bottlenecks. Characterization of method performance may sometimes require visibility into the parameters associated with the method calls. These requirements mean the XML data stream must be observed; this poses a significant operational challenge itself.

Security

Security in the Web services world is an order of magnitude more complicated than that experienced with the Web browser. In the browser-based Web, transport layer security between two endpoints, such as that afforded by SSL, was all that was needed. Web services require the delegation of trust across domains (how do I know I can trust the entity that established your identity?) and monitoring whether security policies are being followed consistently and completely. Individual messages may need to be encrypted or signed or both in order to protect identity and security of participants and the information they are exchanging. The keys

for encryption and decryption and the identity of the owner of the keys is exceedingly complex when the Web service endpoints are not all within a single organization that can authenticate parties to transactions. Constant monitoring of an individual's authenticated identity and of confidentiality and integrity of documents is an absolute requirement for keeping a Web service in compliance with business and regulatory security policies.

Unique Challenges

Web services also present their own unique operational challenges. The sheer number of services and their methods combined with the granularity at which they must be monitored presents new data management challenges. Furthermore, the verbosity of XML means that a significant number of transactions results in a

these boundaries must be exchanged in order to maintain a handle on quality of service.

Limits of Established Technology

The previous generations of monitoring and management technology – specifically network, systems, and application management solutions – were architected to solve a different set of problems and do not apply to Web services.

Web Service Ignorance

Network management solutions focus on the layer of the application stack associated with the quality of network connectivity in a distributed environment; their primary concerns are the throughput and latency characteristics of networks. Systems management solutions focus primarily on the hardware layer of the stack and the health of the operating system. Application man-

dynamic change, with services being added and removed from applications in an organic fashion. Old-generation solutions are simply not agile enough to react to these changes.

Cross-Enterprise Myopia

The notion that all resources are controlled and owned by a central organization was presumed in the design of old-generation solutions. In the world of Web services, resources are spread across administrative domains, hence the systems that monitor and manage them will be under different spheres of control. Web services monitoring systems must be able to operate in such a federated management environment, sharing monitoring data as necessary to ensure proper quality of service.

Lack of Business Perspective

Web services are about enabling business processes. Through WSDL and SOAP, monitoring tools for the first time have a practical way of gaining visibility into the business context of messages. Older-generation solutions are not focused on the business information contained in the XML streams. Instead, they remain focused on system metrics and so cannot provide a business perspective on the systems they manage and monitor.

The New Architecture for Monitoring and Analysis Meeting Operational Challenges

New monitoring systems are needed to meet the operational challenges brought on when deploying Web services. These systems should be easy to administer, provide powerful monitoring and visibility capabilities at fine granularity, and allow flexible ad hoc queries about the content and business value inside the XML messages. Furthermore, they must have very low overhead so as not to detrimentally impact the performance of the Web services they are monitoring. Systems to monitor Web services should be “good Web services citizens,” by which we mean that they should adhere to the same principles that drive people to adopt Web services in the first place. Hence, these systems should be cost-effective, lightweight, and nonintrusive to implement; standards-based; and compatible with all other systems and tools with which they might need to interact.

Cost-Effective and Nonintrusive

To be cost-effective, solutions must be easy to acquire and deploy. They must be affordable and scale with the size of your

“Web services are becoming the technology of choice today for solving pressing point-to-point integration needs”

large volume of data that must be monitored in order to provide operational visibility.

Moving to a service-oriented model means that services will be shared among different applications. This sharing, while introducing a cost benefit from reuse, results in more new operational requirements. You need to track how much resource is being consumed by the various service requestors. For example, there is the potential for one service requestor to overload the system, causing unacceptable performance for other consumers of the same service.

Multi-tier service infrastructures introduce the possibility of a cascade effect in the case of the failure of a Web service on whose data a service several hops away may rely. The ability to understand and trace the cause of failures across a Web service infrastructure becomes very important. Cross-enterprise usage of Web services requires a level of visibility into resources that are outside of your control. As resources are shared across organizational boundaries, monitoring data across

agement solutions focus on the specific server processes involved in serving an application and characterizing the availability and performance of application interfaces. None of these solutions understands the notion of a Web service, let alone a method call or element within a message. The quantities they are able to characterize are at a different layer in the stack and hence cannot provide the granularity necessary to be effective at the Web services layer.

Monolithic and Inflexible

Previous generation solutions were also designed around a static IT infrastructure. These kinds of framework solutions often take several months to initially deploy and several more months to customize to specific customer needs. Because the managed systems were static and the interfaces between systems was proprietary, correct functioning was defined as “the server is up and the expected process is running.” Web services, on the other hand, are about agility and



Where
Open Minds
Meet



CONFERENCE: January 20 – 23, 2004

EXPO: January 21 – 23, 2004

THE JAVITS CENTER: New York, NY

LinuxWorld Conference & Expo is the world's leading and most comprehensive event focusing on Linux and open source solutions. This January, discover how companies across the globe have achieved higher profits and increased their productivity by utilizing Linux, the fastest growing operating system in the world.

- Get in-depth coverage on the latest Linux and open source developments and learn about innovative product solutions from the industry's top companies.
- Network with the leaders in the open source movement and discuss with your peers how to best leverage the technology for your organization.
- Participate in LinuxWorld's world-class education program and benefit from interactive training in the all-new Hands-on Labs!
- Attend the 2nd annual Linux Financial Summit sponsored by BEA, and hear how Wall Street firms are leveraging Linux to achieve a lower total cost of ownership.
- Discover real world practical solutions and find answers to today's most critical IT issues.

Cornerstone Sponsor

ORACLE®

Platinum Sponsors



Register online with reference code: A-LMN

www.linuxworldexpo.com

IDG
WORLD EXPO

deployment. They should reduce the total cost of ownership (TCO) of your entire Web services infrastructure. The solution should require no additional coding or changes to network topology while providing full visibility into the HTTP, SOAP, and XML stream data. You should take care to avoid solutions that add new components into the architecture that could introduce single points of failure and costly additional hardware, software, or administrative requirements.

Flexible

Agility is the point of Web services so you need flexibility built into your monitoring system. The ideal system provides a “what if” level of control and flexibility, enabling near real-time monitoring and analysis of the application. The “what if” kinds of questions that will be asked about Web services will change constantly based on the changing roles of the observer (operations, business, executive) and changing business conditions. The system must have the ability to specify what operations, what performance metrics, what alerts, and what business metrics are of interest. These alerts must be translatable into events that provide actionable information through interfaces with established network or systems management tools.

Standards Based

Why develop an entire middleware strategy around Web services that is platform neutral, lightweight, and standards-based and throw that out the window with monitoring that creates a vendor-specific lock-in? The basic design principle of the monitoring system should be that it too is built out of Web services standards. This makes it work in heterogeneous environments, makes it easy to interface to other WS-compatible tools and systems, and means all investments are protected.



Advanced Requirements

The basic requirements outlined above are the necessary minimum for any Web services monitoring solution. But to be truly effective and successful, the solution must also comply with a set of more advanced requirements. It must be very efficient and not impose any significant overhead on the Web service communications. It must be easy to administer even as the service scales to hundreds and thousands of distributed nodes. And it must provide business insight because critical business data that can give powerful, early insights to business trends are available through the XML Web service streams.

Efficient

To get visibility into the entire XML stream for ad hoc queries that can bring business value to light, and to do this with little or no performance impact, it is essential that the monitoring system have very low overhead. This requires a focus on monitoring, not on processing or transformation of the SOAP envelopes and payloads being observed. Acceptable latency penalties for stream processing should be on the order of a millisecond, not tens or hundreds of milliseconds as is typical for tools on the market today.

Easy to Administer

The ideal monitoring system introduces no additional burden for the operations staff tasked with ensuring their successful operation. Installation should be completely automatic, simple, and foolproof. Once installed, the system should automatically discover all running Web services and immediately provide value by logging and monitoring baseline characteristics about the Web services.

Business Insight

Business users need the ability to see and audit their organization's compliance with regulations that could have a major impact on the business. They will want to track business transactions and have the ability to ask questions about those transactions. Aggregating XML tags across sessions and across servers into coherent business transactions will be required to enable this. Just as operations staff wants alerts on functionality errors or performance issues, business users will need to set up notifications based on business conditions or

thresholds critical to their business goals. Charts, graphs, and reports showing business usage and trends which provide insights that allow business analysts to have agility will prove invaluable to an organization's top and bottom lines.

Call to Action

There is no longer any doubt that Web services will become ubiquitous quickly. There is also no doubt that Web services will create a plethora of unmet operational challenges. Visibility of middleware that integrates applications is essential to business operational success. Web services are a completely new approach to middleware and, not surprisingly, new architectural approaches and practices are needed for successful monitoring as well.

If you have already deployed Web services in your production environment, you need to ensure the operational health and availability of these assets with Web services monitoring and analysis software. Even if you are just getting started with Web services, it's not too early to start building monitoring and analysis into your best practices. Don't wait until you feel the pain from putting your business-critical Web services into production naked.

As you begin to evaluate Web services monitoring software, keep in mind the reasons you have chosen to use Web services in the first place: lower TCO, greater investment leverage, and no vendor lock-in. Your Web services monitoring and analysis solution should not only preserve those benefits, but strengthen them. The design center for a truly effective monitoring and analysis system must be consistent with central Web service tenets – it must be cost-effective, flexible, nonintrusive, and standards based. The best architectures will have all these characteristics and be efficient, easy to administer, and provide significant business insight in addition to operational metrics. ☺

About the Author

Dr. Jothy Rosenberg is a serial entrepreneur and the founder, director, and CEO of Service Integrity. Prior to this venture, Jothy cofounded GeoTrust, the world's second largest certificate authority and a major innovator in enterprise managed security solutions. He is also the author of *How Debuggers Work* and *Understanding Web Services Security* (2003; Addison-Wesley). Jothy holds several patents.

■ ■ ■ jothy@serviceintegrity.com

Open IT.



The future of technology is wide Open.

Open source technologies like Linux, PHP, and MySQL have changed the way companies think about IT. Now it's time to deliver on the promise. You've got to make the right choices for your business, from the operating system on up, and make open technologies play well with the systems already in place.

7 themes that are transforming IT:

As new ideas and technologies arise throughout the IT industry, seven key themes have emerged that will change the way you do business:

- Open Source and Linux
- Wireless and Mobility
- The Digital Enterprise
- Web Services
- Windows Platform
- On-Demand Computing
- Security

Get the big picture. At COMDEX, you'll see all the options side-by-side in a single vendor-neutral environment. Top-tier conference sessions, tutorials, keynotes, and Innovation Centers will show you how to make the latest in IT relevant to your business needs.

See it all at COMDEX. Hear what's coming next from an elite lineup of speakers from Amazon.com, AT&T Wireless, Computer Associates, Hewlett-Packard, Intel, Microsoft, Nokia, Siebel, Sun Microsystems, Symantec, and more. Share ideas with industry visionaries, business leaders, and thousands of your peers. And experience the vitality of an \$870 billion industry being reborn.

Visit COMDEX.com for details.

**Save up to \$200.
Register by October 17.**

Use Priority Code: **ADBDZ4NC**

Las Vegas
Convention Center

November 17-20, 2003
Educational Programs: November 16-20, 2003

COMDEX®
LAS VEGAS 2003

THE GLOBAL TECHNOLOGY MARKETPLACE

Copyright © 2003 MediaLive International, Inc., 795 Folsom Street, San Francisco, CA 94103. All Rights Reserved. MediaLive International, COMDEX, and associated design marks and logos are trademarks or service marks owned or used under license by MediaLive International, Inc., and may be registered in the United States and other countries. Other names mentioned may be trademarks or service marks of their respective owners.

 **MediaLive**
INTERNATIONAL

Glue 4.1 from The Mind Electric



A platform that sticks

■ What do you do after you've cofounded a company that developed award-winning products for distributed computing, won a Young Entrepreneur of the Year award, and finished a book on Web services? If you're Graham Glass you start another company and continue pushing the envelope of distributed computing.

The company is The Mind Electric (TME) and the envelope is one of simplicity. TME has tried to build a Web services platform with a simple conceptual model and an easy-to-understand API. It is working on a next-generation product they're calling a "Web services fabric" code named GAIA but this review focuses on their current platform, Glue 4.1.

Today APIs are proliferating like rabbits and frameworks are getting larger and more complex. When I wear my developer hat, I prefer to work at the API level. It helps to promote an understanding of the mechanics of the system, but it gets harder and harder to keep up. Graphical tools, like wizards, can hide the complexity of large frameworks, but after the initial code-generation phase is done most of these tools are not useful or tend to get in the way. With the ever-increasing complexity of application servers, Glue takes a welcome step in the right direction, the direction of simplicity.

If you're looking to put your toe in the water of Web services but recoil from the complexity of working with a full-blown J2EE application server, then Glue may be the product for you.

Flavors of Glue

There are two versions of Glue: Standard, which is free but unsupported; and



WRITTEN BY
PAUL MAURER

Professional. Although Glue Standard is unsupported by TME, there is an interest group available on Yahoo! that gives you access to the TME user community at large.

Glue Professional comes with support via TME's online issue tracker and includes these high-end features:

- Turbocharged performance
- Real-time management console
- Web service instrumentation
- EJB integration
- Reliable/async messaging over JMS
- LDAP authentication via JAAS
- WS-security, including digital signatures and encryption
- WS-routing
- Remote deployment
- Virtual services
- IDE plug-ins

Getting Started

You can download Glue from the TME Web site. Installation was quick and simple. I was pleasantly surprised to find that the installer automatically set up my environment. I could get down to running through the examples without additional twiddling. This may seem trivial, but you would be surprised how many platforms require additional setup and configuration (e.g., proper CLASSPATH setting) after the installer has run.

Glue comes with a full complement of API documentation in Javadoc format and a Users Guide. The installer also sets up links to the online TME interest group and issue tracker.

The Simplest of Services

Getting a service up and running in Glue is incredibly easy. Any Java object can become a service by simply publishing an instance of the object to the Glue registry. There is no pre-processing or configuration required.

Listing 1 (the code is online at www.sys-con.com/webservices/sourcecode.com) shows a basic object called Converter that converts temperature in Fahrenheit to Celsius. It requires only two lines of code to expose this object as a Web service. Line 25 starts a Web server that accepts messages via the /glue path. Line 26 exports the object as a Web service. Other than the import statements at lines 4 and 5, that's all that's required.

By default, Glue exports all public, static, and instance methods of the object. If I want to expose only a subset of the methods I can create a Java interface that defines the set of methods I choose to expose. The interface.class is published along with the object and Glue exports the methods of the interface. If I don't have control of the source code of the object, I can control method exposure by supplying a Context object that contains a list of method properties.

