

THE FIRST WEB SERVICES

RESOURCE CD!

\$119  
CD  
VALUE  
FROM  
WEB SERVICES  
JOURNAL

March 2002 Volume 2 Issue 3

# WebServices JOURNAL

.NET J2EE XML

ATTENTION RETAILERS  
PLEASE DISPLAY UNTIL MAY 31, 2002

WSJ2.COM

JUNE 24-27, 2002

Web Services Edge  
Conference & Expo

Coming to the Javits Center in New York

Getting **greater**  
**business**  
value from Web services



The most comprehensive programs addressing  
the key issues facing developers and managers  
in today's Internet environment



From the Editor  
Hype, Stealth and the  
Dark Side of Web Services

by Sean Rhody pg. 5

**NET**  
SUCCESS STORY  
PAGE 44

RETAILERS PLEASE DISPLAY  
UNTIL MAY 31, 2002

\$14.99US \$15.99CAN



SYS-CON  
MEDIA



**Industry Insight: Secure Web Services**  
*To more appealing collaborative commerce*



Norbert Mikula  
16

**Focus on Security: Securing Web Services**  
*New efficiencies - and new security challenges*



Mark O'Neill  
20

**Focus on Security: Beyond the Hype...**  
**the Reality of Web Services Adoption**  
*Why and how real-world Web services are in use today*



Scott Durchslag  
28

**Sun ONE: The Sun Rises on Web Services**  
*An overview of the Java XML Pack: five new APIs*

Kyle Gabhart  
32

**WSFL: Getting into the Flow: the Web Services  
Flow Language** *A first step into potentially revolutionary changes*

James Snell  
40

**Focus on Security: Making Second-Generation  
Web Services Secure** *What's next after today's technologies?*



Jim Ducharme  
48

**BPM: Web Services' Impact on Business  
Process Management** *The promise of a single solution for integration*

Vittorio Viarengo  
54

**Tools: Concurrently Accessing Multiple Web  
Service Instances** *Web services toolkits make short work of large searches*



Joe Verzulli  
60

# **Sonic Software**

**[www.sonicsoftware.com](http://www.sonicsoftware.com)**

# **Sonic Software**

**[www.sonicsoftware.com](http://www.sonicsoftware.com)**

# SpiritSoft

[www.spiritsoft.net/climber](http://www.spiritsoft.net/climber)

# WebServices

.NET J2EE XML JOURNAL

## INTERNATIONAL ADVISORY BOARD

JEREMY ALLAIRE, ANDREW ASTOR, STEVE BENFIELD, DAVE CHAPPELL,  
PHILIP DESAUTELS, GRAHAM GLASS, TYLER JEWELL, PAUL LIPTON,  
DAVID LITWACK, NORBERT MIKULA, FRANK MOSS, BOB SUTOR

## TECHNICAL ADVISORY BOARD

DAVE HOWARD, JP MORGENTHAU, ANDY ROBERTS, DAVID RUSSELL,  
AJIT SAGAR, SIMEON SIMEONOV, RICHARD SOLEY

EDITOR-IN-CHIEF: SEAN RHODY  
EDITORIAL DIRECTOR: JEREMY GEELAN  
INDUSTRY EDITOR: NORBERT MIKULA  
PRODUCT REVIEW EDITOR: JOE MITCHKO  
.NET EDITOR: DAVE RADER  
EXECUTIVE EDITOR: GAIL SCHULTZ  
MANAGING EDITOR: CHERYL VAN SISE  
SENIOR EDITOR: M'LOU PINKHAM  
EDITOR: NANCY VALENTINE  
ASSOCIATE EDITORS: JAMIE MATUSOW  
JEAN CASSIDY

## WRITERS IN THIS ISSUE

BRIAN BARBASH, GARY BROWN, JIM DUCHARME, SCOTT DURCHSLAG,  
KYLE GABHART, ANNE THOMAS MANES, NORBERT MIKULA,  
MARK O'NEILL, SEAN RHODY, STEVE ROSS-TALBOT, JAMES SNELL,  
TIM TRYZBIAK, JOE VERZULLI, VITTORIO VIARENGO

PUBLISHER, PRESIDENT AND CEO: FIAT A. KIRCAALI  
VICE PRESIDENT, PRODUCTION & DESIGN: JIM MORGAN  
SENIOR VP, SALES & MARKETING: CARMEN GONZALEZ  
VP, SALES & MARKETING: MILES SILVERMAN  
VP, EVENTS: CATHY WALTERS  
VP, BUSINESS DEVELOPMENT: GRISHA DAVIDA  
CHIEF FINANCIAL OFFICER: BRUCE KANINER  
ASSISTANT CONTROLLER: JUDITH CALMAN  
ACCOUNTS PAYABLE: JOAN LAROSE  
ACCOUNTS RECEIVABLE: JAN BRAIDECHE  
ACCOUNTING CLERK: BETTY WHITE  
ADVERTISING ACCOUNT MANAGERS: MEGAN RING  
ROBYN FORMA

ASSOCIATE SALES MANAGERS: CARRIE GEBERT  
ALISA CATALANO  
KRISTIN KUHNLE

CONFERENCE MANAGER: MICHAEL LYNCH  
SALES EXECUTIVES, EXHIBITS: MICHAEL PESICK  
RICHARD ANDERSON

SHOW ASSISTANT: NIKI PANAGOPOULOS

ART DIRECTOR: ALEX BOTERO

ASSOCIATE ART DIRECTORS: CATHRYN BURAK  
LOUIS CUFFARI

ASSISTANT ART DIRECTORS: RICHARD SILVERBERG  
AARATHI VENKATARAMAN  
TAMI BEATTY

WEBMASTER: ROBERT DIAMOND  
WEB DESIGNERS: STEPHEN KILMURRAY  
CHRISTOPHER CROCE

CONTENT EDITOR: LIN GOETZ  
JDJSTORE.COM: ANTHONY D. SPITZER  
CUSTOMER SERVICE MANAGER: ANTHONY D. SPITZER  
CUSTOMER SERVICE LIAISON: PATTY DEL VECCHIO

## SUBSCRIPTIONS

FOR SUBSCRIPTIONS AND REQUESTS FOR BULK ORDERS,  
PLEASE SEND YOUR LETTERS TO SUBSCRIPTION DEPARTMENT.  
SUBSCRIPTION HOTLINE: [SUBSCRIBE@SYS-CON.COM](mailto:SUBSCRIBE@SYS-CON.COM)  
COVER PRICE: \$6.99/ISSUE  
DOMESTIC: \$69.99/YR (12 ISSUES)  
CANADA/MEXICO: \$99.99/YR OVERSEAS: \$129.99/YR  
(U.S. BANKS OR MONEY ORDERS)  
BACK ISSUES: \$10 EA, INTERNATIONAL \$15 EA  
[SUBSCRIBE@SYS-CON.COM](mailto:SUBSCRIBE@SYS-CON.COM)

## EDITORIAL OFFICES

SYS-CON PUBLICATIONS, INC.  
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., 135 CHESTNUT RIDGE ROAD,  
MONTVALE, NJ 07645  
PERIODICALS POSTAGE PENDING  
POSTMASTER: SEND ADDRESS CHANGES TO:  
WEB SERVICES JOURNAL, SYS-CON PUBLICATIONS, INC.  
135 CHESTNUT RIDGE ROAD, MONTVALE, NJ 07645

## © COPYRIGHT

2002 by SYS-CON Publications, Inc. All rights reserved.  
No part of this publication may be reproduced or transmitted in any form or by any means, electronic or  
mechanical, including photocopying or any information storage and retrieval system, without written  
permission. For promotional reports, contact reprint coordinator. SYS-CON Publications, Inc. reserves the  
right to make, publish and authorize the readers to use the articles submitted for publication.

All brand and product names used on these pages are trade names, service marks or trademarks of their  
respective companies. SYS-CON Publications, Inc. is not affiliated with the companies or products covered  
in Web Services Journal.

# Hype, Stealth and the Dark Side of Web Services

Written by  
Sean Rhody



**Author Bio:**  
*Sean Rhody is the editor-in-chief of Web Services Journal. He is a respected industry expert and a consultant with a leading Internet service company.*  
[SEAN@SYS-CON.COM](mailto:SEAN@SYS-CON.COM)

As the new year finally starts to take hold, we're seeing a number of interesting, challenging, and even disturbing trends in the world of Web services. One of the more interesting business intelligence reports predicted recently that Web services will hit the height of its "hype curve" midway through this year.

In case you've never seen the hype curve, it's a curve with a sharp rise at the near end, and a gradual slope downward. The beginning of the curve signifies nascent products – things that are known only to small numbers of folks and are usually still fairly immature. The height of the curve is the point of widespread interest, with some early adoption but also a great deal of wait-and-see attitude. The downturn of the curve is where the technology finally becomes mainstream. Some technologies never make it to the top of the curve; others linger there and never make it to acceptance.

According to my calendar, if the height of the curve for Web services occurs this summer, we're looking at 18 months total from inception to widespread interest. By way of contrast, my calendar has Java taking about five years to reach that point, somewhere around 1996.

Now, some of my more cynical friends would say that's because Java has legs whereas Web services is an 18-month flash in the pan. I don't think so, and that's because of another trend in Web services – "stealth adoption."

We saw this in the Java world, when people began using EJB servers. Early on, the application server vendors had tremendous difficulty providing references – their customers were reluctant to be seen as on the cutting edge. This is all part of that same chicken and egg syndrome that affects every new technology.

Well, stealth adoption is taking place even as we speak. I spend a good deal of editorial time working with vendors in this space, and my discussions with them lately have all had the same quote: "We've got a big account, but they're not willing to speak about it yet." In other words, stealth adoption. Which is natural, but frustrating. Still, this trend points to Web services making it down the other side of the curve toward broad adoption. It's not a guarantee, naturally, but it does bode well for its future.

Then, of course, there's the seedy side of Web services. January 2002 saw the first Web services virus. Although it was only a lab virus, one conceived by a white knight hacker and sent to Microsoft only, as a warning of a security hole, it's yet another sign of broader interest and adoption – people don't write viruses that they expect no one to use (so to speak). It's also a wake-up call for all of the application vendors who have been ignoring or sidestepping the Web services security issue – it needs to be solved. Which is why, of course, we're focusing on security in this issue.

Security continues to be the straw that may break our camel's back. Microsoft's .NET initiative in particular needs to be bulletproofed, both to overcome the Windows platform's reputation for security lapses and to entice users into providing sensitive information, such as credit card data, in their Passport. Without a feeling of absolute security, very few people are going to gamble their credit history in exchange for ease of use.

The Java world also needs to address the issues of security. In reality, security is an issue for the entire platform. E-commerce needs authentication, encryption, and nonrepudiation in order for Web services to work effectively. Right now, the specifications are somewhat agnostic to this fact. As with Single Sign-On, this provides an opportunity for security vendors to ply their wares in a new arena and to come up with measures that effectively protect the consumer and businesses.

It's that time again as well – time to begin thinking about increasing your Web services skills and networking with other professionals. This year our Web Services Edge 2002 East conference will be held in New York City, at the Jacob Javits Convention Center, June 24–27, 2002. This promises to be a fantastic event, with topics on .NET, Java, XML and business and project management ([www.sys-con.com/WebServicesEdge2002East](http://www.sys-con.com/WebServicesEdge2002East)). I'm also pleased to say that we're colocated with TECHXNY/PC Expo this year, so there are even more reasons to attend the conference. Enjoy this issue, and see you in June. ☺







*AUTHOR BIOS:* Steve Ross-Talbot is CTO of SpiritSoft, Inc.,  
Gary Brown is Advanced Technology Architect, SpiritSoft, Inc.  
STEVE.ROSS-TALBOT@SPIRIT-SOFT.COM  
GARY.BROWN@SPIRIT-SOFT.COM

**T**o the uninitiated, those unversed in the art of computing, 'building to scale' sounds like something architects do when they take one of those models to a board meeting for approval. It sounds a million miles away from what we understand in computing.

Just for the record what *do* we mean when we talk about "building to scale"? We mean that the computer system (the software and hardware) performs predictably under a documented range of conditions. Oddly enough it isn't any different for the architect with the model or an engineer creating some new valve for a process control plant. Good practice and common sense are in short supply and not often used.

Building something simple and making it scalable requires an understanding of the dimensions of the problem. So a search algorithm that's bounded may be best implemented by a simple bitmap, whereas one that's not bounded can't use this technique. Imagine trying to deliver a complex system that scales. The complexity itself makes it more difficult for many people to see the shape and dimensions of the problem they're trying to solve. Web services enables us to focus on the problem by reducing the noise and clutter through the adoption of standards. In this article we'll be applying good practice and common sense to expose some of the design patterns that ensure Web services can be scaled to deliver predictable performance over a range of conditions. The use of the Java Messaging Service API (JMS) and the emerging Java Caching API (JCACHE) are complementary to Web services and form part and parcel of a broad architectural solution space.