A Simple Client

Writing a Web service client using Glue is also easy. Listing 2 shows a test client for the Converter service. Line 11 binds to the service and returns an object that implements the pub-



Company Info

The Mind Electric
15455 Dallas Parkway
Millennium Building, 6th Floor
Addison, TX 75001

Phone: (972) 764-5115

Fax: (972) 764-3215

Web: <http://www.themindelectric.com>

E-mail: info@themindelectric.com

Licensing Information

Developer License: \$1000 per developer

Runtime License: \$2000 per CPU

Testing Environment

OS: Windows XP Professional (Service Pack 1)

Hardware: Intel Pentium III - 996 MHz - 512 MB RAM

lished interface. The service can now be treated as a standard Java object.

There are a few things in this example that need further explanation. The first parameter of the bind method is the URL of the WSDL for the Web service. To obtain the WSDL for the Web service I just need to append .wsdl to the service URL. Glue will generate the WSDL automatically from the interface and cache it for future use. The second argument of the bind is the class of the interface for the service. I didn't define an interface for my Web service, but Glue can generate one from the service definition with its wsdl2java tool.

This example may seem overly simplistic but TME provides a large assortment of examples that exercise a wide range of Glue's features including support for .NET.

Advanced Features

You may worry that a tool that provides such a simple interface would not offer the power and flexibility that a typical large enterprise would need. Glue has more advanced features than most would need.

If you want to monitor or modify the message stream, Glue provides a feature called Interceptors. Interceptors are leveraged throughout Glue to provide advanced processing of SOAP headers and attachments.

Glue supports SOAP over the Java Message Service (JMS). Both synchronous and asynchronous messaging styles are supported and services can be published simultaneously over JMS and HTTP.

For those who insist on only using Java standard APIs, Glue provides implementations of JAX-RPC and JAXM.

Hard-core techies can completely bypass the use of Java interfaces and Java/XML serialization. Glue provides the ability to directly create, invoke, and process raw SOAP messages.

Finally, for high-volume systems Glue can reduce the size of SOAP messages by utilizing optimizations like tag substitution, envelope omission, and HREF inlining.

Interoperability

Although Glue runs easily out of the box as a stand-alone lightweight application server, it

can plug into any application server that supports servlets. In this hosted mode, Glue can expose any stateless session bean as a Web service. The Users Guide provides instructions for installing glue under BEA WebLogic, WebSphere, and SunONE application servers.

If you're an IDE jockey you'll be happy to find plug-ins for JBuilder, Eclipse, and IDEA.

Conclusion

I was definitely happy with the speed at which I could install Glue and get a scratch Web service up and running. But I was really impressed when I began to dig in and found a depth of features I had not expected from a platform that strived for simplicity. I've always liked the adage, "Good tools make simple tasks easy and complex tasks possible." I think that adage sticks with Glue. ©

About the Author

Paul Maurer is a principal in the financial services practice of a leading consulting services company.

■ ■ ■ paul@paulmaurer.net

SHOP ONLINE AT **JDJSTORE.COM** FOR BEST PRICES OR CALL YOUR ORDER IN AT **1-888-303-5282**

BUY THOUSANDS
OF PRODUCTS AT
GUARANTEED
LOWEST PRICES!

GUARANTEED BEST PRICES

FOR ALL YOUR
WEB SERVICES
SOFTWARE NEEDS



N-ARY

\$299.00

n-ary Ticket System v2.0

This installable application is a clean-cut Web-based tool that enables you to log & track important information. It can be used as an intranet or extranet product, giving your customers access to see how their issue is progressing. A unique number ID is assigned to every ticket. The system is extremely flexible and simple to use and can be utilized in numerous different situations. With more features than before, The Ticket System can be used for any number of great applications, which you can tailor as much or as little as you like.



ZION SOFTWARE

\$995.00

JAlerts' Deployment (JMessageServer)

If you need to send real-time Instant Messaging Alerts and Notifications to your employees or customers over the public IM services such as AIM, MSN, ICQ and Yahoo than JAlerts' is your solution.



GREENPOINT INC.

\$1,350.00

WEB CHARTS 3D VER. 4.7

WebCharts 3D is a state-of-the-art visualization package designed for the professional Web developer. It provides general purpose and specialized 2- and 3-dimensional charts, grids, and heat maps that can be delivered as server-generated interactive images (PNG, GIF, JPG, SWF, SVG, WBMP) or applets to browsers and mobile devices.



JALERTS

\$1,495.00

JAlerts Deployment (ICQ)

If you need to send real-time Instant Messaging Alerts and Notifications to your employees or customers over the public IM services such as AIM, MSN, ICQ and Yahoo than JAlerts' is your solution. Based on Zion Software's popular JBuddy SDK technology, JAlerts' provides an even simpler solution for sending real-time messages.



CHUTNEY TECHNOLOGIES

\$755.25

Chutney SOAP+ Toolkit

The Chutney SOAP+ Toolkit is the industry's first Web services monitoring and optimization toolkit. The Toolkit fills the functionality gaps in the leading SOAP development libraries by providing the ability to accurately pinpoint Web services bottlenecks and eliminate them through optimization techniques such as caching. The Chutney SOAP+ Toolkit acts purely as a supplement to these libraries, so no changes to the existing application logic are required. Flexible in its feature set, the Toolkit provides value to applications serving as either Web service consumers or providers.



INFRAGISTICS

\$495.00

Net Advantage Suite 2003 Volume 2

The Most Comprehensive Collection of Components for All Microsoft Development Environments. The ONLY fully integrated suite you'll ever need to create the most flexible, advanced applications for any Microsoft environment.



W W W . J D J S T O R E . C O M

OFFERS SUBJECT TO CHANGE WITHOUT NOTICE

Web Services in the Insurance Industry

A dynamic industry takes the lead

■ The mere mention of insurance seems to have an effect on people. Bring up property or liability or health insurance and they'll probably talk about how much they pay or what claims experience they've had. Discuss life insurance and they'll change the subject or run away. Bring up reinsurance and responses might range from "huh?" to an impolite yawn.

Despite the perceptions, or misperceptions, the insurance industry is a dynamic and far-reaching business. And for good reasons, insurance companies and related software vendors usually embrace the latest technology. The insurance industry is data-centric and thrives on new technologies and communication methodologies. That's why Web services combined with insurance data standards will bring new efficiencies and better experiences for parties up and down the global insurance value chain – from risk management and underwriting to service and reinsurance.

A major challenge faced by all branches of the insurance industry is governmental regulation and the corresponding reporting of data. Insurance is highly regulated throughout the world and on various bureaucratic levels from local to national to international. Reporting to those regulators is less burdensome when utilizing data standards and creating a common ground. Moreover, when you consider that different lines of business — workers' compensation or auto or liability for instance — require different customer data sets and thus forms, it's easy to understand why data standards are so critical for expedient communication and reporting to the appropriate governmental bodies.

Based in Pearl River, NY, the Association for Cooperative Operations Research and Development (ACORD) was founded in 1970 as a not-for-profit, industry-owned entity to



WRITTEN BY
AZIZ HUSSEIN

develop and provide standard forms for insurers and agents. Once focused on EDI standards for the U.S. domestic insurance industry, ACORD now has a global reach — as the needs of members have grown. Standards are critical in helping to move information from the point of sale across borders and back again when claims are settled. Along the way multiple parties are touching all or part of that data. In recent years ACORD developed XML standards in insurance, serving software and hardware vendors, agents, brokers, insurers, and reinsurers. (Reinsurers sell insurance to insurance companies to help protect them against excessive losses.)

Today, insurance software vendors must program changes, update new forms, code everything, test it, get it implemented, and make sure agents and brokers are installing the newest version of software — and by the time that happens there might be a dozen new forms and dozens of changes in the old ones. With a Web services interface, updates are instantaneous. It will be much easier for insurance firms to comply with government regulations, and all parties can be assured they'll be using the most up-to-date ACORD forms, reducing errors. Furthermore, agent and broker technology system vendors won't need to focus on writing code or testing it. Instead, they can rely on what ACORD provides to be functional and timely, which helps them comply with various governmental regulations. And they'll be able to free up

programmers to develop other product enhancements.

Standard E-Forms Being Developed

ACORD has developed more than 500 different forms that professionals in the insurance industry use daily. The bulk of these forms are used by tens of thousands of independent insurance agents and brokers — local business people who deal with clients every day — to handle insurance applications and provide services after sale. ACORD also establishes and promotes standards used to communicate that information through the data flow of member companies. Software providers incorporate these standards into their products and clients use them in this context. When agents enter data on forms through their agency management system, the data is captured in text form and transmitted to the carrier electronically. The insurer then transfers that data to its own policy administration system, thereby eliminating re-keying.

The insurance industry is data-centric. From policies to statistics to financials, the insurance world lives by the information flow. There are no widgets or tangibles in the insurance world, only the culling and collecting, management, and sharing of information throughout the data chain. Since Web services are also information-centric, the marriage of these worlds is a natural. Web services, supported by XML intelligence behind the data, enables quicker and more accurate data flows regardless of systems and platforms used. The ultimate goal is to foster system integration both internally and externally and improve efficiency and communication.

ACORD is migrating and merging its work on forms and XML standards into the Web services arena, creating smart electronic forms — e-forms. The first step is to take established static forms and make them fillable with an intelligent XML back end. This is not new technology, but it is an improvement and will serve as a template for even greater efficiencies down the road. Agents using these e-forms can enter data directly on their PC instead of having to print and fill out blank forms manually. The e-forms can be completed online and then saved, e-mailed, faxed, or printed. They can also be downloaded blank and completed later. The first of these new forms is ACORD's online fillable Certificate of Liability Insurance e-form. Certificates are among the industry's most heavily used items

Register Now!

bea.com/dev2devdays2003

or call

1-925-287-5156

1 DAY

20 CITIES

500 MINUTES OF TRAINING

1,000s OF DEVELOPERS

ALL THE CODE YOU CAN HANDLE

And not a single iota of hype.

BEA dev2dev Days is the worldwide, one-of-a-kind, no-nonsense seminar that gives you the tools you need to solve today's toughest development and integration challenges.

All in a single day. With no hype. And no spin. Just the facts.

You want code? We got code. Code-level demonstrations, in fact, on everything from architecting to building to integrating enterprise solutions on BEA's industry-leading WebLogic® 8.1 platform.

At the seminar, you'll learn information and techniques to help you build Web-based and service-oriented applications; work with XML and Web services; integrate with enterprise resources and applications; and much more.

But you have to act fast. A BEA dev2dev Days seminar is coming soon to a city near you, and spaces are limited. The cost to register is US\$169.00.

BEA dev2dev DAYS 2003

BY DEVELOPERS, FOR DEVELOPERS

Register Now!

bea.com/dev2devdays2003

or call

1-925-287-5156

SEPTEMBER – NOVEMBER 2003

COMING TO A CITY NEAR YOU:

Amsterdam
Atlanta
Bangalore
Beijing
Boston
Chicago
Dallas
Denver
London
Madrid
Mexico City
Munich
New York
Paris
San Francisco
Seoul
Stockholm
Tokyo
Toronto
Washington, D.C.

BROUGHT TO YOU BY:

dev2dev

attachmate

Confluent Software

FTP
FAMCETTE
TECHNICAL
PUBLICATIONS



intel



SYS-CON
MEDIA

and are often reused or edited. ACORD expects to have many e-forms available on its Web site by early 2004.

Flexibility is another benefit with e-forms. By being able to receive data in XML format, participants along the entire processing chain can make better use of it than if it had come in just as plain text. For instance, they can reuse it, link it to other forms, or integrate it into either a back-end system, a local database, or elsewhere.

E-forms will have intelligence built in. They'll help guide users as they fill in data. For instance, a form might catch something as simple as an invalid date, such as June 31.

Software Vendors Getting Aggressive

Major software vendors have announced several strategic partnerships with ACORD around Web services in support of its Advantage Program for agents and brokers. (Advantage provides agents and brokers with unlimited access to ACORD forms, along with tools and other services to help improve workflow and business operations.) The partnerships include:

- Microsoft has announced it will use its new InfoPath tool to design forms with an XML back end. For those thousands of independent agents and brokers who participate in ACORD's Advantage Program, Microsoft will offer in its Office 11 suite of products the InfoPath tool to access the more than 500 fillable ACORD forms from the ACORD Web site.
- ACORD Advantage participants may choose to use the new 6.0 release of Adobe Reader, with an XML platform, for viewing and filling ACORD forms.
- IBM will offer ACORD XML forms in their XForms, an initiative originating with the W3C. It's another option for accessing and leveraging ACORD forms, and demonstrates ACORD's versatility by mapping XML to a variety of standards.
- Victoria, BC-based PureEdge Solutions Inc., is developing ACORD-standard XML forms using PureEdge technology. The forms will become more dynamic and offer more intelligence in order to seamlessly manage and automate the business process associated with the forms.

Insurers Investing in Web Services

While ACORD is now preparing for Web

services, how is the technology playing out so far in the insurance industry? Spending on Web services is up, but the payback isn't here yet according to technology analyst Celent. But Celent estimates that spending on Web services in integration projects by insurers will increase nearly 10-fold between now and 2006. That's because Web services address a key business problem for insurers – reducing the burden of systems integration for both internal and external systems, Celent points out.

Celent estimates that in 2002, U.S. insurers spent approximately \$1.4 billion on internal and external integration for new IT projects alone – roughly 7% of overall spending on IT. Celent estimates that approximately 5% of current integration spending, or roughly \$78 million, is being spent on Web services-related initiatives today.

By 2006, more than 40% of new project systems integration spending will involve Web services, or roughly \$740 million, Celent says. Here are some examples of current insurer projects:

- Allstate used Web services to integrate back-end systems with a new agent portal, creating reusable components and enabling a rapid system deployment.
- State Auto provided communications between internal systems, reducing internal integration costs and laying the groundwork for partner integration.
- Aon Surety provided a cost-effective way to transmit surety bond information to underwriters, letting them handle small transactions profitably.
- EMC Insurance Companies provided as-needed information to its glass replacement provider instead of requiring them to sort through a massive database of customer information.
- Lincoln Financial Group syndicated content and functionality seamlessly into its partners' Web sites.

Insurance Software Vendors Leverage Web Services

The new smart e-forms will also create opportunities for insurance software providers to develop a broad array of applications that can integrate with agents, and brokers' technology systems and streamline the workflow for carriers.

Several major insurance-specific software developers are leveraging Web services

capabilities. One vendor is using ACORD XML to enable real-time data exchange between an agency's system and carriers' back-end systems, public domain Web applications, and third-party software applications, such as rating or credit reporting software. (A prospect will be able to fill out an online application from an agency's Web site, and the architecture will allow that data to be imported into the agency's management system while business rules ensure the data's integrity and security – and eliminating data re-entry.)

That same software provider is letting agents exchange data between their technology system database and third-party software vendors, such as credit reporting firms, based on ACORD XML, Microsoft .NET, and Web services technology.

Another vendor is using a .NET environment and Web services to link to workstations in the agency's office and the agency's data on hosted database servers, allowing the agency to take advantage of third-party services. Another vendor released a Web service for any agent or broker for single-entry sign-on to multiple carriers' agent portal sites, and does direct-bill inquiry from one screen to 15 insurers. It also integrated with a third-party ASP that handles marketing campaigns for that agency's customers and prospects.