### The Basic Technologies

Let's delve into a bit of history. But before we do, it's worth stating why history is so important. From the early days of computational science the progress that's been made has been firmly predicated on all that happened before. The saying, "No experience is a bad experience; it's at least a good experience of a bad experience" really tells us to value those things that clearly work and to understand those that don't. The things that don't work often tell us more than those that do. So history is important because it offers us experience and allows us to draw on that experience to forge new solutions, which are often previous experiences recast in a different setting.



## MARRYING THE JAVA MESSAGING SERVICE AND JAVA CACHING OFFERS A SOLUTION TO THE VERY REAL PROBLEMS INHERENT IN DELIVERING LARGE-SCALE DISTRIBUTED SYSTEMS

Distributed systems are what Web services are all about. Fundamentally they're about passing information from one system to another to achieve some goal. What they offer is an open standards-based canvas upon which to paint a solution. Web services deal with standards to describe what service is offered is how to connect to that service, and what protocol is used to enact some business goal. The fundamental technologies behind Web services are the Simple Object Access Protocol (SOAP), the Web Service Description Language (WSDL), and the Universal Description and Discovery repository (UDDI).

### SOAP

SOAP is an XML schema for encoding function calls and their parameters. It has its genesis, unsurprisingly, in XMLRPC; which is an XML encoding for remote procedure calls. SOAP is at the core of Web services technology. The SOAP protocol is synchronous – although an asynchronous mechanism can be built on top, similar to those in the 1980s when X-Windows and call-backs were all the rage. Using a CORBA analogy, SOAP is akin to an XML encoding of the IIOP protocol.

### WSDL

WSDL is an XML encoding for the description of Web services. A WSDL description includes information about what protocols (HTTP, HTTPS, JMS, JAXM, and so on) a Web service is prepared to accept. WSDL also records details of the Web service itself, such as the SOAP requests it offers. In effect it is the IDL of Web services.

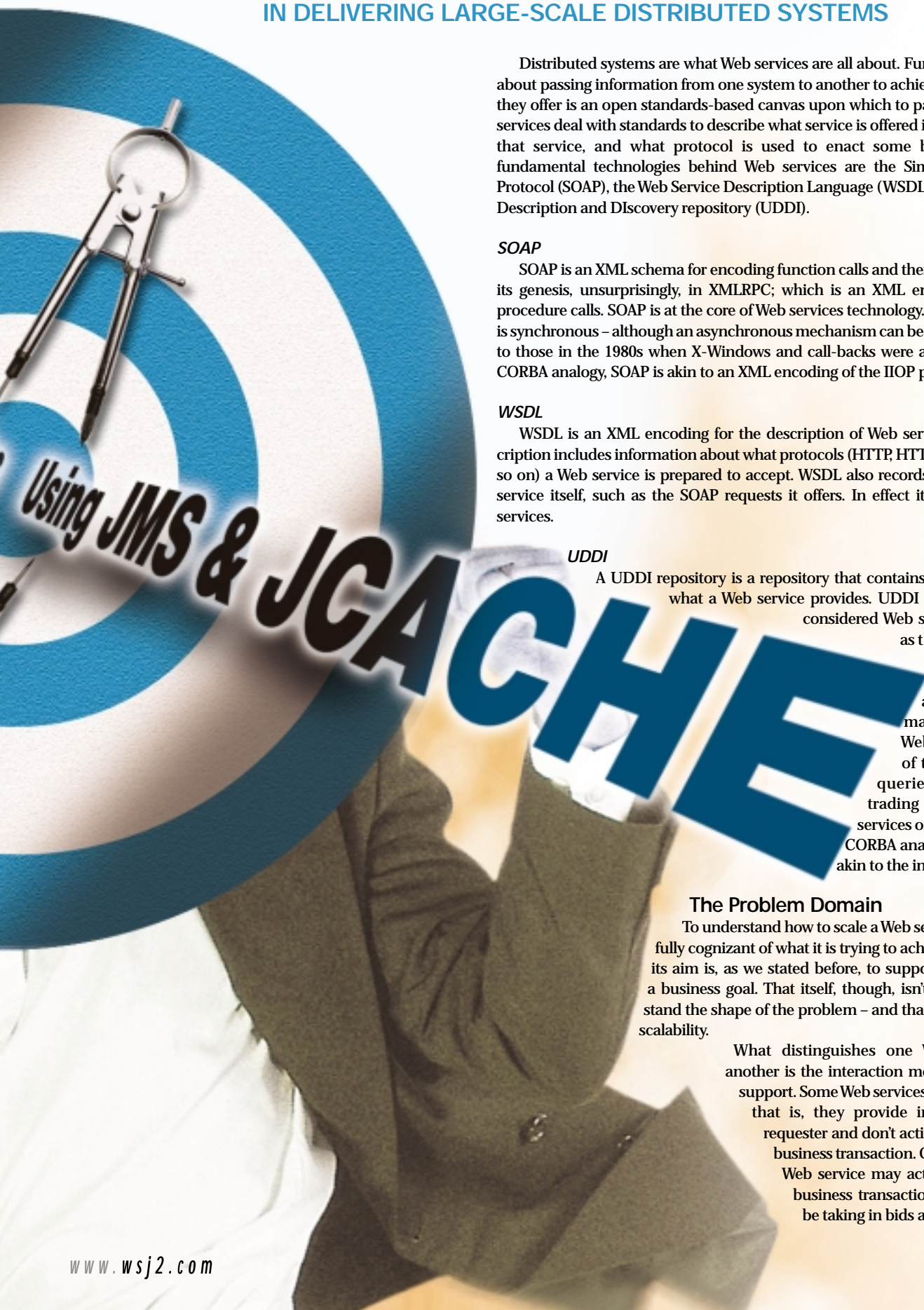
### UDDI

A UDDI repository is a repository that contains information about what a Web service provides. UDDI repositories can be considered Web services themselves, as they offer SOAP interfaces. A UDDI repository acts as a broker of information about various Web services. Users of the repository run queries against it to locate trading partners or Web services of interest. To use the CORBA analogy again, UDDI is akin to the interface repository.

### The Problem Domain

To understand how to scale a Web service we need to be fully cognizant of what it is trying to achieve. In its raw form its aim is, as we stated before, to support the execution of a business goal. That itself, though, isn't enough to understand the shape of the problem – and that shape is the key to scalability.

What distinguishes one Web service from another is the interaction model it's required to support. Some Web services may be read-only – that is, they provide information to the requester and don't actively participate in a business transaction. On the other hand, a Web service may actually be running a business transaction for you. It might be taking in bids and offers, matching





them, and reporting back the trade confirmations as they happen.

In the case of a *read-only* Web service, it might be providing price changes from a wire feed for financial service trading. On the other hand, it might be providing a calendar service to check the working days between two dates, the result of which is used to compute interest accrual. These two examples illustrate different interactions. The demand for pricing information is huge whereas the demand for calendar information is much smaller – because many requesters will monitor price changes, but only when a trade is being executed will a request be sent to the calendar.

In the case of a *transactional* Web service it might provide bids and offers for something quite simple in which the response (the confirmation) is a single response. In a more complex scenario, several partial confirmations might well occur, all of which make up the business transaction. These examples also exhibit different characteristics. The former only requires a request/ response pair since the order is the request and the confirmation is the response. The latter requires state information to be retained by the requester and Web service. The order is still a request, but now the responses need to be coordinated and matched so the correct confirmations are matched with the correct order.

## Architectural Patterns

What can JMS and JCACHE do for us? They can help us, as architects, to connect applications pretty effectively. They can even help us reuse legacy-messaging investments through a more open JMS framework approach and so provide standards-based multiplex asynchronous connectivity, a key component in addressing high fan-out delivery of information and asynchronous interaction. SOAP, together with JMS, gives us a standard synchronous functional protocol over an asynchronous communication mech-

anism for Web services. JCACHE provides a developer with a set of interfaces that, coupled with JMS, provides the bedrock for elaborating standard architectural patterns that are geared to alleviating the problems of server bottlenecks.

If we look at the technologies described so far, they fall into two camps: there are those based on a (request/response) synchronous paradigm, and those that fall into a (publish/-subscribe) asynchronous paradigm. Messaging is generally asyn-

chronous. It has to be to promote fire-and-forget messaging and so support a decoupled solution. But applications are generally built using synchronous mechanisms. The challenge is how to combine these paradigms into an architecture that engenders scalability. In order to do this we need to understand the problems. In part they're a consequence of history, which is why history is important.

### The Role of MOM and JMS

The Java Messaging Service API was born in late 1997 and the first commercial implementation (from SpiritSoft) came out during Easter of 1998. Since then a wider market has grown up around the standard as Fiorano and Progress (now Sonic Software) entered the marketplace. IBM offered a JMS interface to MQSeries shortly after and finally, late in 2001, Tibco Software joined the club.

The adoption of JMS as the first vendor-independent MOM standard testifies to the power of asynchronous messaging as a fundamental technology for delivering distributed solutions. The move toward message-driven beans and the mandatory status of JMS in J2EE 1.3 is a huge step forward – the ability of many JMS products to offer a scalable, load-balanced solution to the distribution, marshalling, and management of requests to an application server supports this. But it's only the start. Life gets much more interesting when you start to apply MOM technology through a JMS standard to caching and so start to understand the basic architectural patterns that it promotes.

### What Does JMS Do For Us?

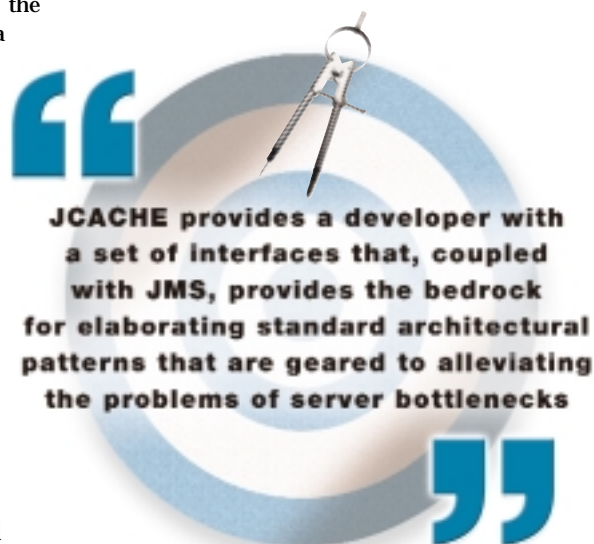
So what is JMS and what does it do for us? The Java Messaging Service is an API, an application programming interface. It's defined as a set of interfaces for producing and consuming messages based on both

publish/subscribe notification and point-to-point queuing models. The JMS specification enables subscribers or receivers to specify a quality of service for messages, so that messages can be reliably delivered or guaranteed. The specification also describes some semantics that JMS vendors must conform to. Amongst these are semantics that dictate how a message is received. Thus a queue allows one receiver and one receiver only to receive a particular message (i.e., it's delivered once and only once), while many subscribers to a topic might receive the same message. What JMS does for Web services is to provide the basic asynchronous delivery mechanism for high fan-out data (like prices and calendar information) as well as to provide the transactional secure delivery of orders and confirmations.

### JCache and the Role of Caching

Although JCACHE (JSR107) is still making its way through the Java Community Process, several vendors have announced or will announce caching products based on it. What JCACHE will provide is a standard set of APIs and some semantics that are the basis for most caching behavior. According to its functional requirements, JCACHE “allows applications to share objects across requests and across users and coordinates the life cycle of the objects across processes.”

JCACHE provides a fairly rich set of APIs, enabling control to be exercised over cache loading, cache eviction, and cache validation. It deals with basic caching patterns in which caches can be fed by other caches. A consequence of this is that it allows implementers to take advantage of distribution technology such as that offered by JMS as well as any other distribution technology deemed appropriate.





# **iWay Software**

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

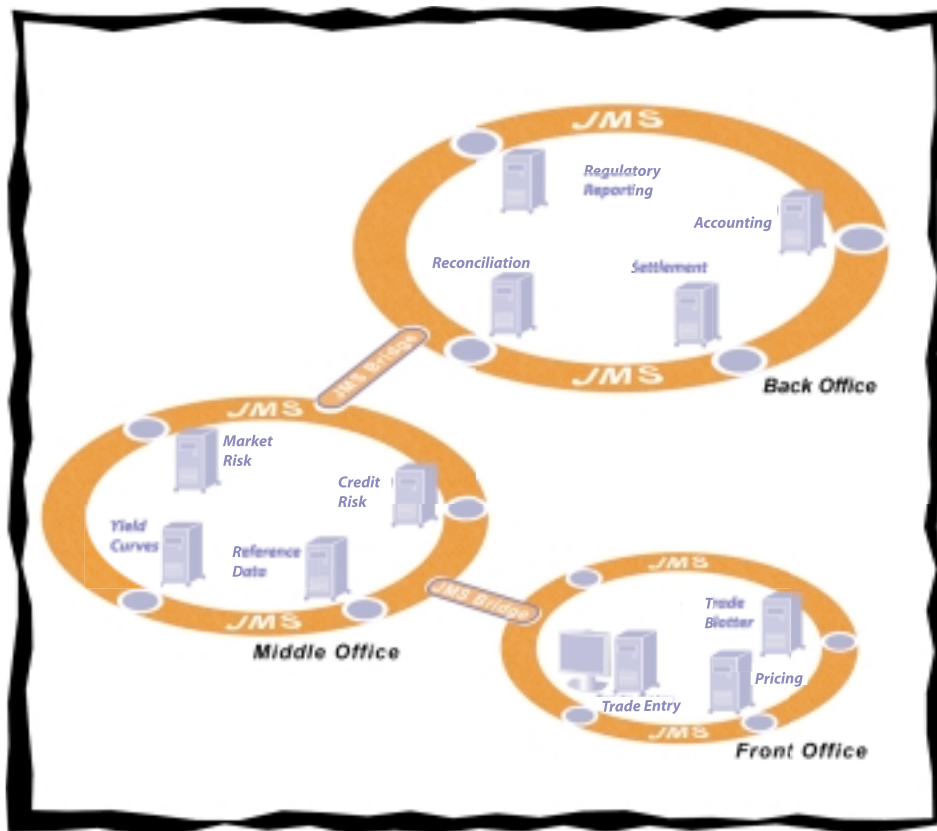


FIGURE 1 | Transactional islands in an investment banking scenario

According to JCACHE, a cache is “specific to each process.” This seemingly simple statement has profound implications and is one of the reasons why the proposed solution is flexible. Each process can implement its own mechanisms for cache loading and replacement or share the necessary functionality as required. When an object is invalidated or updated in one process, the name and associated information can be broadcast to all other instances of the cache; this is where JMS comes to the fore, by acting as the standard distribution interface for a cache. And this in turn allows the entire system of processes to stay synchronized without the overhead of centralized control.

The long and the short of it is that JCACHE is a smart approach for read-only data that’s essential to both the browsing nature of passive business transaction involvement, and to the repeatability of business transactions and active involvement of read-only data.

### Shape

Understanding a problem is all about understanding its form or shape. When we fully appreciate the shape of a problem, we can find a solution that truly fits it. It’s the difference between effecting a cure and merely relieving symptoms. All of us have experienced being pressed for time on a software project. A bug has been holding us up for days so we

decide to change this or that based on intuition...and the problem goes away. We may well have fixed the problem but, because we don’t understand the shape, we can’t be sure. On the other hand a bug holds us up for days and suddenly, perhaps because we were not looking too hard, we have further insight and understand the shape of the problem. It’s what John Bentley calls “an aha! moment,” when we truly understand the nature of a problem and so the solution hits us in the face.

The problem with the synchronous and asynchronous paradigms is that enterprises are neither totally synchronous nor totally asynchronous. They reflect both paradigms in specific ways that underpin their business model. The shape is there and the architecture needs to reflect that shape. In this way we can build systems that reflect and support the business rather than the business being a reflection of the computer systems that support it. A synchronous solution would necessitate all components of an enterprise to participate in some form of distributed transaction. This is a

legacy from history, encompassed by the old mantra that “all solutions start with a database,” whereas solutions are bounded by a problem.

### Server Bottlenecks

Request/response systems are, by their very nature, pull-based and are passive. The applications pull data from the server. Many of these applications manipulate the same logical data many times over within the same time period and across many transactions. As the client population increases, so does the number of requests to the application or DBMS server. The server becomes the bottleneck, swamped by too many spurious requests. Application server, DBMS, and hardware vendors suggest using replication to alleviate the problem and thus reduce the client-to-server ratio. But this just relieves the symptoms and does nothing toward effecting a cure. As the enterprise grows, so the need for more replication and more servers will grow – and so the costs will escalate.

Web services don’t fare any better, because they’re synchronous. They do help deliver systems, getting you there fast, because they offer standard protocols and standard interfaces, but they do nothing to alleviate the inevitable problems of server bottlenecks in large scale Web services within and between enterprises. The more complex the interaction model, the more obvious this becomes.

### Transactional Islands

The first pattern is a reflection of how many organizations can be broken down into logical units. In banking this was always based on splitting the enterprise into front, middle, and back office functions. Each functional unit owns its own transactional space. Each transaction space is then connected by a JMS-bus. Depending on the semantics of the

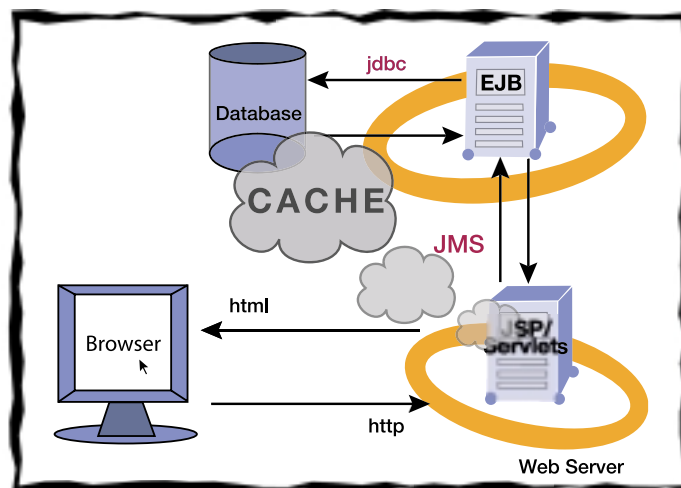
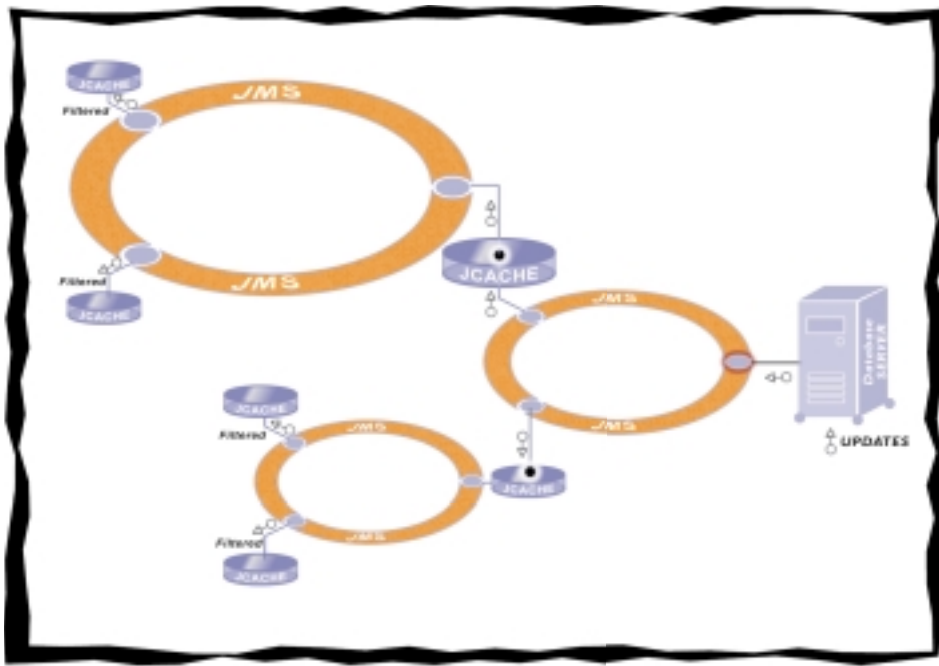


FIGURE 2 | Simple caching

# **Sams Publishing**

**[www.sampublishing.com](http://www.sampublishing.com)**





messages published on the bus, different qualities of service (QoS) may be used. Thus for an order we would want to ensure it is always delivered, but for a price change we might be willing to accept the last price published (see Figure 1). In this way each functional unit of the business is responsible for its own scalability issues. The use of JMS enables them to proceed at the rate they need to. The only functional unit that dictates overall performance is the order entry system, and that's now free to enter orders without having to worry about the other functional units of the enterprise. The flow of messages (or events) can then be controlled as a system-to-system workflow and so provide a better controlling and monitoring environment.

## Simple Caching

The next architectural nuance is to recognize that many systems spend a high proportion of their time reading and then have flurries of activity as they transact and data is changed, added or deleted. During the mid 1990s when financial services took up the challenge of decoupled architectures, many also looked to use MOM to provide a distribution mechanism for data. It was a natural step to try to cache the data nearer the application that needed it and so reduce even further the server bottlenecks.

This next pattern simply recognizes that a typical layer between a consumer of messages from a JMS bus and the bus itself is a read-only cache (see Figure 2). The read-only nature of a cache is important here because the cache is decoupled from the underlying data source. Updating the objects doesn't change the source; updatability is a more advanced pattern. What this pattern requires is a container that understands what we want, how to get it and how to propagate the changes that it receives. This is the basis for traditional caching behavior.

Once we take the view that the applications can have a cache of information from the application server or DBMS, and that the cache can be up to date or active, then the application no longer needs to pull data that it has already requested. Applications in an active architecture pull once for any given object (or objects) and the server simply pushes

thereafter. This seemingly simple change has a major impact on performance and scalability and has a major impact on the design of the applications. Applications now need to become event driven and so, themselves, become active.

## Hierarchical Caching

A more complex form of caching can be derived from the simple form above. If caches can be connected in a hierarchy with each cache subscribing to a JMS bus (see Figure 3), we can use the natural hierarchical shape of an enterprise and some counterintuitive logic to model a hierarchical caching solution across  $n$ -tier architecture.

In an  $n$ -tier architecture, the nearer to the data source you are, the finer the granularity you need to identify the data you want. The further away you are, the more localized the data becomes. Consider this: as all the requests for data converge onto a data source, the data source needs to identify what each requester needs. But further out toward the edge, the requesters already know what they need, be it through object identity or through predicate definition. Thus the counterintuitive principle is to have fine grain subscription closer to the data source and coarse grain subscription further away from the source. The fact that JMS based solutions offer topic hierarchies makes them highly suited to this form of traffic shaping and complementary hierarchical caching.

### Active Queries

The more an active cache is decoupled from a server, the more scalable the solution becomes. Active queries provide a flexible way of describing what we want in our cache, through the use of predicates. This changes the way we build applications. In this fashion, subscription to a cache, and the cache's subscription to a data source, are based on these predicates. Notification of change is based on specific objects or entities, and the results set that this information is moved into (see Figure 4).

## Updatable Caches

Decoupled caches don't lend themselves to conventional methods of updating information. The very fact that they're decoupled means that the transaction consistency of the cache is sacrificed. Conflicts that arise from inconsistency need to be dealt with as exceptions in the business flow that underpins an enterprise. This is what many financial institutions do anyway and is what transactional islands recognize too. What we can do is provide patterns that ensure that the window of opportunity in which these problems may arise is as small as possible. In this

**Altova**  
**[www.altova.com](http://www.altova.com)**

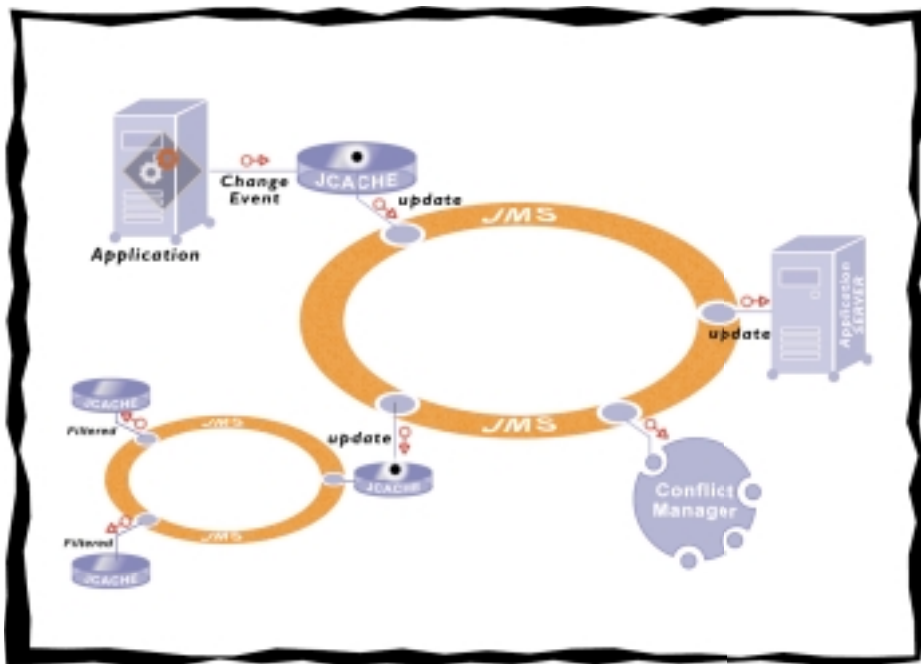


FIGURE 4 | Updatable caching using decoupled approach

way we can create an architecture in which exceptions are exceptional and most of the investment in software deals with the bread and butter.