Conclusion

Certainly, as development matures around Web services, challenges will arise. Achieving the ultimate Web services, where everyone works together, will require diligence in pushing for even greater standardization. We all must examine how to handle different types of XML, for instance. Security will also be an important issue to work through. But ACORD and its members are working together with vendors and the industry at large to address these and other issues and establish standards, which will ease these burdens.

Regardless of how quickly the insurance industry moves on Web services, ACORD continues to focus on its mission of providing standards for efficient and clear communication for all of its members. ☺

About the Author

Aziz Hussein is chief technology officer of ACORD, based in Pearl River, New York. For more information, visit www.acord.org.

■ ■ ■ ahussein@acord.org

International Conference & Expo

Edge 2004 EAST

Development Technologies Exchange

February 24-26, 2004

Hynes Convention Center, Boston, MA

**ARCHITECTING JAVA, .NET, WEB SERVICES,
OPEN SOURCE, AND XML**

Addressing:

- ▶ Application Integration
- ▶ Desktop Java
- ▶ Mobility
- ▶ ASP.NET
- ▶ VS.NET
- ▶ JavaServer Faces
- ▶ SOA
- ▶ Interoperability
- ▶ Java Gaming

Register By
November 21, 2003

Up To
SAVE
\$400



For more information visit
www.sys-con.com
or call
201 802-3069

Over 200 participating companies will display and demonstrate over 500 developer products and solutions.

Over 3,000 Systems Integrators, System Architects, Developers, and Project Managers will attend the conference expo.

Over 100 of the latest sessions on training, certifications, seminars, case-studies, and panel discussions promise to deliver real world benefits, the industry pulse and proven strategies.

Full Day Tutorials Targeting

- Java • .NET • XML
- Web Services • Open Source

Conference program available online!
www.sys-con.com/edgeeast2004

Contact information: 201 802-3069 • events@sys-con.com



Using Web Services for Business

■ SOAP is at the heart of all Web services as the way to deliver messages between two applications or systems. SOAP in its various versions is well known and often discussed. However, SOAP on its own is often not enough, especially if the message is sent outside the enterprise where privacy of the message content and reliable message delivery become much more important.

Even security and reliability may not be sufficient. You may also need to limit the life of a message so that it “expires” at some point, or specify the type of long-running business transaction of which the message is part.

Efforts are under way to provide these and other useful features in various SOAP standards efforts.

Why Do These Standards Matter?

Standards matter because standards lead to better solutions at lower cost, but why?

First, it's not easy to build solutions to problems such as reliably sending a message over the Internet or making sure that someone can't pretend to be you when sending a SOAP message over the Web.

Solving these problems requires a lot of effort from knowledgeable individuals, and even if you could develop your own solutions to these problems, you'd still need to



WRITTEN BY
DAVID BURDETT

persuade your partners to implement technology to support it.

On the other hand, if a solution to these problems has been developed in a standards group, then not only will you probably have the leading experts in the world involved, but you should have a “better solution.”

Multiple vendors developing a standard should also result in multiple vendors offering solutions. So you should get lower costs because of the competition.

Finally, if “good” solutions are available from multiple vendors, then you don't need to build them. It also means that your partners will be implementing compatible solutions that lower their costs so that you can connect to them easier and faster.

So What Are We Waiting For?

Sadly, standards take time to develop – especially if they're to be “done right.” You also need to know which standards to implement;

there are so many and they're still evolving.

Which brings me to the purpose of this article, which describes all of the features that are, or will be, needed to use SOAP to “deliver the message” for linking businesses. It then describes the leading standards that are being developed to support those features under four headings: Message Packaging, Metadata, Addressing, Reliable Delivery and Security (see Figure 1) and finally suggests an architectural approach to solving the problem of multiple evolving standards. Note that the rest of this article references many different standards initiatives. Brief descriptions and links to these initiatives are available online at www.sys-con.com/webservices/sourcec.cfm.

Message Packaging: How to Include Data Inside a Message

Packaging describes how all the different parts of a SOAP message are assembled into the string of characters that is sent “over the wire.” Similar concepts are needed as in the real world. This means you need to define the electronic envelope into which the main content of the message is packed – the equivalent of the envelope used by logistics companies such as UPS or FedEx; how the message is transported over the Net and errors reported – the equivalent of the delivery systems run by postal services and logistics companies; and, how to handle multiple documents at the same time – for example, if you wanted

to send an electronic book and an electronic delivery note in one SOAP message.

Standardization of message packaging has progressed well with SOAP as the definitive electronic envelope. Version 1.1 of SOAP has been widely deployed, with SOAP version 1.2 published in final form in summer 2003. Basically, SOAP standardization is nearly complete. The SOAP specifications also provide detailed descriptions of the “delivery system,” (i.e. how to send and receive a SOAP message over HTTP).

Handling multiple documents still needs work. “SOAP with Attachments” is in widest use although it is not part of any standardization process. The Direct Internet Message Encapsulation (DIME) specification developed to handle attachments by Microsoft did not receive widespread support and seems to have been abandoned. The most likely replacement will be the SOAP Message Transmission Optimization Mechanism (SMTOM) specification being developed in the W3C, although the first draft was published only last summer.

At the same time, the Web Services Interoperability Organization (WS-I) published a profile in August 2003 that defines how SOAP 1.1, WSDL 1.1, and UDDI 2.0 should be used together. WS-I also conducts interoperability testing to help assure that solutions from different vendors work together.

Metadata: Useful Data About a Message

Metadata is additional information about a message that is often needed to process the message correctly. It covers such things as identifying a message with a unique ID; relating (or correlating) one message to another, such as relating a request and its reply; linking together all the messages in a long-running transaction, such as relating the messages containing an order with the related messages containing the delivery note and invoice; the type of long-running business transaction being followed, such as should a supplier that receives an order reply with an invoice or should they send an order response that indicates whether they can satisfy the order; and finally agreement information that identifies the agreement or rules that the sender and receiver of the message have agreed to follow when exchanging messages.

Several specifications include how to identify and correlate messages, for example the WS Reliable Messaging specification developed in OASIS; the WS MessageData specification developed by BEA; and the WS Addressing specification developed by IBM, Microsoft, and BEA. The last two specifications have intellectual property restrictions on their use and have not been submitted to a standards group.

Independent of this, Sun, Oracle, and others have developed the WS-Context specification. It describes how to do correlation and relate the messages in long-running transactions. This specification has been submitted to OASIS.

route that a message should follow in order to get from its origin to its destination.

Standardization of Addressing has not made as much progress as most implementations assume that where you send a SOAP message is defined in the WSDL definition for the Web service and it's always fixed. This means that where you send a message to is always worked out at design time and then hard-coded into the resulting program.

This is fine if you're using Web services to build applications, but if you want to use Web services for delivering business messages it does not work as well since addresses change – businesses are bought, or new systems are implemented. With a



FIGURE 1 Required features

However, no standards activities currently exist that identify the type of business transaction being followed or how to identify any agreement that is being used.

Addressing: Specifying the Destination for a Message

Addresses on messages serve a similar purpose to addresses on letters. For example, they can be used to check that the message has been delivered to the correct recipient; work out how to route a message to its correct destination; and identify alternative methods of delivery if the original method did not work.

Addresses can be of several types, including logical addresses that specify who or what should receive a message without specifying its physical location; physical addresses that specify exactly where the message should be sent rather than who should receive it; return addresses that specify where replies should be sent and address routes that specify the precise

physical address, this requires that the programs change; on the other hand, if a logical address is used, then it can be mapped to the actual physical address at runtime.

There are some specifications that include addressing. These include the WS Reliability specification, which specifies SOAP Headers that identify the logical address and physical address for the message. The WS Addressing specification also provides SOAP Headers that identify logical addresses and physical addresses as well as a return address. The WS Callback specification also defines a return address. The WS Routing specification from Microsoft defines how to route a message through one or more intermediary SOAP processors.

Only WS-Reliability is being developed in a standards group.

Reliable Delivery: How to Ensure a Message Gets There

Reliable Delivery includes the features that make sure a message is delivered and

“ Digital signatures, although they don't help with privacy, can ensure that the message has not changed even after it has been received and its contents unpacked... ”

processed properly. It includes Delivery Receipts where a recipient of a SOAP message sends back another SOAP message confirming the original message was delivered; Reliable Messaging, where the sender of the SOAP message tries to make sure the message is delivered by resending it if they don't receive an acknowledgment back from the recipient; sequencing, which includes sequence numbers in messages to make sure they are delivered to the application in the same sequence the messages were sent – even if they arrive out of sequence; Expiry, which is a way of indicating in a message that it should not be processed if it is received after a certain time; and two-phase commit, which is based on distributed database ideas and includes many of the ideas of Reliable Messaging but also describes what to do to reverse the effect of changes if the message was not delivered.

Standardization of Reliable Delivery is making good progress. For example, the WS Reliability specification provides an acknowledgement message that can be used as a Delivery Receipt. Delivery Receipts are also provided by the WS Reliable Messaging and WS Acknowledgement specifications. The WS Reliability and WS Reliable Messaging specifications also provide support for reliable messaging and message sequencing.

Expiry of messages is supported by the WS Reliability specification.

Two-phase commit is supported by the Business Transaction Protocol (BTP) activity within OASIS and the similar WS Transaction specification developed by IBM, Microsoft, and BEA. Two-phase Commit is also an idea supported by the Web Services Transaction Management (WS-TXM) specification developed by Arjuna, Fujitsu, IONA, Oracle, and Sun.

What's interesting is that the WS Reliability and WS Reliable Messaging specifications are almost identical – the first is being developed in OASIS, the second is not in any standard

group. Of the other specifications, only the BTP and WS-TXM specifications are being developed in standards groups.

Security: How to Secure the Information in a Message

Message security covers ensuring the privacy of the message content, knowing who sent the message, and making sure messages aren't altered after they're sent. It includes transport security, digital signatures, message encryption, authentication, authorization, and key management.

Transport security uses a communication protocol such as SSL to put an “opaque pipe” around the SOAP message so no one can see what's flowing or alter the message while it is in the pipe without detection. However, once the message pops out the end of the pipe, all privacy is gone and the message can be changed at will.

Digital signatures, although they don't help with privacy, can ensure that the message has not changed even after it has been received and its contents unpacked. They work in much the same way as signing a letter helps make sure that the recipient knows the letter has not changed.

Message encryption goes further than transport security and digital signatures by encrypting the message itself in much the same way that “spies” code their messages. This means that even if you can see the message you can't understand it unless you have the correct “key” to decode it. Also, if

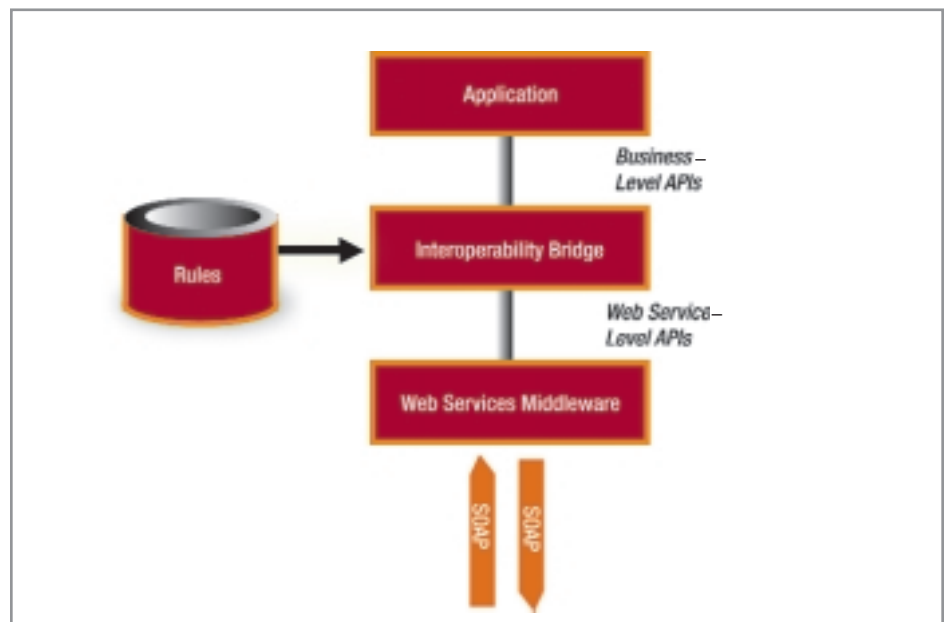


FIGURE 2 Managing the problems

you can successfully decode or decrypt the SOAP message then you also know it was unaltered as any alterations would lead to a decryption failure.

Even if you receive a SOAP message that is digitally signed and encrypted, it doesn't automatically tell you who or what created and sent the message. Authentication takes the next step by checking the digital signature on a message to make sure it is genuine, or for encryption, checking who gave you the "keys" that allowed you to decrypt the message.

Authorization goes the final step and checks not only who the sender of the SOAP message was but also what the authority of the sender is to make the request that receiving the message implies. For example, a driver's license can be used to identify someone (authentication), but it also indicates that the owner is "authorized" to drive.

A lot of progress has been made on the development of standards for security. Secure Sockets Layer (SSL) is "the" standard for transport security and the XML Signature and XML Encryption standards are mature, stable specifications that can be used to digitally sign and encrypt any XML document including a SOAP message. However, they don't specify exactly how to sign or encrypt a SOAP message so the WS-Security initiative in OASIS is specifying how to do this as well as how to authenticate the sender of a SOAP message.

The Liberty Alliance is an initiative that is specifying how to determine the identity of businesses and individuals so that they can then be used to authenticate SOAP messages.

Trust and key management is one of the hardest areas to address, although some specifications, such as WS Trust, exist for establishing trust relationships; WS Secure

Also, the intellectual property restrictions on some standards mean that any user should be wary of committing to those standards in anything more than an experimental way. The recent lawsuits raised by SCO around Linux illustrate that what might appear to be "legally safe" in fact is not. Even if these suits are resolved in favor of Linux, the uncertainty makes the use of IP-constrained standards a risk.

Adopting the Right Architectural Approach


However, waiting for standards to mature will mean missing the benefits of "better lower cost solutions" that were described earlier.

Fortunately, adopting the "right" architectural approach to the solutions you build or buy makes managing the problems easier (see Figure 2).

First, your Web services architecture should have three levels: the application that contains the business logic; the Web services middleware that understands Web services technology; and in between, a rules-driven "interoperability bridge."

The key component is the interoperability bridge as it takes as input a "business-level" request to send a message that specifies the data to be sent and who (not where) the data should go to.

The interoperability bridge can then look up rules to determine what standards the destination wants and the physical address the message should be sent to. It then uses the Web service-level APIs to invoke the Web services middleware that does the actual sending and receiving of the message. A similar approach can be followed for the return path.

This approach separates the business application from the Web services standards so that each can evolve relatively independently. It also means that when changes occur they can be applied in one place – the interoperability bridge – at lower cost and more consistently. 

About the Author

David Burdett is director of Standards Strategy, Web Services for Commerce One. He is responsible for directing Commerce One's use of Web services standards. He is also a consultant with over 20 years' experience in managing and motivating teams in IT strategy, development, architecture, and organization.

■■■david.burdett@commerceone.com

“Gaps in standards and competing standards mean that it will probably be some time before stable standards are established”

With a driver's license, you trust that the authority that issued it checked that the holder is who they say they are. But this only works if you know what a driver's license looks like, who issued it, and whether or not you trust the issuer. For example, after 9/11 it was discovered that some of the terrorists had falsely obtained Virginia driver's licenses with only minimal checks. Similarly, when a SOAP message is received over the Web, you need to have received, in advance, copies of the codes and certificates from a source you trust, that allow you to check the validity of the signatures and decrypt the message.

This distribution of certificates and codes in advance is known as "key management" and often ultimately requires that copies of the certificates and codes be sent out-of-band using separate security management solutions.

Conversation for establishing a secure context for exchanging SOAP messages; and WS Federation for brokering trust between organizations. However, none of these specifications have been submitted to a standards group although all the other security-related specifications have.

What You Should Do Next

The description of the SOAP features needed and the current state of standards development shows that not all the standards needed are under development, for some areas there are competing standards, and finally, some of the specifications have been published outside of a standards group and with intellectual property restrictions.

Gaps in standards and competing standards mean that it will probably be some time before stable standards are established.

Demystifying Service-Oriented Architecture

Take the right approach

■ With the emergence of Web services into the mainstream the developer has to learn how to architect and build service-oriented systems. While service orientation isn't a new concept, the rapid convergence of the industry on Web services technology has brought the concept of service-oriented architectures (SOA) to the forefront of many developers' minds.

Over the last decade we learned how to construct software systems using patterns that adhered to the concepts of object orientation. Now, service orientation requires us to adapt to a new approach to system integration and application development. However, at the moment most of us are still learning about this new technology and so we tend to apply familiar patterns when building Web services-based applications.

Applying object-oriented patterns to service-based systems is generally a poor idea since the scale of a typical object-oriented application (generally within a single enterprise or department) is dwarfed by the scale of a Web services-based application (which may span many enterprises and departments). It is crucial to remember that the use of SOAP and WSDL in our applications does not constitute service orientation.

This article outlines the basics of (Web) service-oriented architectures and looks at the distinguishing features of SOA that make it the right approach to take when developing Web services-based applications. We define the concept of a service, compare the emerging service-oriented programming to object-oriented programming, and outline how to advance to SOA.



WRITTEN BY
JIM WEBBER &



**SAVAS
PARASTATIDIS**

Finally, we explore these concepts in the context of an example banking system.

What Is a Service?

In order to understand SOA, we must first understand what an individual service is. A *service* is a logical manifestation of some physical resources (like databases, programs, devices, or humans) grouped as a process that an organization exposes to the network. A Web service is therefore best thought of as a "user interface for computers" and, just like GUIs for human consumption, a Web service provides controlled access to some underlying computer system (see Figure 1).

A Web service is the set of actions that the hosting enterprise is prepared to execute. A Web service does not expose a set of operations, methods, or functions (though WSDL uses similar names). Instead, it advertises the set of structured messages that it will exchange. When a message is received by a Web service, the appropriate action is performed and a response message may be returned. There is no assumption that subsequent messages received are associated with prior ones (i.e., Web services are stateless), nor are services mutually dependent – the fact that we concentrate on message exchanges, rather than interfaces, supports this notion.

Service Orientation Versus Object Orientation

Perhaps the most important difference between service orientation and object orientation resides in the way software integration is achieved. Fundamental to object orientation is the concept of a type system that is shared among software components. In the general case, a hierarchy of interfaces has to be agreed upon in advance, before those software components can be integrated.

In contrast, services and their consumers achieve integration through the exchange of messages. The only thing that is shared between them is the vocabulary used to define the structure of those messages, which – in the case of Web services – are XML and XML Schema. The absence of a predefined type system enables loose coupling since no information needs to be shared among services and consumers, before implementation and deployment. Service orientation is more appropriate when (potentially heterogeneous) platforms are to be interconnected.

Furthermore, in object orientation, software entities (objects) are smaller units of abstraction than services (e.g., an account versus a bank). Hence, fine-grained interactions are encouraged between these objects and the formation of complicated webs of interrelations and dependencies between them (e.g., objects may hold references to other objects) cannot be avoided. Conversely, service orientation promotes coarse-grained interactions and the reduction of dependencies between services. In this view, we propagate "chunky" messages around a Web services-based application, which contains all the necessary information to invoke an action. Good service-oriented design mandates that all the information necessary to invoke an action is contained in a message, unlike object orientation where "chatty" interactions between client and serving object are the norm. Since Web services are likely to be remote from the application that consumes them, fine-grained interactions will perform poorly because of the overhead imposed by network communication (and indeed the additional XML infrastructure needed), and so it is imperative that coarse-grained interactions are supported. Once we have decomposed a problem domain into coarse-grained message exchanges

#1 Circulation in the World!

*JDJ is the highest circulation
i-Technology magazine in the world!*

162,019*



**Java
Developer's
Journal**

120,031*



Dr. Dobb's

75,561*



MSDN

*JUNE 2003 BPA AUDIT STATEMENTS

*All circulation numbers are publishers' most current own data, six-month average circulation through June 2003.



Carmen Gonzalez
Senior VP Marketing



Miles Silverman
VP Marketing



supported by schemas to describe common vocabularies we are ready to step up to service-oriented architecture.

Stepping Up to SOA

The transition from object orientation to service orientation is not necessarily straightforward, because of inertia more than any real complexities of the approach. As developers and architects that have learned and used object-oriented patterns in distributed application design for more than a decade, we cannot expect to adapt overnight to this new architectural paradigm. Indeed, in the current world of Web services we see examples of old approaches to distributed application development presented as service orientation.

For example, early SOAP toolkits unwittingly encouraged an object view of the world by making SOAP RPC extremely easy to use, an approach that has been deprecated by the community through the WS-I basic profile. Current SOAP toolkits make it very easy to create a one-to-one association between a Web service and the implementing object (e.g., expose EJBs as Web services or create a WSDL of a Web service from a single class). A Web service should be more than that; it should be a front end to a set of objects, programs, databases, humans, etc. A particular symptom of this mindset is the use of the term “operation” in WSDL to describe the set of messages that are related to a particular action of a Web service. This is unhelpful since it has the same kind of connotation as terms like “method” and “function”, where it is better to think in terms of messages. Thinking in terms of messages not only forces designers to think in terms of coarser granularities, but also detracts from the problematic notion of shared type systems where messages, not typed interfaces, are shared between components.

Stemming from this mindset, it is not uncommon to see Web services technologies used to violate enterprise encapsulation by exposing internal resources to the Web service consumers. This inevitably results in an integration nightmare and fragile applications where easily broken interrelationships can be formed that cross enterprise boundaries. We have seen examples of this in particular application domains, like the grid, where Web services are used to support characteristics like

stateful interactions, service transience, factories, publicly visible state, and introspection, which are borrowed from object-based systems like CORBA and J2EE and applied to Web services in a manner that carries the danger of tight coupling. Effectively, a common component model is built on top of Web services when one is not required. Component models are ideal for building closed systems but are not suitable for enterprise-to-enterprise integration. The dangers of layering an object model on top of Web services technology are shown in Figure 2.

The Bank Service Example

The best way of demonstrating these concepts is by an example taken from our day-to-day lives, the “banking service.” A banking service is an abstraction of a set of actions that are offered by the bank to authorized users. We may be able to open/close accounts, deposit/withdraw money, get/repay loans, transfer money from one bank to another, pay/cash checks, establish financial agreements, finance a business, and so on.

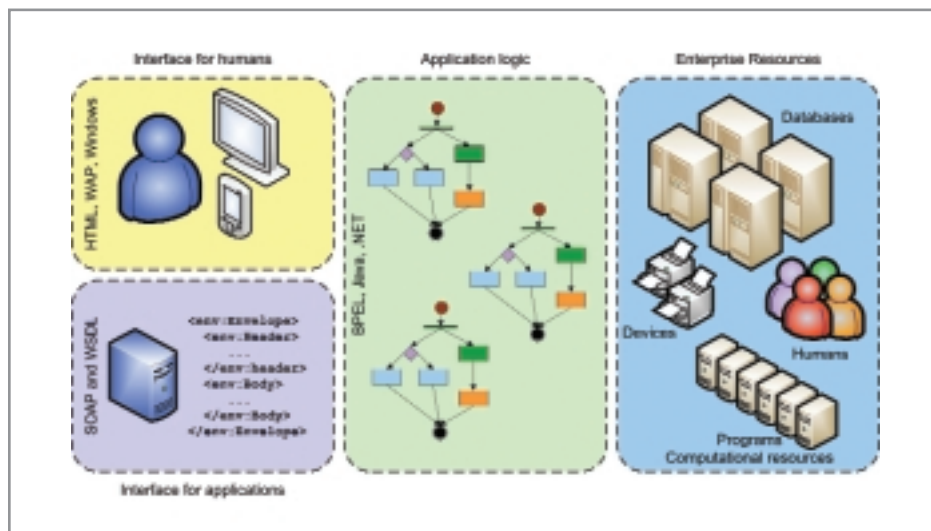


FIGURE 1 Managing the problems

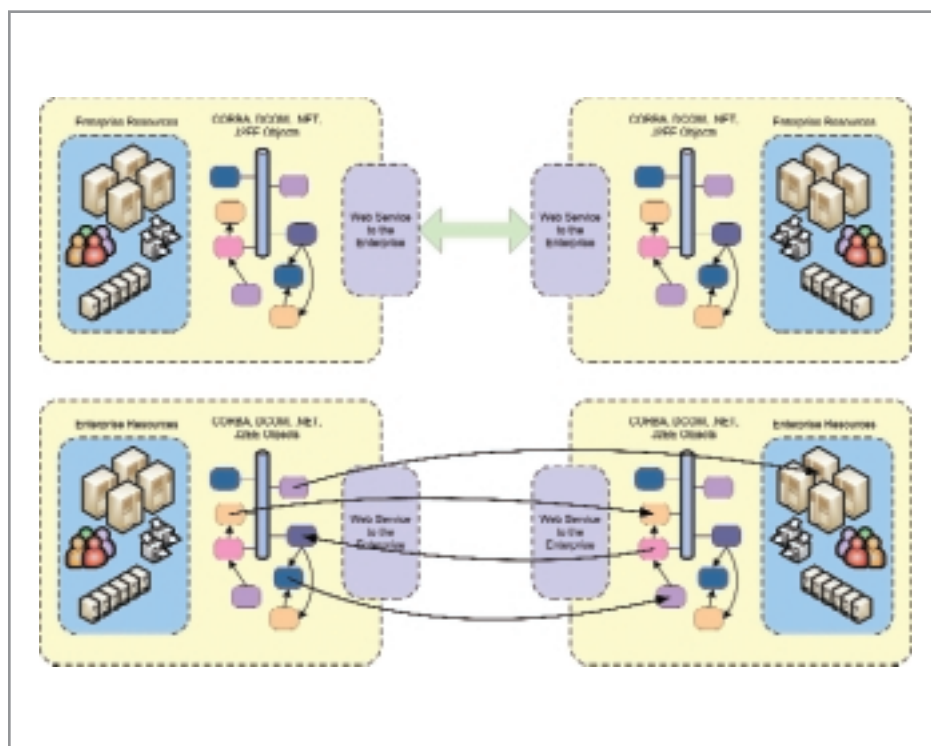


FIGURE 2 Poor Practice: Layering object models on top of services

As customers of the bank, we don't have to see any of the underlying technologies, methodologies, resources, and practices that the bank uses in offering its services. We are free to use any of the available interfaces to access the same offerings albeit at different levels of quality of service. For example, we can use our computer at home, the telephone from work, one of the many ATMs on a street, or even walk into any of the branches of the bank whose service we want to access, as depicted in Figure 3. In all of these cases, the business logic is controlled by the banking service. We are merely presented with a human-to-service interface.

When interacting with our banking service through one of these interfaces we use a common problem domain vocabulary like the terms "account" and "money" or the actions "deposit" and "borrow" as the basis for message exchanges with the bank.

For example, when we want to withdraw money from our account, we are not presented with the actual account "object"

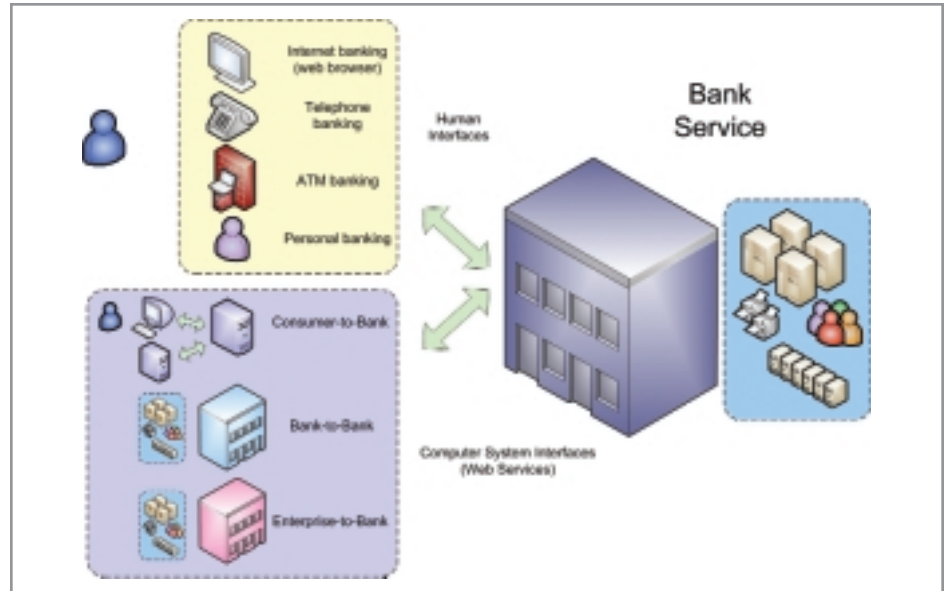


FIGURE 3 | A banking service and its human and computer interfaces

and asked to withdraw the money via a "method call" on that object. We don't know, or need to know, how the information about our account is maintained by the bank. Instead, we interact with the

bank through message exchanges that result in some money being withdrawn.

Taking this approach one step further, we can consider message exchange patterns provisioned through Web services

Service Interactions

Some interaction patterns are more sophisticated than those offered by plain stateless Web services. For example, when we stand in front of an ATM we don't have to identify ourselves as part of every request for an action. Instead, we establish our identity and the account we want to operate on at the beginning of the interactions. Subsequent action requests take place within the "context" that has been established between us and the banking service (a feature that is implicit in an object-based system since we bind directly to a serving object).

However, we can achieve similar contextual semantics with service interactions. Back-end resources like the account remain hidden from the service consumer, but we establish a context that we can use at the application level to "group" actions together. There are two possible approaches rising to prominence in the Web services community:

- **Correlation based on message content:** This is best exemplified by the correlation-set mechanism in BPEL, where parts of messages incident on a Web service can be used to form a unique identifier, implicitly linking related messages together on such "keys" as account number, date, or suchlike.
- **WS-Coordination or WS-Context (part of the WS-CAF suite of specifications):** A context identifying related messages is embedded in a SOAP header block on each related message.