How do we close this window? There are two basic mechanisms. The first uses the concept of a proxy to provide a synchronous update mechanism. If the business object or entity is a proxy within the cache, and that proxy has all it needs to update the information source, then the proxy can update the information source directly (in standard RMI type fashion). When the synchronous update call returns, the proxy can publish the update through the cache to other

interested parties (see Figure 5).

This pattern is fine for low-frequency updates. The synchronous calls to the information source guarantee that all updates are synchronized and so are transactionally safe, but the price to be paid is the latency of the synchronous call and the attendant server bottleneck. While the read frequency remains high and the update frequency low, this pattern has considerable merit.

The second mechanism uses a totally decoupled cache for read and update. When caches are loaded, the information source is no longer the source but is just treated as a subscriber to change. Changes occur in the network by publishing the changes onto the network but the information source uses a durable subscription to ensure that it updates the information source. This does carry with it a need to perform conflict resolution when updates to the same object overlap; this can be easily detected by carrying the pre and post images of the object – but the conflict resolution is application defined.

This pattern works for both higher read and high update frequencies. But it does incur the cost of conflict resolution. As long as the number of conflicts is low, the conflicts can be costed as part of the

overall exception handling that systems generally have to perform. If the update profile is segmented so that different parts of an organization update different data, then the pattern is very effective in increasing overall throughput in the face of change.

## Moving Up the Stack

How does all this fit into the real world? All the patterns mentioned, from transaction islands to decoupled updatable caches, have relevance in today's Web applications, as well as having relevance to the Web services of tomorrow. The inability of Web servers to cache information effectively suggests that caching based on a JMS distribution model is highly applicable. The same would be true of a SOAP-based Web service or for any internal application in which reading is a large part of what the app is used for.

What JMS has done is to trigger the debate and move us up the stack. We're now able to identify patterns and so expose further APIs and develop the related JMS tools market. If you like, it has moved the debate beyond JMS in the same way that the introduction of SQL gave rise to database tools in the 1980s.

## In Search of Greater Flexibility

The marriage of JMS and JCACHE and the use of topics and selectors provides an open standards-based solution to a common architectural problem. What it does is allow us to abstract out some of the architectural patterns that underpin the construction of large-scale distributed systems. For example, the JMS selectors are what enables the messaging and caching to deliver what we need based on a predicate. It's simply the ability to distribute change events, apply a condition to the change, decide what to change in the cache, and then inform an application of the change to the cache. It's a case of managing events, conditions, and then actions – and applying it to JMS and JCACHE. Flexibility, the provision of runtime change to such approaches, is what we all seek.

## Summary

What we have shown is that the coupling of JMS and JCACHE offers a solution to the very real problems inherent in delivering scalable Web services. The patterns we've described are a testament to this. Furthermore we've shown that such solutions can be flexible by adding a RuleML ingredient to the architecture to provide personalized caching, which moves us closer to reflecting the changing needs of users and so keeping them at the core of our architecture. ©

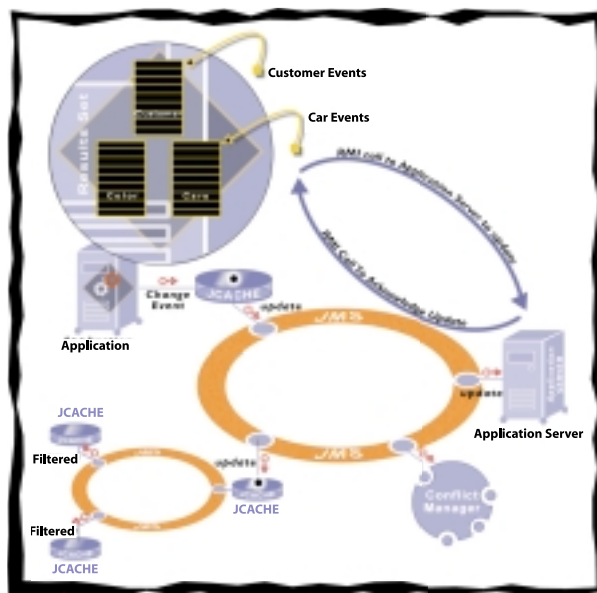


FIGURE 5 | Updatable caching using synchronous mechanism



# **XML Global**

**[www.xmlglobal.com/newangle](http://www.xmlglobal.com/newangle)**

# Secure Web Services

*Could 'trust certificates' be a step toward making electronic marketplaces and collaborative commerce initiatives more appealing?*

**S**ecurity concerns, especially since the events of last Fall, are at the center of many industry discussions. Ever-increasing reports of hacker activities and security holes in well-known software products further fuel the debate, and rightfully so.

Web services is a great new technology that will form the underpinning for electronic business of the future. So making Web services secure should be, and is, one of the activities our industry needs to focus on most. Beyond the very basic aspects of security, reliability, authentication and nonrepudiation issues, however, another related issue also deserves to be looked at: trust.

## Security, Trust, Web Services, and E-Business

Trust will in my view be a critical factor in the further adoption of Web services. When I say trust, I really mean more than just trust as "created" through various security technologies. Rather, what I mean is trust much more in the ordinary sense of the word.

## A World of Web Services Built on TRUSTe?

To find a past case where trust issues and information technology intersected, we have to look no further than at the early days of the Internet and Web-based commerce. As detailed in the *Online Privacy Resource Book* published by the not-for-profit organization TRUSTe ([www.truste.com](http://www.truste.com)), issues around trust deficiency and its role as a potential inhibitor to building a worldwide electronic community of open business relationships

were already identified in the very early days of the Web.

In the world of TRUSTe, a program designed around the philosophy of self-governance, four core requirements form the basis for its rules and procedures:

1. **Notice:** A Web site must declare what kind of information will be gathered about a user



2. **Choice:** A user's ability to freely permit (or not to permit) data gathering
3. **Access:** A user's ability to verify what data has been collected and ensure its accuracy
4. **Security:** The guarantee that a user's data is collected and stored in a secure manner

In the future, in the ultimate world of dynamic business webs, we'll see Web services-based agents dynamically publish, discover, and integrate services from third parties around the globe. Many of these services may

be provided by previously unknown (and also very often hidden by the user) service providers. Since all the actions of these agents are done on behalf of the user or corporate entity, the exact same challenges encountered by on-line commerce will also be faced in dynamic business webs. And similar solutions may apply as well.

In the world of TRUSTe, a Web site posts a logo on its pages that indicates that the operating company has a contractual relationship with TRUSTe and that it follows TRUSTe's policies on privacy and conflict resolution. Similarly, we may now see Web services provide a digital credential that would indicate to a service consumer that it follows similar policies and procedures for trusted Web services.

## Components of Trust Certificates

For the purpose of this discussion, let's call these certificates "trust certificates" and see what some of the potential components for trust certificates could be, and how and where trust certificates would be used.

## No Trust Without Security

The fourth pillar of TRUSTe's policies, security, would certainly be at the core of any trust certificate. Security is very broad discipline and involves diverse and plentiful challenges, but the different standards and specifications that address them are equally numerous. Indeed, many technologies in this area have either been developed already or are being developed as I write.

To secure the communication between a Web service and its clients, we can use the infrastructure already developed for the "old" Internet – for example, HTTP over SSL (Secure Sockets Layer) or more recently TLS (Transport Layer Security) for secure data transfer (see Figure 1). HTTP Basic Authentication, to authenticate users, or rather user agents, has also been with us for quite some time now (for too long many security experts would probably argue).

In the new world of Web services and SOAP,



### Author Bio

Norbert Mikula is industry editor of *Web Services Journal* and has more than 10 years of experience in building and delivering Internet and e-business technologies. He serves as vice-chairman of the board of directors of OASIS, is recognized internationally as an expert in Internet and e-business technologies, and speaks regularly at industry events.  
[NORBERT@SYS-CON.COM](mailto:NORBERT@SYS-CON.COM)

# SilverStream

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



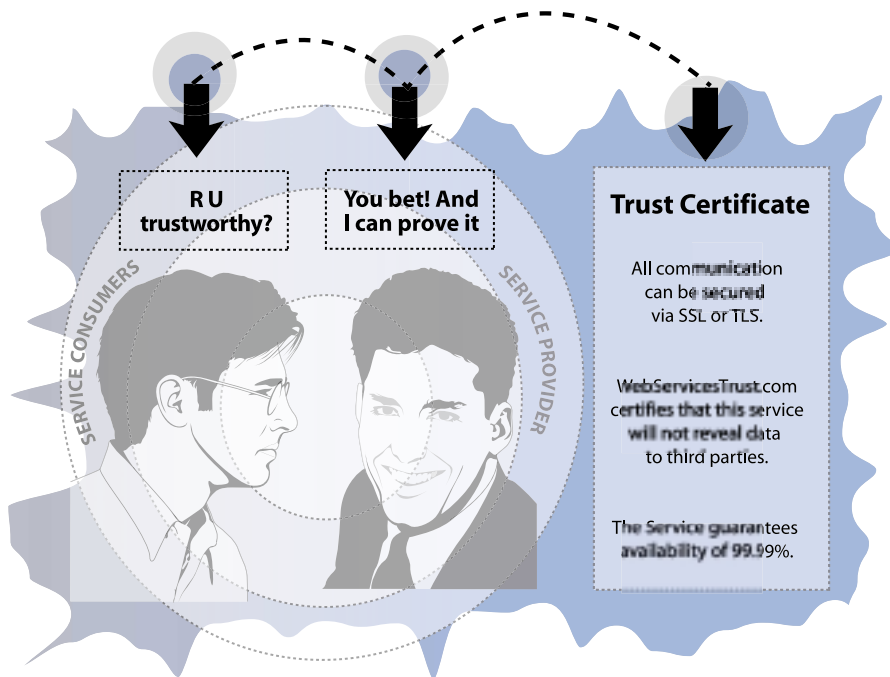


FIGURE 1 | A trustworthy service

many new and important specifications are being developed. SOAP DSIG (Digital Signatures) is an important, more recent, specification that describes how SOAP messages (or parts thereof) can be digitally signed in order to ensure the integrity of a message. Furthermore the multiparty nature of SOAP message routing will require approaches that allow us to encrypt parts of the message while leaving other parts in plaintext.

If we go beyond these still very basic protocols and specifications, we find OASIS's XACML (eXtensible Access Control Markup Language). The goal of the XACML technical committee is to develop "an XML schema for representing authorization and entitlement policies." Also at OASIS, we find SAML (Security Assertions Markup Language). One of the problems SAML will help us solve is nothing less than the issue of single-sign-on across (potentially) hundreds of thousands of services around the world.

A Web service's compliance with one or more of these specifications would represent a big portion of a trust certificate.

#### Policies for Data Collection

Whenever services communicate, an exchange of user-specific (or corporation-specific) information will be necessary. Who controls this process will be as important an

issue for dynamic business webs as it already is for Web sites (see again TRUSTe) and providers of Internet-wide single sign-on services and identity management such as the Liberty Alliance ([www.projectliberty.org](http://www.projectliberty.org)) and Microsoft's .NET Passport ([www.passport.com/](http://www.passport.com/)).

In the world of Web site and Intranet security, it's becoming common to go beyond simple access control lists (ACL) and to protect and manage access to resources via policies that specify what actors are allowed to do – e.g., read a file – and under what conditions. For the purposes of Web services, we may choose to follow a similar approach and express policies that specify what data will be exposed to specific groups of services. One example would be to be very restrictive with "untrusted" service providers and to expose more information to properly authenticated and trusted services.

#### Service Quality and Reliability

As we're building distributed applications and forming dynamic business webs we may – by definition – use services outside of our control. The questions then become: how reliable is the service we're using and what service level can the provider of a service guarantee? Service Level Agreements (SLAs) and their negotiation and execution in traditional IT is already challenging enough,

but in a world of dynamic binding of services we'll have to find electronic means to express metrics and commitments as to the quality of the services offered. As a developer, I wouldn't trust a service to the extent that I'd integrate it into my software unless I could receive certain guarantees about the performance and availability of that service. (One could also envision that SLA negotiation could be tied to the billing service, so a service provider and client could automatically negotiate a certain level of service for a certain fee.)

#### A Bullish Future

The Business Software Alliance ([www.bsa.org](http://www.bsa.org)) has conducted a survey of the CEOs of its member companies. According to this survey, the group is very bullish about the future of security technologies, going as far as stating that "concerns about the security of Internet transactions would largely be resolved by 2005." Let's hope so.

#### Usage Models for Trust Certificates

Trust certificates could be part of the many diverse components that form the necessary framework for electronic collaboration between business partners. The electronic business initiative ebXML ([www.ebxml.org](http://www.ebxml.org)) – jointly sponsored by OASIS and UN/CEFACT – has put a lot of thought into how trading partners can describe their individual preferences and capabilities and how two parties can use these descriptions to come to individual agreements.

EbXML introduces the two concepts of CPPs and CPAs. A CPP is a "collaboration protocol profile" and describes the technical capabilities of a particular party. These records include information about supported transport protocols, security requirements, and messaging capabilities, as well as references to applicable business processes. A CPA, or "collaboration protocol agreement," is formed when two parties (or rather the automated agents representing these parties) agree on the individual technical parameters for this particular business interaction.

I could see trust certificates becoming yet another component that a CPA would reference or contain. When an automated agent searches for a suitable service provider (based on business and technical parameters) it could also check a particular party's trust certificate and, for instance, filter out services that don't follow the

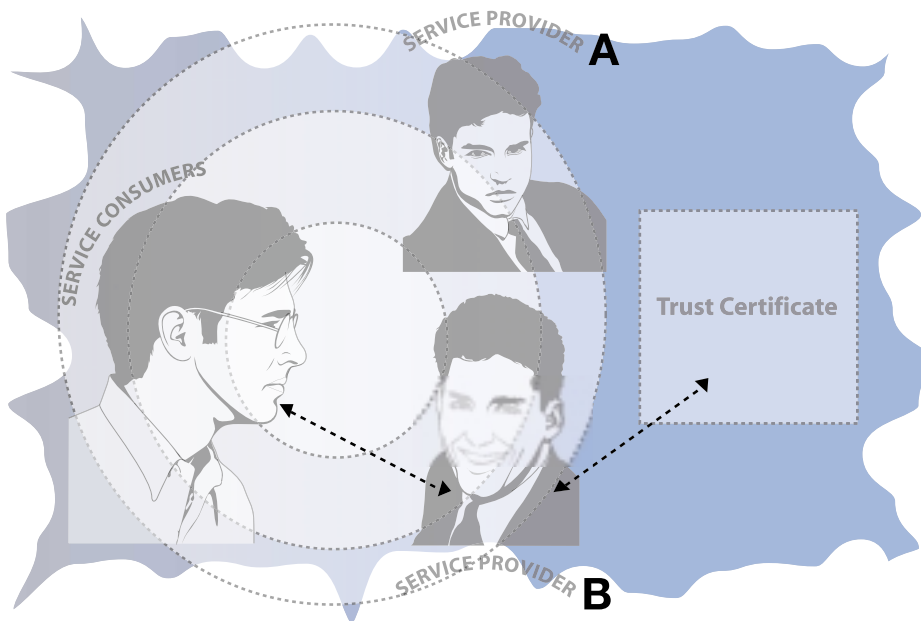


FIGURE 2 | The service filters out an untrustworthy party

required trust and privacy guidelines.

Trust certificates could also play a role in registry and repository infrastructures such as UDDI, the Universal Description Discovery and Integration initiative ([www.uddi.org](http://www.uddi.org)). UDDI uses the tModel concept, in which a tModel is essentially a pointer to a particular technical component of a service – for instance, a WSDL file.

In the future we might see trust certificates being registered with UDDI as a particular tModel and being used via UDDI search-APIs to find services suitable for consumption.

## Trust Beyond Web Services

One industry that's been plagued by issues around trust, confidentiality, and security is the electronic marketplace. Besides business issues such as lower margins and

the fear of commoditization, trust-related concerns are listed as one of the top factors that hinder the adoption of e-marketplaces.

As we're trying to increase operational efficiencies throughout the supply chain, providers of products and services are often required to share more and more information considered sensitive and business-critical in nature. Very often companies choose not to participate in collaborative commerce networks and marketplaces because to do so would require them to make public information – such as inventory levels, forecasting information, and pricing – that they'd rather their competitors didn't see. The Alliance for eCommerce Trust ([www.eyquem.com](http://www.eyquem.com)) has put together a very informative and detailed compendium about trust and B2B. (I also have the pleasure of serving on the advisory board of that group.)

Trust certificates, as proposed in this article, could be a step toward making electronic marketplaces and collaborative commerce initiatives more appealing to all participants.

## Future Opportunities

### Certification Authorities

Just as TRUSTe and other such initiatives play the role of a trusted party in the world of online commerce, certificate authorities play a similar role in the world

of encryption and digital signatures. Should the idea of trust certificates ever become a reality, then there's an opportunity for enterprises that offer services such as policy development, certification, monitoring, and dispute resolution.

### Intermediaries

A third party may start to act as an intermediary that offers services for service providers and service clients alike. Examples of these services are payment processing, hosting of registries, and maybe even quality control of the Web service interfaces they host and manage on behalf of the actual provider of the service. Some of these intermediaries may then emerge as "places to go to" in order to find Web services that can be "trusted."

Even without explicit trust certificates, certification authorities, and intermediaries, the issue of trust may be addressed in different ways. Some companies will simply become "trusted" parties and services providers because of their consistent high standards and the brand associations they develop around these standards.

### Value-Added Registries

While registries for services such as UDDI certainly provide a valuable infrastructure for service management, I can also see value-added registries that not only contain pointers to service providers but also monitor them (e.g., their up- and downtime) and track how many service consumers have engaged in long-term and/or short-term integration with a particular service. While certainly this in itself creates many questions about privacy and confidentiality, this approach would use group behavior to derive trust-related information. In other words, if a million consumers bind to a particular service A compared to 10 who bind to service B, you may derive (subjective) trust assumptions from that – such as, if a million entities trust them, why shouldn't I...?

## Summary

Security and trust will be concerns that will remain hot topics for many years to come. To begin with, every potential provider of Web services should start thinking about how they can increase the trustworthiness of their service offerings. ©



FIGURE 3 | A trustworthy intermediary

# Securing Web Services

A new software paradigm  
that offers new efficiencies—and  
new security challenges



## Web Services and SOAP

The actual definition of a Web service is a matter of some debate because the world of Web services can extend from small closed networks to global discovery services implemented using UDDI (Universal Description, Discovery, and Integration). But at a practical implementation level it is useful to think of a Web service as any software service that can be defined using WSDL (Web Services Description Language) and which uses SOAP for communication between a requester and a listener. This communication uses SOAP as the enveloping protocol.

Tools such as SOAP:Lite ([www.soaplite.com](http://www.soaplite.com)) written by Paul Kulchenko, or the IBM Web Services toolkit (<http://alpha.works.ibm.com/tech/webservicestoolkit>) allow for SOAP requests to be created simply and easily. SOAP listeners are offered in many products, including application servers such as IBM WebSphere (SOAPConnect) and databases such as Oracle 9i. Together, SOAP listeners and SOAP requesters provide a simple means of providing services to customers.

**W**eb services allow organizations to expose software services to customers, trading partners, and even the entire world, in a simple fashion. SOAP is the ubiquitous Web services protocol and is designed to be easy to use – which is why the S in SOAP stands for “Simple.” But this simplicity comes with a downside – because if security is compromised, then the benefits of Web services are wasted.

The need for security in Web services is not only related to the monetary value of goods or services that are being exchanged through the service. More abstract items such as brand value and reputation are at stake when security is compromised. In addition, a Web service may allow an attacker to reach into a company’s internal systems, and this has the potential to be dangerous, whether the Web service is commercial or not. The value of the systems and/or data that are being exposed to the Internet therefore also has to be taken into account. In short, security is a requirement for practically all Web services. This article will look at how security applies to Web services – beginning with technologies that carry over from the browser-based Internet and then moving on to new initiatives specific to Web services, such as SAML and SOAP-SEC.

## New Risks

Although it may seem obvious, it’s some-

times forgotten that Web services make use of the same Web technologies that have suffered many security compromises in the past. Web services don’t necessarily use the Web – indeed, SOAP can be sent over SMTP (i.e., e-mail) Sun’s definition of a Web service describes it as: “specific business functionality exposed by a company, usually through an Internet connection, for the purpose of providing a way for another company or software program to use the service.”

In practice, however, the vast majority of Web services rely on a Web server. If improperly configured, Web servers are vulnerable to such things as buffer overflow attacks or URL “button-pushing” attacks. Buffer overflow attacks consist of a stream of text sent maliciously to a Web server in order to load executable code into memory and effectively hijack the Web server. “Button-pushing” URL attacks are an effective way of probing for vulnerabilities in Web servers by mischievously sending bogus data to Web applications.

Many best practice papers and security patches already exist for Web servers. For example, Microsoft produced a guide to securing IIS 5.0 (the Web server for .NET) at [www.microsoft.com/serviceproviders/whitepapers/securing\\_iis\\_whpaper.doc](http://www.microsoft.com/serviceproviders/whitepapers/securing_iis_whpaper.doc). However, SOAP adds a new dimension to attacks on Web servers. In addition to buffer overflow attacks against the Web server itself, these attacks can now be directed through the Web server at the SOAP engine. For example, if a Web service expects an XML document to contain a 20-character text string as data in an XML element, what happens if a 30-character text string is sent? That’s why it makes sense for an incoming document to be parsed against an XML Schema, run through a DOM or SAX parser, in order to check that it’s valid. The old maxim of CGI script authors – never trust your input – is still valid for SOAP.

## Past Security Measures

When Netscape introduced SSL (Secure Sockets Layer) in the Netscape 1.1 browser, back in March 1995, SSL estab-



**Author Bio**  
Mark O'Neill is CTO of Vordel, where he oversees the development of Vordel's technical strategy and product development in the areas of XML and public key cryptography. Mark is a member of the OASIS Security Services Technical Committee and an advisor to the XML.org industry newsletter.  
[MARK.ONEILL@VORDEL.COM](mailto:MARK.ONEILL@VORDEL.COM)

# Macromedia

[www.macromedia.com/downloads](http://www.macromedia.com/downloads)

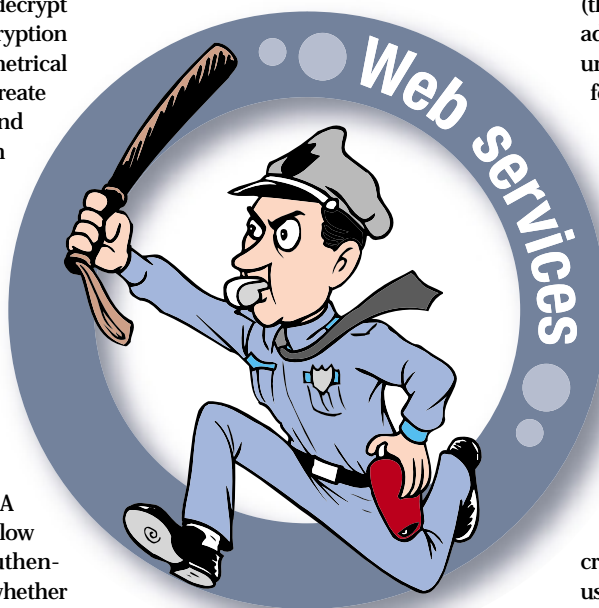


lished itself as the de facto method for encrypting data between Web browsers and Web servers. SSL creates a secure channel through which sensitive data, such as credit card details, can pass. SSL uses digital certificates to provide the transparent authentication of Web servers and, optionally, Web browsers as well. Digital certificates store the public key of a person or entity and bind this key to information about that person, and to the issuer of the certificate. A public key is used to encrypt data in such a way that only the holder of the corresponding private key can decrypt the data – using an asymmetric encryption algorithm. SSL uses the RSA asymmetrical encryption algorithm as the means to create a secure “pipe” between the browser and the Web server. After the SSL connection is established, the RC4R algorithm is the preferred algorithm for encryption and decryption of data, since it is faster than RSA.

After a number of early SSL vulnerabilities were addressed, SSL proved itself to be dependable and secure. However it still relies on careful implementation and as recently as September 2001 a vulnerability was found in the SSL implementation contained in RSA Security’s BSAFE toolkit, that could allow an attacker to bypass SSL client authentication, the process of determining whether someone or something is, in fact, who or what it’s declared to be. SSL provides this capability through the use of digital certificates stored “at the client” – i.e., with the Web browser.

Following the success of SSL for business-to-consumer (B2C) transactions, the same technology was applied to business-to-business (B2B) transactions. Initially this was focused on “marketplaces” – in which a person manually keyed information into a Web site for product procurement. More recently, it

was realized that the real value for B2B e-commerce was for “lights-out” data traffic not involving Web browsers at all. For over 15 years, electronic business documents have been sent over EDI networks that are “fenced off” from the Internet. However, these networks tend to be expensive, so it’s tempting to use the public Internet instead. XML-based Web services encompass and surpass many of the capabilities of these



private EDI networks, especially in the area of interoperable document types and service discovery. However, in order for Web services to truly recreate the value of private networks on the public Internet, communications confidentiality is required, along with authorization and access control, communications integrity, and an audit trail.

## Digital Signatures

SSL provides the first item and a portion of the second – so let’s examine the rest...

SSL is found wanting in the area of audit trails. It’s difficult to prove that an SSL session occurred, or even to prove that a piece of data has been received using SSL. This is where digital signatures come into play: digital signatures use the same underlying technology as encryption, but rather than hiding data from prying eyes, digital signatures prove that a signatory did indeed sign a piece of data (this is commonly called “non repudiation”). In addition, a digital signature protects a document from tampering. This is one key difference between a digital signature and a handwritten signature. To quote EU Directive 1999/93/EC, a digital signature is “linked to the data to which it relates in such a manner that any subsequent change of the data is detectable.”