Which approach we choose is determined by our deployment scenario since both broadly achieve the same goal. Where the simplicity and potential interoperability benefits of a standard, explicit context-based approach are appealing, we need additional infrastructure to deal with context generation and lifetime management. Conversely, with the implicit correlation-based approach, we need no external infrastructure, though we run the risk of altering message content solely to be able to generate uniqueness (e.g., by adding a unique "message id" field where there is no business-level requirement).

Whichever context form is selected, the upshot is the same: context allows consumers to have "stateful" interactions with Web services, but does not require back-end objects to be exposed. This means the Web service has a free hand in deciding how to manage its back-end resources, since they remain encapsulated away from the consumer. This is in contrast to the object-based approach, where invocation context is set by references to objects and the invocation history of referenced objects. The big drawback with this scheme is that object references between enterprises are likely to be brittle and impose too tight a level of coupling between the client and the serving object.

technologies that enable other computing systems to interact with the bank. The way the bank manages its infrastructure and resources and the way it operates does not have to change – it's business as usual. Remember that Web services is an enabling technology for computer-to-computer interaction rather than a radical new way of doing business. So, even though our bank has deployed a set of Web services as a way to access its business logic, it still does not have to expose its internal resources, implementation details, or practices. What Web services have brought to our banking example is the ability to automate interactions that cross its administrative boundaries.

Now it's possible for programs running on computers that are not controlled by the bank to consume the banking service and, perhaps, combine it with business logic that is unknown to the bank. We can envisage a personal finance application that can

SOA Checklist

- Service orientation focuses on shared message structure, not shared type systems.
- Messages carry sufficient data to allow the action to be completed, and are coarse grained.
- Service orientation encourages loose coupling, but does not guarantee it.
- WSDL is a contract language that specifies message structure and permitted message exchange patterns, and is categorically not an object IDL.

how best to apply this new paradigm to designing and building our applications.

Though it is tempting to view this new style as objects-plus-XML, this is a strategy that will hamper the development of truly scalable and loosely coupled services. Instead, we should aim to deliver services that expose coarse-grained processes to our partners (and not simple get/set operations); we should build stateless services for loose coupling; and we should not

www.w3.org/TR/ws-arch

- Foster, I., et al. (2002) "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration". Global Grid Forum.
- Parastatidis, S., et al. (2003) "A Grid Application Framework Based on Web Services Specifications and Practices." www.neresc.ac.uk/projects/gaf
- OASIS Web Services Business Process Execution Language: www.oasis-open.org/committees/wsbpel.
- WS-Coordination: <http://msdn.microsoft.com/ws/2002/08/WSCoord>.
- Web Services Context (WS-CTX): www.iona.com/devcenter/standards/WS-CAF/WSCTX.pdf
- Web Services Composite Application Framework (WS-CAF): www.iona.com/devcenter/standards/WS-CAF
- Purdy, D. (2003) "Service Oriented Architecture" (presentation): <http://www.douglas.com/talks/web400.zip>

About the Authors

Dr. Jim Webber is an architect and Web services fanatic at Arjuna Technologies, where he works on Web services transactioning and Grid computing technology. Prior to joining Arjuna Technologies, he was the lead developer with Hewlett-Packard working on their BTP-based Web Services Transactions product – the industry's first Web services transaction solution. An active speaker and Web services proponent, Jim is the coauthor of *Developing Enterprise Web Services*.

■ ■ ■ jim.webber@arjuna.com

Dr. Savas Parastatidis is the chief software architect at the North-East Regional e-Science Center (NEReSC), Newcastle, UK. He is NEReSC's expert in Grid computing technologies and standards. Previously, he co-led an R&D team at HP's middleware division that produced the world's first XML-based transactioning system and represented HP during the early stages of the OASIS BTP standardization effort.

■ ■ ■ savas.parastatidis@ncl.ac.uk

“these constraints are a good starting point to making applications truly service-oriented”

talk directly to our bank through Web services and at the same time combine information from our credit card provider and stock broker to offer us an up-to-date view of our financial situation. Of course, a Web services-based approach enables us to scale this application up to enable the integration of our banking service into larger processes and its participation into automated business-to-business activities (see sidebar, "Service Interactions").

Conclusions

The advent of Web services means another paradigm shift for the development community. As with the previous paradigm shift to object orientation, we are going to have to learn (sometimes by our mistakes)

expose (even conceptually) any resources from a service back end directly to a consumer because this makes applications brittle, and unable to cope with change.

Though these are simple rules of thumb (see sidebar, "SOA Checklist") and we expect our understanding of how to deploy this technology will become vastly more sophisticated in time, for now we believe these constraints are a good starting point to making applications truly service oriented. ☺

Acknowledgements

We would like to thank Mike Sick from Serene Software for his valuable comments.

References

- *Web Services Architecture*:

A LIMITED TIME SAVINGS OFFER FROM SYS-CON MEDIA

SUBSCRIBE TODAY TO MULTIPLE MAGAZINES

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



**RECEIVE
YOUR DIGITAL
EDITION
ACCESS CODE
INSTANTLY
WITH YOUR PAID
SUBSCRIPTIONS**

3-Pack

Pick any 3 of our
magazines and save
up to **\$275⁰⁰**
Pay only \$175 for a
1 year subscription
plus a **FREE CD**

- 2 Year – \$299.00
- Canada/Mexico – \$245.00
- International – \$315.00

6-Pack

Pick any 6 of our
magazines and save
up to **\$350⁰⁰**
Pay only \$395 for a
1 year subscription
plus 2 **FREE CDs**

- 2 Year – \$669.00
- Canada/Mexico – \$555.00
- International – \$710.00

9-Pack

Pick 9 of our
magazines and save
up to **\$400⁰⁰**
Pay only \$495 for a
1 year subscription
plus 3 **FREE CDs**

- 2 Year – \$839.00
- Canada/Mexico – \$695.00
- International – \$890.00

CALL TODAY! 888-303-5282

☐ MX Developer's Journal

U.S. - Two Years (24) Cover: \$143	You Pay: \$49.99 /	Save: \$167 + FREE \$198 CD
U.S. - One Year (12) Cover: \$72	You Pay: \$29.99 /	Save: \$60
Can/Mex - Two Years (24) \$168	You Pay: \$79.99 /	Save: \$137 + FREE \$198 CD
Can/Mex - One Year (12) \$84	You Pay: \$49.99 /	Save: \$40
Int'l - Two Years (24) \$216	You Pay: \$89.99 /	Save: \$127 + FREE \$198 CD
Int'l - One Year (12) \$108	You Pay: \$59.99 /	Save: \$30
Digital Edition - One Year (12)	You Pay: \$19.99	

☐ Linux World Magazine

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

☐ Java Developer's Journal

U.S. - Two Years (24) Cover: \$144	You Pay: \$89 /	Save: \$55 + FREE \$198 CD
U.S. - One Year (12) Cover: \$72	You Pay: \$49.99 /	Save: \$22
Can/Mex - Two Years (24) \$168	You Pay: \$119.99 /	Save: \$48 + FREE \$198 CD
Can/Mex - One Year (12) \$84	You Pay: \$79.99 /	Save: \$4
Int'l - Two Years (24) \$216	You Pay: \$176 /	Save: \$40 + FREE \$198 CD
Int'l - One Year (12) \$108	You Pay: \$99.99 /	Save: \$8

☐ Web Services Journal

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

☐ .NET Developer's Journal

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

☐ XML-Journal

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

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

<input type="checkbox"/> 3-Pack	<input type="checkbox"/> 1YR	<input type="checkbox"/> 2YR	<input type="checkbox"/> U.S.	<input type="checkbox"/> Can/Mex	<input type="checkbox"/> Intl.
<input type="checkbox"/> 6-Pack	<input type="checkbox"/> 1YR	<input type="checkbox"/> 2YR	<input type="checkbox"/> U.S.	<input type="checkbox"/> Can/Mex	<input type="checkbox"/> Intl.
<input type="checkbox"/> 9-Pack	<input type="checkbox"/> 1YR	<input type="checkbox"/> 2YR	<input type="checkbox"/> U.S.	<input type="checkbox"/> Can/Mex	<input type="checkbox"/> 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.

☐ WebLogic Developer's Journal

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

☐ ColdFusion Developer's Journal

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

☐ Wireless Business & Technology

U.S. - Two Years (24) Cover: \$144	You Pay: \$89 /	Save: \$55 + FREE \$198 CD
U.S. - One Year (12) Cover: \$72	You Pay: \$49.99 /	Save: \$22
Can/Mex - Two Years (24) \$192	You Pay: \$139 /	Save: \$53 + FREE \$198 CD
Can/Mex - One Year (12) \$96	You Pay: \$79.99 /	Save: \$16
Int'l - Two Years (24) \$216	You Pay: \$170 /	Save: \$46 + FREE \$198 CD
Int'l - One Year (12) \$108	You Pay: \$99.99 /	Save: \$8

☐ WebSphere Developer's Journal

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

☐ PowerBuilder Developer's Journal

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

*WHILE SUPPLIES LAST. OFFER SUBJECT TO CHANGE WITHOUT NOTICE

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

**SYS-CON
MEDIA**

Overcoming the Web Services Insecurity Complex

New standards and solutions address the security concerns of companies deploying Web services

■ Once merely so much hype, Web services are finally taking root in corporate IT. However, as organizations grow their Web services from pilots to internal integration projects to extra-enterprise deployments, they face a tangle of new security challenges.

Before they can realize the full benefits of their flexible, interoperable applications, businesses must implement trustworthy means of ensuring the integrity, confidentiality, and security of their loosely coupled systems. The good news? New solutions and standards are emerging to address these security challenges.

The benefits of Web services become more evident with each new implementation. Businesses are starting to see very positive results from this loosely coupled, platform-agnostic, service-oriented approach to integrating applications within organizations, across enterprises, and over the Internet. Yet, as Web services move from early adoption to mainstream acceptance, concerns for security rise proportionately – particularly when those applications extend beyond the safety of a secured enterprise network.

Web services security is, of course, a valid concern. The popular thinking is, if you build your information assets so anyone can access them, what's to keep everyone from getting at your critical data – or, for that matter, attacking your enterprise?



WRITTEN BY
GENE THURSTON

And while there is already a proliferation of tools that help developers build and deploy Web services interfaces to enterprise systems, Web services security has been lagging conspicuously behind.

Most Web services projects still take place inside the firewall, where there are fewer security issues.

Simple Web services deployments within an enterprise network require little more than traditional, network-based security mechanisms – such as Secure Socket Layer (SSL). However, while Web services must leverage the corporate infrastructure to be cost-effective, it is increasingly apparent that existing security solutions will not suffice as Web services environments evolve into production systems that interface with multiple consumers and providers.

Fundamental Security Requirements

Let's start by looking at what hasn't changed. Securing Web services comes down to the same fundamental security measures required by other application architectures.

- **Confidentiality:** Guaranteeing that messages are protected against eavesdroppers

- **Integrity:** Ensuring that messages have not been tampered with
- **Authentication:** Restricting access to clients that can provide proof of identity
- **Authorization:** Enabling specific clients to make appropriate requests to operations
- **Non-repudiation:** Ensuring that clients cannot refute that they sent requests and that services cannot refute that they sent responses

Fact is, much of the security infrastructure needed by Web services already exists. However, while it's important that enterprises leverage their current capabilities to address new security requirements, service-oriented architectures pose a new set of challenges that are not fully addressed by traditional security measures.

Inherently Open and Vulnerable

Web services simplify the loosely coupled integration of distributed, federated systems – systems that have been implemented with open APIs. What's more, Web services send machine- and human-readable messages that fly across the Internet. Print out an XML message and your niece could read its content. The point is, Web services-based systems are much more vulnerable to eavesdropping.

Security Distributed within the Perimeter

HTTP requests with any type of payload can easily tunnel through corporate firewalls to interact with sensitive data within the enterprise. That's why it's not enough to monitor network packets. It is also insufficient to assume that once a Web service request gets past the perimeter of the enterprise it has a legitimate right to access any corporate system within the firewall. Think of your enterprise as an office building. Some people aren't allowed in the front door. Others are permitted to enter the main lobby but cannot access specific floors. Some can access a floor of the building but not the offices on that floor. You get the picture. It's much the same with Web services. Due to their inherent openness, they require security at the perimeter of your enterprise as well as anywhere it is needed within your network to ensure that only authorized requests are met. They demand renewed focus on the way application-level security concerns are addressed, combining traditional forms of security with new

approaches to authentication and authorization, securing exposed interactions, and ensuring transport integrity.

End-to-End, Not Point-to-Point

Existing solutions provide point-to-point security. For example, SSL/TLS offers such features as authentication, data integrity, and confidentiality for secure point-to-point sessions. Problem is, Web services aren't designed for point-to-point sessions. With service-oriented architectures, the SOAP message model operates on logical endpoints that abstract the physical network and application infrastructure and, therefore, frequently incorporates multiple hops with intermediaries. As SOAP messages are sent from an initial requester to a service, intermediaries might intercept them to perform such functions as routing, requesting additional data, modifying the message, encrypting or decrypting portions of the message, or adding security tokens.

The challenge, then, with such distributed environments, is ensuring that intermediaries do not invalidate message integrity, violate the trust model, or destroy accountability. Most service-oriented architectures require end-to-end security across multiple intermediaries – both inside and outside corporate boundaries. So, while SSL might suffice for point-to-point security, it does not offer adequate protection for the privacy and confidentiality of sensitive data that is passed between multiple Web services.

Integrated, Abstracted, and Based on Standards

Securing Web services calls for unifying concepts. It requires solutions to both technical issues, such as secure messaging; and business process concerns, like policy, risk, and trust. And it demands a coordinated effort from platform vendors, application developers, network and infrastructure providers, and customers themselves.

“ Service-oriented architectures pose a new set of challenges ”

Appropriate standards organizations are hashing out the emerging security model and security-focused specifications such as WS-Security, XML Signatures, and XML Encryption (see Table 1), which will be broadly available from multiple vendors. Together, the model, specifications, and standards process will enable businesses to quickly and cost-effectively increase the security of their existing applications and confidently develop interoperable and secure Web services.

Web services are designed to simplify the integration of disparate applications, transports, and platforms, each of which might have their own security implementations. However, such islands of security are limited to their respective arenas. Furthermore, as new Web services are deployed to meet new business requirements, the burden of building the security for each new SOAP interface from scratch would quickly become prohibitive. Instead, organizations should consider adopting an overarching security abstraction layer that enables them to set security policies, create security services, and provide multiple levels of authentication, authorization, and encryption. By abstracting security, an organization need not concern itself with what type of security technologies its partners are using. It would need to know only that its partners have adequate levels of security, such as

encrypting sensitive transaction data. Furthermore, security abstraction creates platform independence, enabling a business with a heterogeneous computing environment to define access control policies centrally for implementation on each platform, rather than managing policy for each platform independently.