A digital signature is an encrypted digest of a document. A digest is obtained by passing a document through a hashing algorithm, which produces a short string of bytes that’s almost uniquely linked to that document. This digest is then encrypted with the signer’s private key. Checking a digital signature involves recomputing the digest from the document, decrypting the digest in the digital signature using the signer’s public key, and making sure that the two digests match.

## xml-dsig

XML Digital Signature (xml-dsig) is a joint initiative of the IEFT (Internet Engineering Task Force) and the W3C (World Wide Web Consortium). xml-dsig defines a means of rendering a digital signature in XML. It’s often mistakenly assumed that XML Digital Signature is just for digitally signing XML documents, whereas in fact it is for signing “any digital content,” to quote the specification at [www.w3.org/Signature](http://www.w3.org/Signature). xml-dsig brings the well-known benefits of XML to digital signatures, making them human-readable, easily parsed, platform independent, and generally more simple to implement than preceding standards like PKCS#7. In addition, xml-dsig defines a method for multiple documents to be referenced in one signature, and mandates that information about the encryption algorithms used is also signed.

An XML Signature may be “enveloped,” in which case it’s located in source XML, “enveloping,” in which case it wraps around the

**“SOAP adds a new dimension to attacks on Web servers. In addition to buffer overflow attacks against the Web server itself, buffer overflow attacks can now be directed through the Web server at the SOAP engine”**

# Sitraka

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

source XML, or “external,” in which case it’s in a separate document from the source XML. External XML Signatures may point to any URI on the Internet.

As mentioned above, xml-dsig isn’t just for signing XML. Nonetheless, when XML in particular is being signed, certain considerations apply. It’s important that, if the signing decision depends on what the user sees, then “sign what you see” is implemented. A user views an HTML page to fill out a Web form, resulting in a SOAP message being produced and digitally signed, and in this case it’s important that the visual information presented to the user is signed. That’s because the XML rendered as HTML may depend on fonts or inline images, which make it very different from the underlying XML.

Digitally signing a SOAP message presents special requirements. Portions of the Header of a SOAP message may change when the message is routed between Web services. If these are signed, then the signature is guaranteed to break. An example of a volatile SOAP Header element is the actor element, whose data changes when a SOAP message is routed upstream to a further SOAP actor.

## SOAP-SEC

There are two separate proposals that describe techniques for signing a SOAP message. The first is SOAP-SEC, submitted to the W3C on February 9, 2001, by IBM and Microsoft. SOAP-SEC defines a method of signing SOAP 1.1 messages using a new header entry called SOAP-SEC:Signature. Two existing SOAP header items are reused for SOAP-SEC: “actor” to indicate the recipient of a header element, and “mustUnderstand” to indicate whether or not an application must attempt the validation of the enclosed XML Signature.

An example of a SOAP-SEC signed message is shown as Listing 1. Here we can see that the XML Signature in the SOAP message is ref-

erencing the signed data by using the “WhatWeAreSigning” ID. In addition, we can see that a Canonicalization algorithm is used. Canonicalization is needed for XML Signature because an XML document can change slightly but remain as valid XML. Examples of changes to an XML document include DOM processing, which may remove unnecessary white space between elements, or simply the differences between line endings between UNIX and Windows systems. Any of these seemingly minor changes to a document invalidates its digital signature. Prior to being signed, an XML document must therefore first be converted to a canonical form by removing spaces between elements and any platform-specific data.

## WS-Security

On October 23, 2001, Microsoft produced a draft of a specification called WS-Security. WS-Security defines XML elements called <credentials>, <integrity>, and <confidentiality> that can be used in a SOAP message for, respectively, digital certificates, digital signatures, and encrypted data. As such, WS-Security is a much more detailed specification than SOAP-SEC and encompasses more than just digital signatures. It remains to be seen if WS-Security will replace SOAP-SEC.

## SAML

Security Assertion Markup Language (SAML) is an OASIS ([www.oasis-open.org](http://www.oasis-open.org)) initiative that aims to define a standard way to securely exchange authentication and authorization information for Web services. Organizations may require that their Web services be accessed only by authenticated users. This access to authenticated users may well become as important in the future as access to suppliers or markets.

SAML breaks down a bottleneck in access management by allowing users to authenticate

to one authentication service, and carry their credentials to another service. There’s no need for users to reauthenticate in order to access further services. Similarly, there’s no special requirement for all users of a Web service to authenticate in the same manner (e.g., by using only username/password or using only client certificate). Until now, access management and single sign-on products were limited to single organizations and/or administrative domains. SAML extends these products across the Internet by defining XML Schemas for assertions. These assertions are sent as SOAP messages over HTTP, and refer to decisions regarding authentication, attributes, and authorization.

To put this into context, Listing 2 gives an example of each type of assertion. As the listing shows, assertion is used to indicate that the user has proven their identity (i.e., authenticated) to a given authentication authority, or security domain. In this case, a password was used to authenticate. The assertion is created by the authenticating security domain in order to indicate to a second domain that the user was authenticated. In addition, information about the lifetime of the assertion is included in the “NotBefore” and “NotAfter” elements. SAML uses XML Digital Signature to ensure that this message is uniquely linked to the issuer, and so that any changes to the message can be detected.

Listing 3 shows an example of an Attribute Assertion. This assertion asserts an attribute of the user – in this case their SubscriptionStatus. Again, XML Digital Signature can be used in order to ensure that this message is uniquely linked to the issuer, and cannot be changed without detection.

Finally, Listing 4 shows an Authorization Decision Assertion. This is used to assert the result of a policy – in this case the permission to access a Web page. Here we can see the Web page referenced in the “Resource” element.

SAML 1.0 was expected to be submitted as a recommendation to OASIS on March 1, 2002. Of the items deferred from SAML 1.0, the most notable is Microsoft Passport integration. Microsoft Passport will most likely be the most widely used authentication system on Earth. Integration with Passport will be crucial to SAML’s success beyond being a system for linking access management products across enterprises. The OASIS Security Services Technical Committee has already committed to further development of SAML beyond version 1.0.

**// Organizations may require that their Web Services are only accessed by authenticated users. This may well become as important in the future as access to suppliers or markets”**

# **New Atlanta Communications**

**[www.newatlanta.com](http://www.newatlanta.com)**

## XACML

XACML (XML Access Control Markup Language), also developed by OASIS, is a technology complementary to SAML that allows access control policies to be expressed in XML. When a SAML assertion is received by a Web service, the Web service sends a request to a SAML PDP (Policy Decision Point), which checks an XACML policy via a PRP (Policy Retrieval Point). The advantage of using XML for access control means that policies from various access control products can be replicated easily, using XACML as a common data format.

## XML Encryption

XML Encryption is a proposal that is being developed by the W3C – see [www.w3.org/Encryption](http://www.w3.org/Encryption). Just as XML Signature is more than just a means to sign XML, XML encryption is more than just a means to encrypt XML. It expresses meta-information about a signed digital document, so that a processor of that document is aware of what algorithms were used to encrypt the document.

XML Encryption allows for the encryption only of what must be kept confidential, allowing a business systems to continue to use, as before, the unencrypted data. Element names can be encrypted alone, or with the data contained within them, or both. Sometimes it's preferable to encrypt just the data itself, not the XML element surrounding the data, because a schema or DTD contains information about the name of the XML element that has been encrypted. This gives an attacker a valuable clue to use in a brute-force decryption attack.

XML Encryption and XML Signature are used together when a document is both signed and encrypted. For example, a SOAP message may be signed using SOAP-SEC and also carry an encrypted attachment. In order to check

the signature of the data, the signed document must first be decrypted by means of a transform.

This decryption transform is being proposed by the W3C at [www.w3.org/TR/xmlenc-decrypt](http://www.w3.org/TR/xmlenc-decrypt). The proposal notes that when a document is signed first and then encrypted, the XML Signature reveals the digest value of the signed resource. This is useful information for an attacker who implements a guessing attack on the plaintext data, especially since digest algorithms are suitably fast for guessing attacks. In addition, it's noted that a signature over encrypted data is not the same as a signature over the data prior to encryption – recall the discussion above on “Sign what you see.”

## XKMS

The final protocol to consider is XKMS (XML Key Management Protocol). XKMS isn't just a building block for secure Web services, it's also a means of using Web services to simplify a number of complicated PKI (public key infrastructure) protocols. As such, it is as much about applying Web services to security as applying security to Web services. XKMS incorporates X-KISS (XML Key Information Service Specification), which provides an important component of Web services security – namely a means to query the trustworthiness of a user's digital certificate. A digital certificate can be registered with an XKMS service using another subcomponent of XKMS – X-KRSS (XML Key Registration Service Specification). XKMS has been submitted to the W3C as a W3C Note – see [www.w3.org/TR/xkms](http://www.w3.org/TR/xkms).

## Combining Security Measures

UDDI has a number of unique security requirements, many of which may be addressed by a combination of XML Digital Signature, SAML, and XACML. The UDDI v1

specification makes no mention of security, so security for UDDI is presently a matter for debate. But one likely scenario would be:

1. The UDDI registry and service provider must authenticate one another so that a third party cannot register services on behalf of the genuine service provider.
2. The UDDI registry must determine if the service provider has authority to register its service. The act of publishing a service in a UDDI directory may be protected by digital signatures and, possibly, encryption, so that the service provider can't repudiate the offering of the service; and “sniffers” can't determine details of service.

Similarly, SAML and XACML allow the UDDI registry to determine if the service requester has authority to find the service registered by the service provider. Digital signatures and encryption can be used to protect the authenticity, integrity, and confidentiality of exchanged data. If the service requester is authorized to find the service, the UDDI registry signs the WSDL that's returned to the service requester. It may also encrypt the data if the service type is intended to be kept confidential. In addition, any alteration to a published WSDL interface is effectively a denial-of-service attack because the published service can no longer be discovered. Therefore, it's important that published WSDL interfaces are digitally signed. Note that the digital signature on the WSDL would often be that of the service provider, not the UDDI registry. The UDDI registry would sign the SOAP message containing the WSDL sent back to the service requester. This is an example of a signature over data which has already been signed.

## Conclusion

Many of the technologies discussed in this article remain works in progress. XML Signature is closest to ratification, XML Encryption is largely defined, but security for UDDI remains a matter for debate. It seems certain that a combination of XML Signature, XML Encryption, SAML, XACML, and XKMS will need to be used to secure Web services. As Web services pass beyond the initial pilot and proof-of-concept stage, security is vital for them to achieve the scale of mass deployment that has been widely predicted. ©

**“As Web services pass beyond the initial pilot and proof-of-concept stage, security is vital for Web services to achieve the scale of mass deployment that has been widely predicted”**



**Listing 1: SOAP message signed using SOAP-SEC**

```

<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Header>
<SOAP-SEC:Signature
xmlns:SOAP-SEC="http://schemas.xmlsoap.org/soap/securi-
ty/2000-12" SOAP-ENV:mustUnderstand="1">
<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
<ds:SignedInfo>
<ds:CanonicalizationMethod
Algorithm="http://www.w3.org/TR/2000/CR-xml-c14n-20001026"/>
<ds:SignatureMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#dsa-sha1"/>
<ds:Reference URI="#WhatWeAreSigning">
<ds:Transforms>
<ds:Transform
Algorithm="http://www.w3.org/TR/2000/CR-xml-c14n-20001026"/>
</ds:Transforms>
<ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmld-
sig#sha1"/>
<ds:DigestValue>erjwleWEWrewrfmpaeaesA</ds:DigestValue>
</ds:Reference>
</ds:SignedInfo>
<ds:SignatureValue>wfmSSasASFASqasf=...</ds:SignatureValue>
<ds:KeyInfo>
<ds:KeyName>Marko</ds:KeyName>
</ds:KeyInfo>
</ds:Signature>
</SOAP-SEC:Signature>
</SOAP-ENV:Header>
<SOAP-ENV:Body
xmlns:SOAP-SEC="http://schemas.xmlsoap.org/soap/securi-
ty/2000-12" SOAP-SEC:id="WhatWeAreSigning">
<StockOrder:buy
xmlns:StockOrder="http://www.stockorder.com/Stock">
<StockOrder:symbol>SGP</StockOrder:symbol>
<StockOrder:quantity>2000</StockOrder:quantity>
<StockOrder:market>New York</StockOrder:market>
</order:buy>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

**Listing 2: SAML authentication assertion**

```

<saml:Assertion
  MajorVersion="1" MinorVersion="0"
  AssertionID="312.5.32.2.6422421"
  Issuer="Vordel"
  IssueInstant="2002-01-05T13:32:00Z">
<saml:Conditions

```

```

  NotBefore="2002-01-05T13:30:00Z"
  NotAfter="2001-12-03T13:28:00Z" />
<saml:AuthenticationStatement
  AuthenticationMethod="password"
  AuthenticationInstant="2002-01-05T13:30:00Z">
<saml:Subject>
  <saml:NameIdentifier
    SecurityDomain="vordel.com"
    Name="marko" />
</saml:Subject>
</saml:AuthenticationStatement>
</saml:Assertion>

```

**Listing 3: SAML attribute assertion**