Take an Evolutionary Approach

Most organizations already have a security infrastructure in place. Rather than starting anew, it's prudent to leverage that infrastructure as much as possible and add new security capabilities to address the specific needs that Web services bring. This evolutionary approach fosters the creation of secure, interoperable Web services based on a set of security abstractions that enable the integration of formerly dissimilar technologies. For example, you might add message-level integrity or persistent confidentiality (by encrypting message elements using XML Encryption) to an existing Web service whose messages are carried through SSL/TLS. Thus, messages have integrity (and confidentiality) that persists beyond the transport layer. This enables specialization within an overall security framework to suit specific customer requirements while at the same time permitting incrementally deployed, evolutionary technologies.

An integrated security infrastructure must therefore offer distinct security services for Web services to consume. Every customer and every Web service has its own unique security requirements based on their business needs and operating environment. Within work-group settings, for instance, simplicity and ease of operation are top concerns, while for publicly accessible Web services the ability to withstand concerted denial-of-service attacks is a higher priority.

Different services require different levels of

“ Standards organizations are
hashing out the emerging
security model ”

Some security standards are fully developed...

Standard	Security Function	Standards Body
Secure Sockets Layer (SSL) 3.0	An implementation of the TLS standard. Provides confidentiality, integrity, and authentication. Provides private, reliable connection between authenticated endpoints.	IETF
XML Signatures	Provides integrity and non-repudiation. Defines XML syntax for representing electronic (digital) signatures, based on existing technologies (XML, XPATH, namespaces, etc.). Prerequisite for WS-Security.	IETF and W3C
XML Encryption	Provides confidentiality and integrity. Encryption syntax for XML messages or portions thereof. Prerequisite for WS-Security.	W3C
XACML	Provides a standard language for writing authorization policies, along with a protocol to request and receive access decisions	OASIS
Security Assertion Markup Language (SAML)	Provides a language for an authority to assert authentication, authorization, and attribute information, and simplifies security integration across heterogeneous environments. The authentication assertions can be used to provide "single sign-on" capability across a federation	OASIS

Others are still emerging...

Emerging Standard	Security Function	Standards Body/Contributors
WS-Security	Describes how to attach XML signature, XML encryption and authentication (including SAML Authentication Assertions, Kerberos Tickets, Username/Password tokens, etc.) to SOAP headers	Oasis, IBM, Microsoft, and VeriSign
XKMS	Reduces complexity of key management by allowing certificates to be managed, registered, looked up, or revoked, all via a Web services interface	W3C
WS-Policy	Designed to allow extensibility (with WS-Policy Attachment and WS-Policy Assertions) and provide a generic framework for all kinds of policies	BEA, IBM, Microsoft, and SAP
WS-PolicyAssertions (WS-Policy)	Defines a common set of generic policy assertions that can be used in accordance with the general policy language defined in WS-Policy to specify policy related to specification versions, natural languages, text encoding schemes, etc.	BEA, IBM, Microsoft, and SAP
WS-PolicyAttachment (WS-Policy)	Defines mechanisms for "attaching" WS-Policy policies to entities in general, and specifically how to associate policy expressions with WSDL-type definitions and UDDI entities	BEA, IBM, Microsoft, and SAP
WS-SecurityPolicy (WS-Policy)	Defines a set of "message security" policy assertions that can be used in accordance with the general policy language defined in WS-Policy to specify a Web service's message security policy	BEA, IBM, Microsoft and SAP
WS-Trust	Describes the model for establishing both direct and brokered trust relationships, including third parties and intermediaries	IBM, Microsoft, RSA Security, and VeriSign

TABLE 1 | Web Services security standards

Emerging Standard	Security Function	Standards Body/Contributors
WS-Privacy	Describes a model for how Web services and requesters state subject privacy preferences and organizational privacy practice statements	N/A
WS-SecureConversation	Secure Web services sessions	IBM, Microsoft, RSA Security, and VeriSign
WS-Federation	Will describe how to manage and broker the trust relationships (including support for federated identities) in a heterogeneous federated environment	N/A
WS-Authorization	Will describe how to manage authorization data and authorization policies	N/A

security, from passwords to digital certificates. For example, an organization might build password-based authentication services for its internally accessed inventory service but might use digital certificates-based authentication for the order-entry Web service its partners use.

These requirements may be combined in many ways and expressed at different levels of specificity. A successful approach to Web services security requires a set of flexible, interoperable security capabilities that, through policy and configuration, enables a variety of secure solutions. In this way, organizations can achieve security strategies that are appropriate for them.

Content is Key

XML messages come with rich payloads of information and provide a standard, structured format for accessing content. For that reason, Web services bring an unprecedented ability to harvest real-time data and apply that information to such things as security.

Additionally, Web services systems can leverage contextual information – who's accessing the data, what are their access rights, and what type of client device is the person using, to name a few examples. By using profile data in conjunction with business content, a business system can grant user-specific, content-specific, read and/or write access to a Web service. By interacting with the existing authentication system, it can dynamically tailor responses based on each user's most current security profile and the content of the request.

Here's an example. A company that employs an order-processing Web service

might design its system so that its business partners can enter orders up to \$100,000, but that orders exceeding \$100,000 must be entered by internal staff. Accomplishing this requires making use of in-flight message content (order value) and context (client identity).

Multi-Vendor Cooperation

To be clear, Web services security is a tough knot. Designed for distributed, federated, heterogeneous systems, the new

seamlessly. Web services has been a tremendous catalyst to collaboration. It has brought the industry together to address age-old security issues as well as several new concerns, in the quest for an interoperable, standards-based framework on which companies can run business-critical applications. As a result, new security standards (see Table 1) and solutions will soon help corporate IT teams clear the last remaining hurdle to easily deploying trustworthy, production-ready Web services. ©

“ Different services require different levels of security, from passwords to digital certificates ”

application architecture raises security issues that no single vendor can solve. Businesses should expect to work with application, operating system, management, and security companies to implement an effective and comprehensive security solution. If that sounds like a real headache, you'll be glad to know that thanks to an unprecedented level of cooperation among software companies, such multi-vendor solutions will interoperate

About the Author

Gene Thurston serves on the Web services security council for OASIS. He has more than 22 years of software industry experience, the last 10 of which have been dedicated to computer security. Gene is the security architect for AmberPoint, a leading Web services management solutions company. Core to the AmberPoint product suite is a strong policy-based authorization system that enables organizations to use identity, XML content, system state information and many other factors to produce permit/deny authorization systems.

■ ■ ■ gthurston@amberpoint.com

Transacting Business with Web Services, part 2

The coming fusion of business transaction management and business process management

■ In the first part of this article (*WSJ*, Vol. 3, issue 9), we examined the need to integrate business transaction management (BTM) software into business process management standards and products. We believe that BTM offers previously inaccessible levels of application coordination and process synchronization, radically simplifying the design and implementation of transactional business processes.

The most promising Web service/XML BPM standardization work is taking place in the OASIS WS BPEL Technical Committee. Members of the committee have been working on proposals to increase and modify the available syntax of the BPEL language to support the use of BTM. A late August submission to the committee (www.oasis-open.org/committees/download.php/3263/BPEL.and.Business.Transaction.Management.Choreology.Submission.html) raised issues #53 to #59, which were discussed in detail at the September face-to-face meeting in Redmond, WA. Here we'll look at the state of that discussion, and possible outcomes for the final WS-BPEL standard that will emerge.

We addressed the creation of business transactions in external services and within BPEL processes, and scopes; the propagation of business transaction contexts between services and processes, and the way in which a BPEL process can model a business transaction participant. This article shows the kind of syntax that process designers could use to define business transaction behaviors in their BPEL processes. We briefly



WRITTEN BY

ALASTAIR GREEN &



PETER FURNISS

show some possible variants that emerged from the recent technical committee discussion. (The new XML elements and attributes referring to business transactions in the code examples below reflect the original submission. They illustrate the principles involved, but may well be altered or rejected by the committee process during the remainder of this year.)

A client, in a pre-existing package or custom application that is external to a BPEL process, can request that the coordination service create a business transaction, receiving in reply a business transaction context. A WS-Coordination `<wscoor:CoordinationContext>` is one example of a

context structure that could be used for this purpose. Equally, a BPEL process should be able to request the creation of a business transaction, storing the resulting context element as the value of a BPEL variable. Appropriate syntax must be added to the BPEL language to support this. (In the example XML that follows, the default namespace is assumed to be that of a future, standardized version of the BPEL language.

Variables to hold business transaction contexts can be declared thus:

```
<variables>
  <variable name=
    "receivedBusinessTransactionContext"

type="wscoor:CoordinationContext"/>
  <variable name=
    "ourBusinessTransactionContext"

type="wscoor:CoordinationContext"/>
</variables>
```

A business transaction context can be created and stored using the following proposed syntax:

```
<businessTransaction action="new"
context="ourBusinessTransactionContext"/>
```

If a client invokes a WSDL-defined Web service operation that is offered by a BPEL process, then it should be able to attach a business transaction context to the invocation message, and have that context stored in a variable in the receiving process. This implies additional syntax in the `<receive/>` verb-element:

```
<receive
  partnerLink="customer"
  portType="al:reservationPT"
  operation="createReservation"
  variable="reservationRequest"
  businessTransactionContext=
    "received BusinessTransaction
    Context" />
```

Equally, a BPEL process that holds a context (either by virtue of importing it via a `<receive/>` or by creating it using the new verb-element `<businessTransaction/>`) should be able to transmit that context when it invokes a Web service. This again implies new syntax:

```
<invoke
  partnerLink="airline"
  portType="al:reservationPT"
  operation="createReservation"
  inputVariable="reservationRequest"
  inputBusinessTransactionContext="our
    BusinessTransactionContext"
  outputVariable="reservationDetails"
  outputBusinessTransactionParticipants
    ="FlightComponent" />
```

The purpose of sending a business transaction context to a service is to enable the service to register participants. Registration is carried out using a registration capability that allows coordinator-participant relation-

ships to be constructed (OASIS BTP, WS-Coordination, and WS-CAF all have this feature). Services register participants with the coordination service, not with the invoking application. However, for correlation purposes the identity of the registered participants are returned to the invoker using the attribute `outputBusinessTransactionParticipants`.

The context/participant traffic may also travel in the reverse direction. Sometimes a client makes a request that the service provide a business transaction context, allowing the client to then register as a participant with the service's business transaction. Two new attributes, `<outputBusinessTransactionContext>` and `<inputBusinessTransactionParticipants>` are used to support this advanced behavior.

Creating and Terminating a Business Transaction

We have already shown you a simple example of creating a business transaction. It should also be possible to create a business transaction that is the child of a parent transaction. This allows transactions trees to be created, which can aid in complex synchronization and ordering cases:

```
<businessTransaction action="new"
  context="ourBusinessTransactionContext"
  parentContext="receivedBusiness
  TransactionContext">
```

Once a process has finished interacting with the services that carry out a business transaction's work, it must either confirm or cancel the transaction:

```
<businessTransaction action="confirm"
  context="ourBusinessTransaction
  Context"/>
```

This requests that the underlying BTM coordination service confirm all participants. Likewise, a business transaction can be instructed to cancel all of its participants:

```
<businessTransaction action="cancel"
  context="ourBusinessTransaction
  Context"/>
```

A process may wish to terminate a subset of the participants in a transaction. To do this it can use the identity of a participant:

```
<businessTransaction action="confirm"
  context="ourBusinessTransactionContext"
  participants="flightComponent
  hotelComponent"/>
```

This instruction implicitly cancels any participants that are not specified.

The BTM coordination service ensures that termination instructions of this kind are correctly and completely delivered, even in the case of temporary process, processor, or network failures. Note that the BTM coordination service may be deployed as part of a BPEL execution engine, or may be freestanding.

How BPEL Processes Model Business Transaction Participant Behavior

BPEL relates in two ways to participants. A Web service, external to a BPEL process, may act as a participant. And a BPEL process (which presents itself to other processes as a Web service) may also act directly as a participant.

BPEL is not designed to manage persistent

quote, and one to turn that quote into an order (and possibly, one to cancel the quote).

A business transaction participant can use these service operations to model provisional, contingent behavior; and finalization behavior (confirmation, cancellation). In this case, a participant within a BPEL process might invoke `getQuote` on the service as its prepare operation, and would then invoke `cancelQuote` as its cancel operation, or invoke `executeOrder` as its confirm behavior.

The example shows this case, with a BPEL process registering itself as a participant in a business transaction using a received business transaction context (see Listing 1).

New handlers are added to deal with BTM cancel and confirm decisions. The existing fault-handler is triggered if a failure occurs during the forward work – including the case where a BTM cancel instruction is received before the forward work completes. The confirm and cancel handlers are proposed as a supplement to the existing (nontransactional) compensation-handler

“ A business transaction context can be created and stored ”

resources internally within an executable process. BPEL variables are largely intended for control flow and for passing data between the process and ancillary Web services. It is up to vendors to provide implementations of the relevant BTM standards, such as WS-T and BTP, that can be inserted into Web service operations to allow services to correctly interoperate with a coordination service.

There is one important case, however, where a BPEL process is the appropriate place to define participant behavior. Take an environment where an existing application offers operations on its service interface (or where such an interface can easily be added, to allow the application to be accessed as a Web service). It may not be possible to modify or enlarge the application's suite of operations. For example, a CRM or ERP package may offer an operation to create an order, and one to delete an order. Or it may have an operation to offer a

model for local exception processing in BPEL (although a cancel-handler and compensation-handler could be merged).

The registration of the participant is performed by `<businessTransaction action="register">`. This enables triggering of the confirm and cancel handlers by the BTM messages.

All of the new constructs described here would be used both in abstract BPEL processes (which define collaboration protocols), and in executable processes.

The Changes and Additions Needed for BPEL to Support BTM

In the first part of this article, the authors posed four questions that must be answered in the BPEL standardization process to properly integrate BTM. The answers are being considered in the BPEL community. There is a spectrum of opinion within the committee on how far BPEL should go in recognizing or supporting BTM features in the first version of the standard.

How Do Business Transaction Coordination Protocols Work?

All transactional coordination protocols have some common features (see Figure 1). Participant systems update business information: each piece of state data must be changed in accordance with the instructions of a central coordination service. The coordination service is in turn the servant of an assembly application, and of a terminating application. (Frequently the assembler and the terminator are fused in a single controlling application.)

The assembler requests that the coordination service create a coordination or transaction, and tags its communications with participants with a transaction identity and the address of the coordination service (this is termed propagation or infection).

The participants then signal the coordination service that they are prepared to be instructed by the coordination service. Typically this means that they have effected provisional, reversible state changes. (Participants may also communicate that they have failed to prepare, either for business or technical reasons. In this case they are not available to the terminator for inclusion in the final outcome.)

The terminator now decides what completion instructions should be communicated to each participant. Typical examples of mutually exclusive completion instructions are confirm and cancel. (These are the conceptual instructions used by BTP and WS-T; future, special-purpose

coordination protocols might employ a larger set of possible outcomes.)