```

<saml:Assertion ...>
  <saml:Conditions .../>
  <saml:AttributeStatement>
    <saml:Subject>
      <saml:NameIdentifier
        SecurityDomain="vordel.com"
        Name="marko" />
    </saml:Subject>
    <saml:Attribute
      AttributeName="SubscriptionStatus"
      AttributeNamespace="http://vordel.com">
      <saml:AttributeValue>
        UpToDate
      </saml:AttributeValue>
    </saml:Attribute>
  </saml:AttributeStatement>
</saml:Assertion>

```

**Listing 4: SAML authorization decision assertion**

```

<saml:Assertion ...>
  <saml:Conditions .../>
  <saml:AuthorizationStatement
    Decision="Permit"
    Resource="http://vordel.com/news/index.html">
  <saml:Subject>
    <saml:NameIdentifier
      SecurityDomain="vordel.com"
      Name="marko" />
  </saml:Subject>
  </saml:AuthorizationStatement>
</saml:Assertion>

```

Download the code at  
[sys-con.com/webservices](http://sys-con.com/webservices)

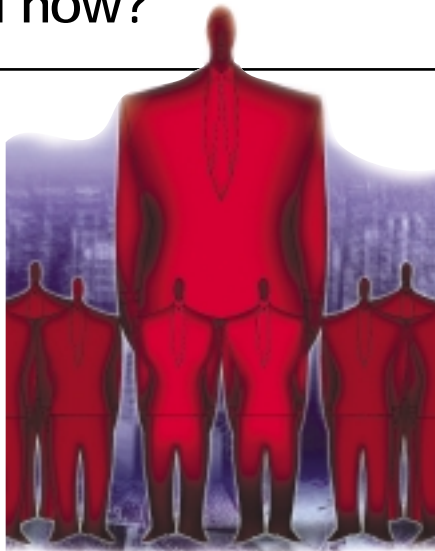
# Beyond the Hype... the Reality of Early Web Services Adoption

Who's using Web services for real  
business applications today,  
why and how?

**W**eb services have enormous promise, but not a single company today is yet fully tapping their potential.

Indeed, early adopters are experimenting through carefully controlled pilots that take advantage of the evolutionary nature of the technology, and CIOs and IT organizations – fatigued by yet another “new new thing” – are adopting a show-me attitude that requires Web services companies to prove that their offering works...and will create measurable value.

Nevertheless, in select verticals for specific applications, numerous innovative companies *are* already using Web services to execute real business processes with partners and deliver value to customers. These pioneering companies are gaining valuable experience, allowing them to create a new business architecture that will position them for long-term growth.



## Moving Beyond the First Phase

So much has been written about the promise of Web services that more words on the topic are not only unnecessary, but potentially counterproductive. The good news is that most CIOs and IT organizations have now heard about Web services; the bad news is that they are skeptical about all the hype. As recently as six months ago, a great deal of effort was being expended by early Web services providers like Grand Central Networks and Cape Clear on educating potential buyers on the benefits of Web services. Today, that's no longer necessary – thanks to the marketing machines at IBM, BEA, Microsoft, and Sun. Instead, Web services providers must focus their sales efforts on addressing concerns from large enterprises about security, return on investment (ROI), and the long-term viability of small companies in

this adverse economic environment.

It's still very early in the adoption cycle; the chasm to broad penetration across most vertical markets will probably be crossed only over the next two years.

That said, the reality of Web services adoption – while not broad – is surprisingly deep in select verticals for some specific applications. If you've tracked Web services over the last 18 months, you may remember that the consensus a year ago was that early adopters would be small and medium enterprises several tiers down in the supply chain who would prefer open standards and couldn't afford expensive, time-consuming EAI software. Instead, it's now clear that the early adopters are larger enterprises in search of more cost-effective demand chain solutions, in verticals such as banking, insurance, travel, and manufacturing. While there are hundreds of companies across many verticals conducting simple Web services pilots behind the firewall, thanks to the ease with which Web services can be created on the latest versions of application servers, there are dozens of companies in these select verticals using Web services for mission-critical applications that integrate one or more players outside the firewall.

This article will analyze the early adopting verticals, describe the most common use cases, and detail examples of real companies using Web services to execute actual business processes.

## Early Adopting Verticals

Who is using Web services for real business applications today – and why?

In today's challenging environment of shrinking IT budgets and scarce resources, deciding where in the organization to first deploy Web services is an issue for all IT managers.

Who are the companies thinking about using Web services today? They're typically innovative market leaders who want to gain experience with a more loosely coupled architecture that has the promise of quickly delivering measurable cost reductions. More specifically, they tend to be companies that need to integrate multiple par-



### Author Bio

Scott Durchslag is an independent consultant who was formerly a senior member of 12 Entrepreneur's Strategy group. While at 12, Scott focused on supporting Grand Central's communications' customer acquisition efforts and led 12's outreach efforts to key business leaders. Please send feedback on this article to [AGROSS@GRANDCENTRAL.COM](mailto:AGROSS@GRANDCENTRAL.COM)

THE LARGEST INTERNATIONAL

# WEB SERVICES CONFERENCE & EXPO IN THE WORLD!

**WIN A  
\$35,000  
LUXURY CAR!**



ATTENDEES WILL BE INVITED TO TAKE A GOLF SWING TO WIN AND RIDE OFF IN A \$35,000 LUXURY CAR!

## Focus on Web Services

Web Services, the next generation technology that will enable the Internet to work for you and your business, and finally provide that ROI you have been after, will be put under a microscope at Web Services Edge East 2002.

Information-packed sessions, exhibits, and tutorials will examine Web Services from every angle and will provide cutting-edge solutions and a glimpse at current and future implementations. Hear from the innovators and thought leaders in Web Services. Enjoy a highly interactive CEO Keynote panel that will debate and



## WEB SERVICES SKILLS, STRATEGY, AND VISION

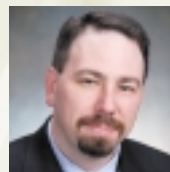
REGISTER ONLINE TODAY

FOR LOWEST CONFERENCE RATES  
EARLY SELL-OUT GUARANTEED!

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

### A Sampling of Web Services-Focused Sessions

- PRACTICAL EXPERIENCES WITH WEB SERVICES AND J2EE
- STATE OF THE WEB SERVICES INDUSTRY
- THE ODD COUPLE: MAKING .NET AND J2EE WORK TOGETHER
- EXPLORING THE .NET MY SERVICES INITIATIVE
- STANDARDS WATCH
- GUARDING THE CASTLE: SECURITY & WEB SERVICES



SEAN RHODY  
CONFERENCE TECH CHAIR  
WEB SERVICES TRACK CHAIR  
EDITOR-IN-CHIEF  
WEB SERVICES JOURNAL

### Featuring...

- UNMATCHED KEYNOTES AND FACULTY
- THE LARGEST INDEPENDENT JAVA, WEB SERVICES, AND XML EXPOS
- AN UNPARALLELED OPPORTUNITY TO NETWORK WITH OVER 5,000 I-TECHNOLOGY PROFESSIONALS

### Who Should Attend...

- DEVELOPERS, PROGRAMMERS, ENGINEERS
- I-TECHNOLOGY PROFESSIONALS
- SENIOR BUSINESS MANAGEMENT
- SENIOR IT/IS MANAGEMENT/C LEVEL EXECUTIVES
- ANALYSTS, CONSULTANTS

Hear these thought leaders in interactive, cutting-edge keynote addresses and panels...



JEAN FRANCOIS ABRAMATIC  
SENIOR VP R&D, FORMER  
CHAIRMAN, W3C • ILOG



DAVE CHAPPELL  
VP CHIEF TECHNOLOGY EVANGELIST  
SONIC SOFTWARE



GREGG KIESSLING  
CEO  
SITRAKA



ANNE THOMAS MANES  
CTO  
SYSTEMET



BARRY MORRIS  
CEO  
IONA



ERIC RUDDER  
SENIOR VP DEVELOPER  
AND PLATFORM EVANGELISM  
MICROSOFT



PATRICIA SEYBOLD  
FOUNDER & CEO  
SEYBOLD

#### For Exhibit information

CONTACT: MICHAEL PESICK  
135 CHESTNUT RIDGE RD.  
MONTVALE, NJ 07645  
201 802-3057  
[MICHAEL@SYS-CON.COM](mailto:MICHAEL@SYS-CON.COM)

web services **EDGE**  
conference & expo

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

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

#### SPONSORED BY:



#### MEDIA SPONSORS



OWNED AND PRODUCED BY



JAVA AND JAVA-BASED MARKS ARE TRADEMARKS OR REGISTERED TRADEMARKS OF SUN MICROSYSTEMS, INC., IN THE UNITED STATES AND OTHER COUNTRIES. SYS-CON PUBLICATIONS, INC., IS INDEPENDENT OF SUN MICROSYSTEMS, INC. ALL BRAND AND PRODUCT NAMES USED ON THESE PAGES ARE TRADE NAMES, SERVICE MARKS, OR TRADEMARKS OF THEIR RESPECTIVE COMPANIES.





tners or customers who are on heterogeneous technology platforms.

In banking, this means companies such as Merrill Lynch, Charles Schwab, Fidelity Investments, Thomas Weisel Partners, Robertson Stephens, ABN Amro, Thompson Financial, and Wachovia. In insurance, it's companies such as Blue Cross/ Blue Shield, Mega Life & Health, Storebrand, and several smaller companies with strong positions in local markets. In travel, early adopters include companies like Dollar and Galileo. In manufacturing, it's companies such as Dell, Ford, GM, Osram Sylvania, VendQuest, and Eastman Chemical.

Why are these the verticals using or assessing Web services to execute real business processes with partners and customers? In short, it's because these industries are relatively data-intensive, and because the need to cost effectively meet their customer needs requires them to collaborate with multiple parties. For example, banking consumes expensive external data feeds and produces research reports and analyses that must be distributed to a wide variety of clients with very specific individual needs. Similarly, as any traveling professional knows who rents cars or checks into hotels, travel requires detailed customer profiles and inventory data. There are large economic benefits – to customer and company alike – to be gained from being able to fill in data only once, then have it move across service providers, with automatic notification of cancelled or delayed arrivals so that car/hotel inventory can be optimized.

## Common Applications of Web Services

A complete explanation of why these verticals are early adopters requires understanding the common use cases, with examples of how actual companies are using Web services

Based on an analysis of companies applying Web services to business processes today, there seem to be four major application areas:

1. Demand Chain Applications
2. Supply Chain Applications
3. Private Marketplaces
4. Application or Web Service Provider Integrations.

### *Demand Chain Applications*

There are several demand-side applications being implemented today. First, enterprises that want to grow profits by distributing their applications or data to partners and/or customers. These applications of Web services are rapidly penetrating the travel and financial services verticals. Second, companies seeking to grow profits by integrating components of their demand chain itself with distributors, resellers, and/or customers. This is penetrating the financial services and manufacturing verticals.

### *Supply Chain Applications*

There are two potential supply side applications. First, enterprises wanting to integrate more cost effectively with direct materials suppliers to confirm orders, pricing, and ship dates. Second, companies wanting to integrate more cost effectively with suppliers of indirect materials. Web services are beginning to get some traction in the manufacturing vertical.

### *Private Marketplace/ Collaboration Hub Integrations*

Marketplaces and their next generation, "collaboration hubs," use Web services as an infrastructure to enable many-to-many integration. This provides a simple, quick, low-cost way to connect new businesses to the hub using open standards, facilitates the orchestration of business processes between collaborating partners, delivers the right information at the right time to the partner or customer, and enables the management and maintenance of connections with all involved parties. This is rapidly penetrating the insurance and manufacturing verticals.

### *ASP/WSP Integrations*

Companies may seek to reduce costs by transporting select data from or between application service providers (ASPs) or Web service providers (WSPs). This information can be used to trigger events or to trigger other information feeds based on the information in a specific field or fields. This is being used in financial services.

## Some Examples of Real Web Services Implementations

Most importantly, there are large companies assessing and already implementing Web services in each of these application areas.

These applications of Web services are not the same across all the early adopting vertical markets. I'll talk later about which applications are being used in what verticals, and also indicates where potential applications may arise as adoption deepens over time.

### *Demand Chain Applications*

Demand chain applications of Web services are an important area of early adoption. In banking, both investment banks and data service providers are using Web services to integrate with each other and with customers. For example, both Robertson Stephens and Wachovia are using Web services to consume Thompson Financial data. They then add value to it through their own research analysts, and use Web services to distribute the specific research reports wanted by each customer. Wachovia is also using Web services to download the Thompson Financial data on consensus estimates into Excel, and then monitor how their research recommendations move the consensus estimate by comparing the average before and after their report is issued. Fidelity is using Web services to eliminate the complex web of protocols and transformations required to give customers an integrated view of their accounts across different products. This reduces the demands on their IT organization while lowering the total cost of ownership.

Insurance is another important early-adopting vertical. A leading national life insurance company is using Web services to integrate with customers so the human resources department can directly update any changes in employees' status. This increases value to customers while reducing the costs of investigating claims that were based on obsolete data. Storebrand, Norway's largest insurance company, has been using Web services to replace the manual process by which it was calculating the potential benefits for 390,000 employees at 6,500 different customers under a variety of insurance offerings. They now have an automated process that extracts information directly from the customer's payroll system and transmits it via Web services to Storebrand's mainframe, where the scenarios are run for each customer.