Another way of looking at this progression is shown in Figure 2. A BTM participant moves from an active state to a prepared state (where it has carried out provisional work), and then to a final state (either

confirmed or cancelled). The state transition diagram shows how application messages and BTM protocol messages cause these state changes.

When a participant receives its completion instruction it takes whatever internal action is needed to conform with the semantics of the instruction for the particular type of business transaction being performed.

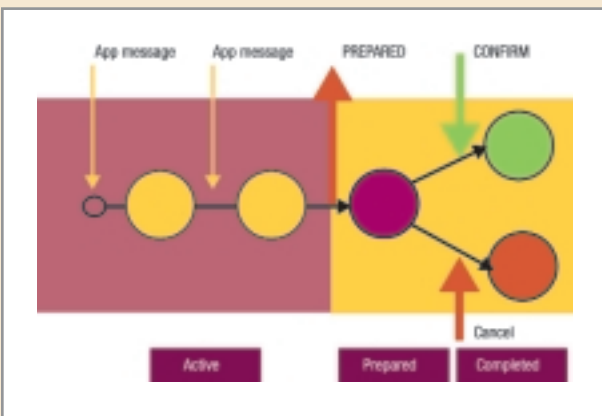


FIGURE 2 State transition diagram

For example, an airline reservation system might prepare by creating a provisional reservation. If instructed to cancel, it would delete the reservation (possibly levying an administration fee and/or retaining information for subsequent business intelligence operations). If instructed to confirm this particular busi-

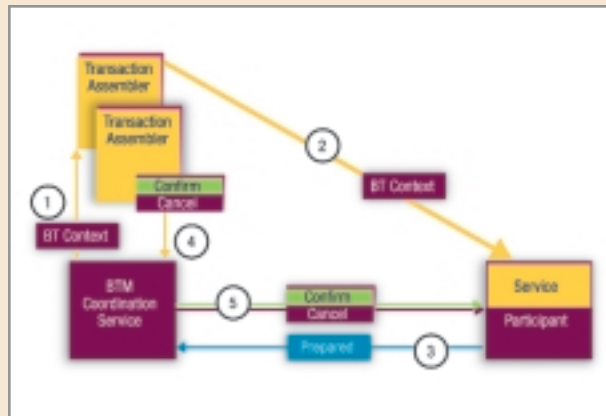


FIGURE 1 Business transaction coordination protocols at work

• How do you propagate business transactions between Web services and business processes?

A business transaction context needs to travel between the parties for propagation. The committee is divided on how this behavior should be described or specified by process designers. An "implicit" approach would involve either runtime configuration or design-time marking of Web service invocations as being "transactional." The "explicit" approach makes context transmission visible at the BPEL level. (The implicit approach using deployment-time configuration may be very difficult to understand, and probably presents significant problems in defining abstract processes.)

Either way, the parties need to agree on the type of business transaction context being communicated. BPEL should permit this choice at deployment time, not design time. The representation of the agreed context "on the wire" is achieved by WSDL's ability to map an abstract message part to a type and position within a concrete message.

• How do you initiate and terminate new business transactions within a business process?

Provide new verb-elements in BPEL: <businessTransaction>. Some technical committee members have suggested that business transaction creation and termination should occur implicitly when scopes are entered and left. This would require some construct to mark scopes as "transactional." This is tied to the notion of implicit context transmission.

• How do you propagate business transactions between business processes and nested scopes?

If an explicit approach to creation/termination is preferred, then simply allow nested scopes to use a context variable that has been declared and assigned in an outer scope/process. If an implicit approach is adopted, then inner scopes would inherit the business transactional context of their outer scope.

• How do you define the reaction of processes and subprocesses to the progress of a business transaction?

Allow a process to register itself as a participant, using the <businessTransaction action="register"> variant of the new verb-element. There seems to be little appetite for permitting scopes within processes to act as BTM participants.

ness transaction the system would change the reservation status to confirmed, and trigger related fulfillment and billing/payments processing. Note that the participating service fully controls its own internal implementation of the prepare, confirm, and cancel operations; these are written to produce an externally visible effect that is compatible with the overarching business contract that the transaction is effecting. The assembler/terminator applications are unaware of the participant's implementation; the whole interaction is conformant, in BPEL terms, with an abstract process that describes the contents and legitimate sequences of process-to-process messages.

The terminating application uses its business rules to orchestrate the completion of the business transaction. Such rules will determine whether there is a viable set of participants, or whether a critical participant's inability to prepare has vitiated the whole transaction. The terminator's rules may be used to select a subset of possible participants for confirmation, discarding the unwanted remainder, or it may simply confirm all of the participants. (This ability to manipulate the final population of participating services is known as a "cohesive business transaction" or "cohesion" in BTP terminology. WS-T BA can be used to implement similar behavior.)

“ BPEL variables are largely intended for flow control and for passing data between the process and ancillary Web services ”

In addition, we expect that considering the standardization of BPEL's interaction with BTM protocols will increase the desire to achieve a single, agreed BTM standard (perhaps to be called WS-Business Transaction?). The current confusion between BTP (an OASIS Committee Specification) and WS-Coordination plus WS-Transaction (draft proprietary specifications from three key vendors) needs to be cleared up so that end users can stop worrying about the current standards flux and implementers can get products to market more quickly. (The recent emergence of yet more draft proprietary transaction management specifications in the WS-Composite Application Framework accentuates the need.) We believe that there should be wide agreement that the fundamental two-phase outcome principles of BTP and WS-T Business Activity align sufficiently to end up with a common, single standard for Web service transactions. (In our view, WS-T Atomic Transaction duplicates, as a special case, the capabilities of WS-T BA, and is therefore equivalent.)

Given clarity on the standard for Web service

business transactions, and the proposed BPEL revisions, end users will be much better able to realize the promises of BPM (relative simplicity and reliability), in environments that support the processing of valuable economic transactions. ©

■ About the Authors

Alastair Green is CEO/CTO and cofounder of Choreology Ltd, the business transaction management (BTM) company. A coauthor of the OASIS Business Transaction Protocol standard and an active member of the OASIS WSBPEL committee, he has spent over 20 years helping to produce distributed transaction products for software vendors, and deploying them in end-user business processes, particularly in banks.

■■■ alastair.green@choreology.com

Dr Peter Furniss is chief scientist at Choreology Ltd, where he is responsible, along with Alastair Green and Tony Fletcher, for Choreology's involvement in OASIS BTP, and is the editor of the BTP specification. Peter is a cofounder of Choreology and was formerly at HP Arjuna Labs, working as a product architect on the failure-recovery aspects of Arjuna's Java transaction service

■■■ peter.furniss@choreology.com

Listing 1

```
<process>
  <variables>
    <variable name="receivedContext" ... />
    <variable name="quoteParticipant" ... />
  </variables>
  ...
  <!-- handlers appear in the script before the forward
work -->
  <faultHandler/> <!-- clean up partial provisional
work -->
    <!-- no op: getQuote operation failure
leaves no trace -->
  <cancelHandler> <!-- cancel prepared, provisional
work -->
    <invoke operation="cancelQuote" ... />
  </cancelHandler>
  <confirmHandler> <!-- finalize prepared, provisional
```

```
work -->
    <invoke operation="executeOrder" ... />
  </confirmHandler>

  <!-- the forward work>
  <receive ...
    businessTransactionContext="receivedContext">
    ...
  </receive>
  <businessTransaction action="register"
    registerWith="receivedContext"
    registeredAs="quoteParticipant">
  <!-- provisional work, implicit prepare -->
  <invoke
    operation="getQuote" ... />
  <reply
    businessTransactionParticipant="quoteParticipant"
    ... >
  </process>
```

Download the code at
sys-con.com/webservices

The State of Web Services, A.D. 2003

They're 'a tool for the times,' say the experts

■ What do you get if you cross an early 21st-century visionary CTO with a late 19th-century employee of the Edison Electric Light Company? Answer: a fantastic keynote address at Web Services Edge 2003 West, held in Santa Clara last month.

The visionary in question was Allan Vermeulen, coauthor of the codehead's classic *The Elements of Java Style*, and now CTO of the world's largest online retailer, Amazon.com. The Edison employee was Sam Unsell, whose contribution to the development of technology – Vermeulen explained – was to develop an economic model for electricity use in Chicago.

As with electricity then, so with Web services now. This, in Vermeulen's view, is the next shoe that needs to drop.

"Somebody has to be the Sam Unsell of Web services," he proclaimed, meaning that someone in the Web services space has to come up with a good idea for what kind of economic model is best suited to underpinning the technology.

Commercially available electricity, he explained, was only able to catch on and become pervasive because, with Unsell's help, the Edison Electric Light Company invented not just the first commercially practical incandescent lamp but a complete electrical distribution system for light and power – including generators, motors, light sockets with the Edison base, junction boxes, safety fuses, underground conductors, and other devices.

The comparison held the packed audience at the Santa Clara Convention Center, quite literally, spellbound. It was deemed by all who attended to be one of the most memorable and – pun intended – illuminating keynotes in the history of the Web Services Edge series of Conferences and Expos, which is saying some-



WRITTEN BY
JEREMY GEELAN

thing since in previous years keynotes have been given by folks like the "Father of Java," Sun's James Gosling; and the "Father of Markup," Charles F. Goldfarb.

Vermeulen's ebullient opening keynote characterized well a conference that for three days brimmed with good content and animated discussions.

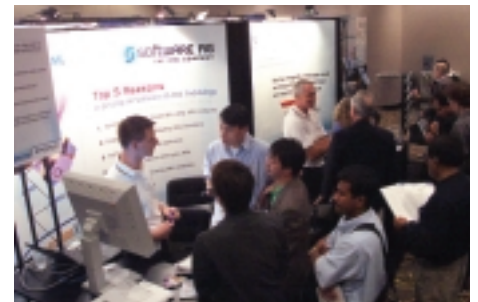
The Complexity Crisis

Keynote discussion panels featured the likes of John Schmidt, CTO of the No. 1 specialty retailer in the U.S., Best Buy, who brought to bear his enormous real-world experience of Web services: Best Buy moves about 100 gigabytes of data a day – inventory data, foundation data (pricing, etc.) – and top management throughout industry, Schmidt reported, is starting to recognize the issues of complexity in IT.

"We need," he observed, "to help take layers of complexity out of our IT environment." Whereas Web services, in Schmidt's view, may take us in the opposite direction.

Coming from a seasoned expert like Schmidt, who also chairs the Methodology Committee of the EAI Consortium, this was a compelling message – especially once he had set the stage with a reference to what he called "the dark side of systems integration – the complexity crisis."

Best Buy alone has over 600 technologies to support 165 technology capabilities, Schmidt reported. "A couple of years ago it took about 20–30 days to build a complete interface," he said. "Nowadays it takes



about 4–5 days. Best Buy now adds over 550 interfaces every month (over the past 3 months)."

In other words, and this was Schmidt's point, "As complex as our environment is at the moment, Web services is going to make it even more complicated."

A Web service can be built almost at the push of a button, Schmidt concluded.

"Accordingly, they will proliferate on a massive scale."

Keynote Panel: Web Services Paradigm Has Evolved

At another keynote discussion panel the question was "Interoperability: Is Web Services Delivering?"

When panel moderator Derek Ferguson, editor-in-chief of *.NET Developer's Journal*, asked the panel members to set the parameters of the discussion by first defining Web services, it became clear that the invited experts on the keynote stage were agreed that, while defined by the interop protocol known as SOAP 1.1, no longer do Web services necessarily have to be XML, or even over HTTP. The paradigm has evolved.

David Chappell, VP and chief technology evangelist, Sonic Software, stressed that in his view, while Web services interactions do not have to be across HTTP, "XML is key to defining what a Web services interaction should be. It's best suited for the role of serving as the language for describing the data that needs to be exchanged between applications."

Gary Brunell, VP of professional services for Parasoft, pointed out that "If we're going to use the term 'Web services,' it does suggest the Web, and so HTTP and HTTPS. XML is very important too," he added.

Meantime, David White of Microsoft said he disagreed with the "Web" part of the term 'Web services.' "I'm a big believer in transport agnosticism," White said. "I'm really more concerned about the data representation and the invocation, rather than the transport. The key is to get something back and forth without great expense."

Chappell agreed: "To me the 'services' word is the more important, the service-oriented architecture part. 'Web services' is now a more generic term, for 'the next thing that's going to solve the problems we're trying to solve.'"

Next the panel moved on to pinpoint whether Web services has yet become com-

mon beyond the firewall, or is still mostly being used for intra-company use.

Chappell noted that in his experience there is about an 80:20 divide in terms of adoption. "80% is within the corporation's control, and 20% involves the public Internet (the Web) – dealing with other business partners, for example." Brunell agreed that mission-critical apps were still "few and far between," adding, "That's why we are all coming to these conferences."

Microsoft's White noted that on the contrary he had seen mission-critical things happen inside Web services. "We've only just gotten there," he said, "but I have absolutely seen mission-critical Web services in our customer mass." Not out in the B2B space, he conceded.

JBoss Group's CTO, Scott Stark, pinpointed one crucial piece of the jigsaw that's still missing: "Single Sign-On is a joke, I have about 35 accounts; no one has an agreement yet on a one-stop solution, and no enterprise technology can surmount that. J2EE is still basically a middleware technology," Stark continued, "it's not out there bridging enterprises."

The bridging role, then, remains perfect for Web services. But these things take time, Stark added. "Developers are going to have to get comfortable with Web services first:

Overheard in Santa Clara

"Do think of the 'W' in Web service as a way to ask 'Why not?' when presented with the difficulties or challenges of opening up a system or sharing information across departmental systems?"

–Velan Thillairajah

EAI Technologies

Founding Member of the EAI Industry Consortium

J2EE has taken 7 years to become a reasonably accepted technology." He pointed out that XML wasn't without its shortcomings. "XML is a double-edged sword. My head starts spinning after I've read the 10 different XML Schemas. So the usual technology curve also impedes the adoption of Web services. But that's just the nature of the beast."

Asked if XML might be replaced, White explained that one of the problems is that good tools are often the last thing to appear after a "technology burst" such as the one we are seeing around Web services. "I'm not a seer," White said, but the key to widespread adoption of any new technology is completion of the specs (we're there), demos (we're getting there), and then the tools (they're coming)."

JBoss's Stark agreed. "XML isn't going anywhere. Before there was IIOP and it went nowhere. Clearly XML is the only technology, however complex it might be, that's tried to address the problem. Besides, IIOP was even more complicated, and writing, say, a TCP/IP stack, is not a productive endeavor."

Stark then minted the phrase of the conference. "People have more comfort now with distributed programming; it's a tool for the times." ©



Mindreef and XMethods Partner

(Hollis, NH and San Jose, CA) – Mindreef, Inc., a provider of Web services diagnostics, and XMethods, the place on the Web for discovering interesting Web services, have announced Scope-it on



XMethods. Scope-it allows XMethods users to easily investigate any of the hundreds of Web services available on XMethods with an online service based on Mindreef's SOAPscope diagnostic system.

XMethods services represent a broad range of vendors through

services built with technology from companies including IBM, Microsoft, Oracle, BEA,

Systinet, Borland, and The Mind Electric. With Scope-it, users simply fill out a form generated from the service's WSDL document to invoke the service. Users also gain insight into how the service was designed and constructed by using the online version of SOAPscope's viewers and analysis. www.mindreef.com, www.xmethods.net

Singapore Leads in Setting Standards for Web Services

(San Francisco) – Singapore is the first Asian country to partner with OASIS to lead in creating an international Web services standard. The founding members of the OASIS Framework for Web Services Implementation (FWSI) Technical Committee include co-chairs, IDA and the Singapore Institute of Manufacturing Technology (SIMTech), a secretariat, the Information Technology Standards



Committee (iTSC), as well as six local companies – Crimson Logic, Ecquaria, EG Innovations, ESS Software, ReadIMinds, and Singapore Computer Systems. They will be joined by Sun Microsystems, NEC, and Yellow Dragon, two international e-business consortia – RosettaNet and CommerceNet – and a

research institution, the Centre for E-Commerce Infrastructure Development (CEDID) at the University of Hong Kong.

The new standard will help companies accelerate Web services implementation and adoption while generating revenue.

To kick-start the standards work, OASIS has issued a Call-for-Participation to its more than 600 global member companies and individuals.

www.ida.gov.sg, www.oasis-open.org

Actional Unveils New Web Services Management Platform

(Mountain View, CA) – Actional Corporation has announced significant enhancements across its entire Web services management platform. The new release includes Actional SOAPstation 5.0, Actional Looking Glass 5.0, updates to its Actional Active Agents, and two new products – Actional SOAPstation Edge and the Actional MyServices Portal.

Actional SOAPstation Edge adds full-featured XML firewall capabilities to the Web service brokering facilities of Actional SOAPstation. Actional MyServices Portal is a browser-based dashboard providing highly customizable views for monitoring Web service activity and SLA compliance. Actional is also introducing Actional Service Stabilizer to help identify and automatically correct undesirable Web service and service-based application operating conditions before they become problems.

www.actional.com

Savvion Enables the Performance-Driven Enterprise

(Santa Clara, CA) – Savvion, Inc., has announced general availability of Savvion BusinessManager 5.0, the latest version of its



business process management (BPM) system. Savvion

BusinessManager 5.0 is the first BPM system to deliver process life-cycle management, the complete end-to-end delivery of business processes from modeling to deployment to management to process improvement. Unique in the industry, this BPM system puts role-specific tools directly into the hands of those closest to the process at the appropriate time.

www.savvion.com

Flashline Releases Flashpack for FEA

(Cleveland) – Flashline, a provider of software asset management for enterprise programs, has released FlashPack for Federal Enterprise Architecture (FEA). This FlashPack is an add-in to the Flashline Registry, giving federal agencies a preconfigured solution that delivers commercial best practices for managing assets and projects in accordance with the FEA. In addition, the FlashPack enables agencies to manage compliance with, and speed adoption of, the FEA, and assists agencies with managing their Office of Management and Budget Exhibit 300 busi-



ness case submissions.

The FlashPack for FEA incorporates the four recently released Federal Enterprise Architecture Reference Models and includes the nearly 1,000 Reference Model categories. Additional Reference Models will be integrated as they become available.

The FlashPack for FEA includes Flashline professional services to customize the FlashPack for specific agency requirements. These services include custom extensions of FEA categories, asset metadata configuration, and sample asset setup.

www.flashline.com

Confluent Software, Netegrity to Help Customers Deploy Web Services

(Sunnyvale, CA) – Confluent Software, Inc., has teamed with Netegrity to develop the Web Services Reference Architecture, which helps define the components necessary to manage and secure a services-based environment. Confluent collaborated with Netegrity to clarify the role of Web services management in securing Web services. Confluent also announced integration with the latest version of the Netegrity TransactionMinder



—Continued from page 7

vastly increasing the possibilities for partnering.

Web services will be most useful, and widely usable, if they comply with widely subscribed standards for business objects, such as those published by the Open Applications Group Inc. (OAGI) and other standards groups. OAGI standards for CRM objects are under development. Industry-specific standards similar in concept to OAGI, like RosettaNet for high tech and HL7 for healthcare, but for industry-specific business objects and processes are in development. These standards, depending on their adoption, will determine Web services' value in communicating and leveraging customer data among companies within the same industry.

Until standards are widely available and subscribed to, you will have to agree on an internal standard such that each Web service you make available encompasses all your internal applications as well as those of partners. Be sure to ask your CRM vendor if they have delivered Web services to consolidate and access customer data. While Web services aren't the only tool you will need to mine your customer information, used strategically they will take you a long way toward a successful CRM implementation that improves customer satisfaction and encourages ongoing, profitable customer relationships. ☺

■ About the Author

John Wookey is senior vice president of Oracle Applications, responsible for the Sales and Marketing products lines of Oracle's CRM suite, as well as Oracle's Healthcare, Life Sciences, and Student Systems development. Since joining Oracle in 1995, John has also worked with a number of Oracle Applications development groups, including financial applications, Oracle projects, applications globalization, and financial services.

■■■ john.wookey@oracle.com

WSJ ADVERTISER INDEX

ADVERTISER	URL	PHONE	PAGE
Altova	www.altova.com		9
BEA dev2dev 2003	bea.com/dev2devdays2003	925-287-5156	33
Comdex	www.comdex.com		29
Ektron	www.ektron.com/ws		13
IBM	ibm.com/websphere/seeit/portals		60
Java Developer's Journal	www.sys-con.com/java	201 802 3021	41
JDJ Store	www.jdjstore.com	888-303-5282	31
LinuxWorld Conference & Expo	www.linuxworldexpo.com		27
Macromedia	www.macromedia.com/go/cfmxad		23
Mindreef	www.mindreef.com	603-465-2204	4 & 5
Parasoft	www.parasoft.com/ws11	888-305-0041	2
Quest Software	www.java.quest.com/jcsr/ws	800-663-4723	59
Strike Iron	www.strikeiron.com		6 & 17
SYS-CON Events	www.sys-con.com	201 802 3069	35
SYS-CON Media	www.sys-con.com/suboffer.cfm	888-303-5282	45
SYS-CON Media	www.sys-con.com	888-303-5282	21

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.

IN THE NEXT ISSUE OF WSJ...

Focus on Portals and Open Source

Portals & Web Services:

When Business Issues Are Technical

We've all heard the terms: portals, gadgets, portlets, dashboarding. But what does it all mean? And what role do Web services play in this exciting new world of componentized content?

Will Standards Turn Portals into Commodities?

Portal frameworks will in all likelihood become commodities (though it's debatable whether normal market pressures, rather than standards, aren't the real driving force). Given this prediction, you are probably asking what you should do to position your applications.

Developing Web Services with Open Source

Development organizations need to start using Web services technology, but can't always afford to make significant investments in tools that ultimately prove critical. The open source model helps these groups by allowing them access to a low-cost solution for Web services development.

PLUS

Optimizing Web Services Using Java, Part 1

What lies behind Web services? Some say the answer depends on the power of the language used in the implementation, in addition to all known standards like XML, SOAP, and WSDL.

Using Web Services for Business – Making Yourself Understood

If the content of a SOAP message isn't understood, or the recipient of a message does not know what to do with it when they get it, then using Web services for business, even with extensions for reliable delivery and security, just won't work. To solve this problem you need to define what the contents of a message mean.

WebServices
JOURNAL



solution, allowing customers to quickly deploy standards-compliant Web services security solutions. Confluent provides a codeless mechanism to leverage Netegrity TransactionMinder while extending security enforcement and visibility into integrated applications.

The Netegrity Web Services Reference Architecture aims to help simplify Web services security and management by explaining how the prevention, enablement, and enforcement layers of a Web services architecture fit together. Confluent contributed feedback from its various customer engagements, providing significant input regarding the role of Web services management as a security-enforcement layer, and its implications in the broader context of enterprise-wide application-level operational policy enforcement.

www.confluentsoftware.com, www.netegrity.com

Sonic Software Delivers Comprehensive ESB-Based Business Integration Suite

(Bedford, MA) – Sonic Software has completed delivery of the Sonic Business Integration

Suite, a comprehensive integration product family built on an enterprise service bus (ESB), and has announced availability of Sonic Orchestration Server, Sonic XML Server, and Sonic Integration Workbench. These three products complement the capabilities of Sonic ESB to provide sophisticated business process management, XML processing, storage and query, and an integrated toolset for the suite. The Sonic Business Integration Suite, which was announced in April 2003, delivers a distributed, flexible, and standards-based integration infrastructure that allows companies to connect, orchestrate, and manage any number of services or applications from a few up to many thousands across the extended enterprise.

www.sonicsoftware.com

Layer 7 Technologies Launches Security and Policy Coordination Solution

(Santa Clara, CA) – Layer 7 Technologies has introduced the SecureSpan Solution, a

security and policy coordination technology for Web services. SecureSpan delivers an end-to-end solution for controlling security and connection policies across Web services-enabled software. With SecureSpan, organizations can now manage and coordinate security and

connection policies without inflexible programming.

The SecureSpan Solution is delivered as a series of integrated components for configuring, deploying, and auditing security and policy preferences between Web services.

www.layer7-tech.com

AmberPoint Introduces Distributed Exception Management Solution

(Oakland, CA) – AmberPoint, Inc., has announced a distributed exception management solution, AmberPoint Exception Manager. This

offering detects, diagnoses, and resolves operational and business exceptions in Web services systems – from simple data entry errors to complex business faults, such as orders lost in processing. Using AmberPoint Exception Manager, enterprises can react more quickly to operational and business contingencies, minimize inefficiencies, and reduce the costs of maintaining their Web services environments. Due to its distributed, agent-based architecture, AmberPoint Exception Manager is able to detect and resolve distributed exceptions, where the clues to the condition reside in multiple messages.

www.amberpoint.com

RosettaNet Enters Strategic Alliance with China Government

(San Jose, CA) – RosettaNet, a global e-business consortium, and China's Ministry of

Science and Technology, a central government agency under the State Council responsible for the nation's science and technology activities, have formed a strategic alliance that will promote the adoption and implementation of RosettaNet standards within China's high-technology industry and multinational companies. The deployment of RosettaNet standards within the region is expected to fuel economic growth by enhancing trading partner relationships and increasing e-commerce business opportunities for technology companies.

www.rosettanel.org, www.most.gov.cn

Empirix Reduces Risk for Web Services Implementations

(Waltham, MA) – Empirix Inc., a provider of integrated test and monitoring solutions for

Web, voice, and network applications, has launched two new solutions designed specifically for Web services testing

and monitoring. With Empirix e-TEST suite 6.8 and OneSight 4.8, companies can quickly and easily test and monitor Web services, increasing the chances of a successful Web services implementation.

e-TEST suite and OneSight feature a new, optional Web services script wizard that simplifies the scripting required to test and monitor Web services. As a result, complex, multistep Web services transaction scripts can be created without any programming and deployed in minutes.

www.empirix.com

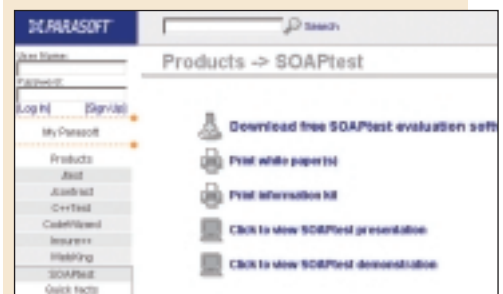
Parasoft SOAPtest Checks Conformance to WS-I Basic Profile 1.0

(Monrovia, CA) – Parasoft, a provider of automated error prevention (AEP) software

solutions, has announced that SOAPtest 2.1 now has extended testing capabilities for Basic Profile 1.0, a set of specifications developed by the Web Services Interoperability Organization (WS-I), which establishes a baseline for interoperable Web services.

SOAPtest 2.1 tests the Web Services Description Language (WSDL) for conformance to Basic Profile 1.0 using the test tools developed by WS-I. Parasoft contributed to writing the test tools as a member of the WS-I Test Tools Working Group. An official version will be released by WS-I in November. SOAPtest parses the WSDL, passes it to the test tools, and produces a WS-I conformance report that can be viewed in a Web browser.

www.parasoft.com/soaptest



In the article "Get Smart" by Kevin Hakman, in the September issue of *Web Services Journal* (Vol. 3, issue 9), on page 36, the sentence should read:

Technical approaches similar to those offered by Curl focus on pumping up the browser's capabilities by asking the user to add plug-ins or extra Java runtimes and components. Approaches like that of Droplets seek to supplant the browser altogether by providing an alternative client to install and a special server to communicate with the alternative client.

Deliver **dynamic** PDFs from your J2EE application server

JClass® ServerReport

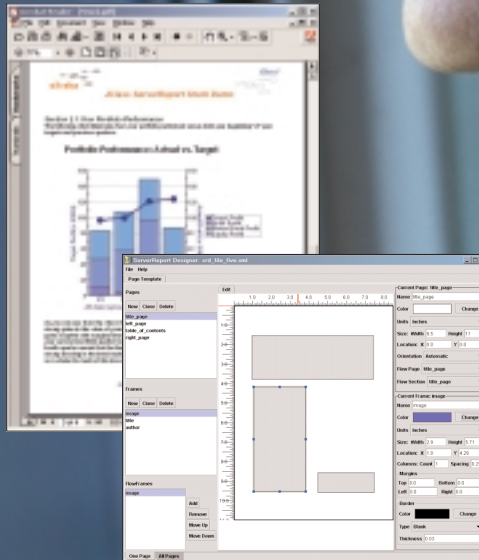
Bring high-quality printable documents, dynamically generated with up-to-the-minute data, to your Servlet, JSP and J2EE applications. Perfect for financial reports or e-billing statements, JClass ServerReport flows your content and data into customizable page templates, generating customized Adobe PDF documents that display and print anywhere.

100% Java class library for easy integration with your application server

Scalable design for high-volume web application environments

Now easier to use with new visual page design tool

New support for foreign-language content, including Asian characters and fonts



Evaluate and experience JClass today - visit:

<http://java.quest.com/jcsr/ws>

WebSphere® software

See information organized.
See access personalized.
See service recognized.

Can you see it?

IBM WebSphere, the market leader in portals, gathers information from multiple sources into one personalized view, so employees, partners and customers see what they need, when they need it. On demand. WebSphere is open, so it works with current IT investments. Combined with Lotus® dynamic interaction, everything from customer loyalty to ROI starts looking up. For a portal InfoKit, visit ibm.com/websphere/seeit/portals

@business on demand™ software



IBM, WebSphere, Lotus, the e-business logo and e-business on demand are registered trademarks or trademarks of International Business Machines Corporation in the United States and/or other countries. Information about portal market share is based on new license revenue for 2002 from the Gartner report, "IBM Has Top Share in All Application Integration, Middleware Markets" (5/03). ©2003 Gartner, Inc. ©2003 IBM Corporation. All rights reserved.