In travel, Dollar Rent-a-Car wants to expose their reservations process on as many travel Web sites as possible. Using Web services, they're able to distribute the code required to link in with their proprietary systems through an architecture that avoids taking eyeballs away from their partners' sites. Galileo is ex-

perimenting with Web services to offer travel services and to integrate with their travel suppliers. (Galileo connects 42,000 travel agency locations to 511 airlines, 37 car rental companies, 350 tour operators, and 47,000 hotels.) Another large travel hub has a hotel reservation application that they want to embed on as many individual travel sites as possible to enable customers to make their hotel reservations through a Web service that integrates directly with hotels in their network.

In manufacturing, Eastman Chemical is using Web services to publish a catalog tailored to the needs of each customer, with pricing reflecting their corporate rate, and offering the ability to execute transactions. Large automobile manufacturer is using Web services to integrate ordering and inventory management with their global dealer network. They found it impossible to convince all of their dealers to adopt the same EAI standards, and Web services enables them to quickly and cheaply connect using a PC and the Internet. The economic benefit of reducing inventory creates hundreds of millions in cost reductions. Revenue growth is also anticipated to enable dealers to access each other's inventories so order lead times can be cut...and more customers will maybe buy cars as a result.

#### **Supply Chain Integrations**

While there are less diverse use cases for Web services projects in supply chain initiatives, there are proven and valuable applications. As companies such as Ariba and CommerceOne have shown, this area represents an important part of XML adoption. Within Web services applications, there's traction in integrating smaller suppliers of direct materials. For example, a large paper company is using Web services to integrate with suppliers they'd previously fax orders to and confirm delivery dates or pricing with by telephone. This creates real cost savings and has a clear enough value proposition to indicate that adoption is merely a matter of time.

Similarly, technology manufacturers are

looking at using Web services to tighten the smaller tiers of their supply chains. These smaller suppliers had resisted integration because they feared having their margins squeezed by greater buyer power, and they shied away from adopting proprietary software standards. Web services enables them to gain the benefits of integration without being locked into their customer's systems.

#### **Private Marketplace/ Collaboration Hub Integrations**

Companies often run private marketplaces or collaboration hubs, a next-generation marketplace, to bring together partners to help execute a business process. An insurance service provider has a collaboration hub that aggregates all the sub/processes required to execute an insurance claim. They're using Web services to bring together the assessor, bank, repair shop, or health care provider. Claims can be easily tracked and unusual charges automatically flagged. In manufacturing, a large provider of agricultural equipment uses Web services as the infrastructure to bring together their dealers, equipment loan providers, and other agricultural service providers. GM is doing something similar in automobiles as an inexpensive platform to integrate with their global dealer network.

#### **ASP/WSP Integrations**

Finally, Web services are getting some traction as a low-cost integration platform for application and Web services. Putnam Lovell is an investment bank that's using Web services to integrate Salesforce (CRM) and Blue Matrix (research report distribution) with their customers. Putnam keeps their customer profiles in Salesforce.com, and transports the data on the ticker symbols of companies the customer wishes to track. Using Web services, this data is delivered to Blue Matrix, which automatically sends the relevant research reports on the companies of interest directly

to the customer. All of this occurs outside the firewall, yet it appears to the customer that everything is done by Putnam.

#### **Conclusions**

Innovative large enterprises in select verticals are already using Web services today to create and deliver value to their customers. They're gaining valuable experience in implementing a Web services architecture that reduces costs, increases revenues, and enables greater agility in meeting dynamic customer needs. More importantly, they're learning how their business architecture must change to fully reap the rewards of strategic agility. They're developing loosely coupled business processes that enable them and their partners to be more focused on what each does best in creating value for the customer.

Loosely coupled business processes involve taking a modular approach to designing and managing processes with standard interfaces that allow dynamic swapping of process components to tailor the processes to the needs of individual customers. This is the business analog to the technology architecture shift toward Web services, which involves moving away from hardwired software to modular components based on open standards such as SOAP, WSDL, and UDDI.

Indeed, the full potential of Web services will remain untapped until collaborating companies make the more challenging transition to this new business architecture. This shift will position early adopting companies to develop a "process network" through which they orchestrate a more flexible, tailored value chain that could build a long-term leveraged growth platform. Process networks are an expanding group of select service providers, with one company acting as an orchestrator of all the activities required to deliver value to the customer. They will be ahead of their competitors in creating a virtuous cycle based on recruiting the best partners, developing incentives that ensure each more specialized company is aligned with the customer, and creating more granular feedback loops on performance that generate more and more value over time.

Experience in other industries proves that early movers who create these types of networks capture increasing returns that provide extraordinary profitability and dominant market share. Web services make this possible in a much broader array of industries, and there are bold companies moving in today's difficult environment to gain the experience necessary to win this new game. ©

**// The full potential of Web services  
will remain untapped until collaborating companies  
make the more challenging transition  
to this new business architecture"**



# The Sun Rises on Web Services

## An overview of Sun's 'Java XML Pack': five new APIs and architectures

**W**ith Microsoft's .NET marketing campaign in full swing, Sun announced its Sun ONE (Open Net Environment) initiative in October of 2001. While J2EE provides a robust, scalable, and portable enterprise platform, until then it had been sorely lacking in the area of standardized support for Web services. As a part of the Sun ONE initiative, Sun has since released the Java XML Pack, a suite of Java APIs for working with XML.

### Java XML Pack Basics

So what exactly is the Java XML Pack? From a Web services perspective, the Java XML Pack facilitates the development, publishing, locating, and invocation of XML services via the Java 2 Platform.

The Java XML Pack provides a truly extensible and flexible interface for deploying enterprise-services architectures and is designed to provide a core set of standardized Java APIs for building, deploying, and accessing XML applications – and, specifically, dynamic XML-enabled Web services. The Java XML Pack APIs constitute a standard set of abstract APIs that aren't tied to a particular implementation technology. Thus, as is true with the rest of the Java 2 Platform, developers can code their application to an API, and easily adapt to changes in the industry without the need to rework or recompile their application code. Developers are free to use any JAXP-compliant XML parser, JAXB-compliant



schema mechanism, JAXM-compliant messaging provider, JAXR-compliant registry service, or JAX-RPC-compliant invocation mechanism.

Additionally, although SOAP, WSDL, and UDDI are the current technologies du jour, the industry landscape may very well shift to other technologies such as XP (the W3C spec) and ebXML. When this happens, a Java XML Pack-compliant implementation can be plugged into an existing Java XML Pack-enabled architecture, and your applications will instantly leverage the new technologies to accomplish the same business services as before. Let's see your .NET technology do that!

### Architecture Overview

The JAX Pack technologies can be divided into two broad categories: those that deal with an XML document holistically, and those that deal with it from a procedural perspective.

#### Document-centric

- Java API for XML Processing (JAXP) —

processes XML documents using a JAXP-compliant XML parser

- Java Architecture for XML Binding (JAXB) — provides a mapping between XML document elements and Java classes, allowing XML documents to be treated as native Java objects

#### Procedure-centric

- Java API for XML Messaging (JAXM) — facilitates XML-based messaging via the Java programming language
- Java API for XML Registries (JAXR) — offers a uniform mechanism for accessing business registries through Java
- Java API for XML-based RPC (JAX-RPC) — facilitates XML-based RPC over the Internet, allowing XML-formatted parameters to be passed to remote services, and XML-formatted values returned

This combination of APIs capable of dealing with an XML document as a whole, and providing support for invoking procedures with XML, constitutes a powerful services framework for the Java platform.

### Drilling Down

Now let's look at each of these five APIs in a little more detail:

#### 1. JAXP – Processing XML

The Java API for XML Processing (JAXP) predates the rest of the Java XML Pack. It provides basic XML capabilities for the Java platform through a flexible, standards-based API. JAXP is namespace aware and provides support for parsing XML documents, as well as transforming them into other formats. From an architectural perspective, JAXP is designed from the ground up to provide optimum flexibility. It employs a pluggability layer allowing compliant XML parsers and XML processors to be substituted at runtime, as well as supporting multiple parsing paradigms.

When parsing an XML document, developers can use the JAXP API in conjunction with any JAXP-compliant parser to consume XML documents and extract the data inside them. Both



#### Author Bio

Kyle Gabhart, director of the Java division of Objective Solutions ([www.objectsoln.com](http://www.objectsoln.com)), is a prolific writer with more than a dozen technical articles and books to his name. Objective Solutions is a leading provider of Java and Web services mentoring and training. [JAVA@OBJECTSOLN.COM](mailto:JAVA@OBJECTSOLN.COM)

THE LARGEST INDEPENDENT

# JAVA

## DEVELOPER CONFERENCE IN THE WORLD!

**WIN A  
\$35,000  
LUXURY CAR!**



ATTENDEES WILL BE INVITED TO TAKE A GOLF SWING  
TO WIN AND RIDE OFF IN A \$35,000 LUXURY CAR!

### Focus on Java

Java, now mainstream, is the dominant back-end technology upon which next-generation technologies are evolving.

Hear from the leading minds in Java how this essential technology offers robust solutions to technology professionals and senior IT/IS strategy decision makers.

The Java Fast Paths on June 24, a Java-focused CEO Keynote Panel, and comprehensive conference sessions will focus you on Java every information-packed day!



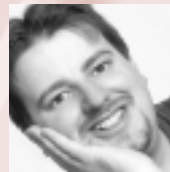
## JAVA IN JUNE ESPECIALLY IN NEW YORK REGISTER ONLINE TODAY

FOR LOWEST CONFERENCE RATES  
EARLY SELL-OUT GUARANTEED!

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

### A Sampling of Java-Focused Sessions

- JAVA 1.4: WHAT'S NEW?
- BUILDING TRULY PORTABLE J2EE APPLICATIONS
- JAVA TOOLS FOR EXTREME PROGRAMMING
- BUILDING ASYNCHRONOUS APPLICATIONS USING JAVA MESSAGING
- .NET VS. J2EE
- J2EE: SETTING UP THE DEVELOPMENT ENVIRONMENT
- BUILDING WEB SERVICES WITH J2EE
- DETECTING, DIAGNOSING, AND OVERCOMING THE FIVE MOST COMMON J2EE APPLICATION PERFORMANCE OBSTACLES



ALAN WILLIAMSON  
JAVA CHAIR • EDITOR-IN-CHIEF  
JAVA DEVELOPER'S JOURNAL

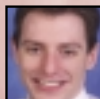
### Featuring...

- UNMATCHED KEYNOTES AND FACULTY
- THE LARGEST INDEPENDENT JAVA, WEB SERVICES, AND XML EXPOS
- AN UNPARALLELED OPPORTUNITY TO NETWORK WITH OVER 5,000 I-TECHNOLOGY PROFESSIONALS

### Who Should Attend...

- DEVELOPERS, PROGRAMMERS, ENGINEERS
- I-TECHNOLOGY PROFESSIONALS
- SENIOR BUSINESS MANAGEMENT
- SENIOR IT/IS MANAGEMENT/C LEVEL EXECUTIVES
- ANALYSTS, CONSULTANTS

Hear these thought leaders in interactive, cutting-edge keynote addresses and panels...



TYLER JEWELL  
PRINCIPAL TECH.  
EVANGELIST • BEA



GREGG KIESSLING  
CEO  
SITAKA



DAVID LITWACK  
CEO  
SILVERSTREAM



BARRY MORRIS  
CEO  
IONA



GREGG O'CONNOR  
PRESIDENT  
SONIC SOFTWARE



RICK ROSS  
FOUNDER  
JAVA LOBBY



JAMES DUNCAN  
DAVIDSON  
FATHER OF ANT

#### For Exhibit information

CONTACT: MICHAEL PESICK  
135 CHESTNUT RIDGE RD.  
MONTVALE, NJ 07645  
201 802-3057  
[MICHAEL@SYS-CON.COM](mailto:MICHAEL@SYS-CON.COM)

**JDJEDGE**  
conference & expo

**JUNE 24-27**

JACOB JAVITS  
CONVENTION CENTER  
NEW YORK, NY

**OCTOBER 1-3**

SAN JOSE  
CONVENTION CENTER  
SAN JOSE, CA

#### SPONSORED BY:




#### MEDIA SPONSORS



#### OWNED AND PRODUCED BY



JAVA AND JAVA-BASED MARKS ARE TRADEMARKS OR REGISTERED TRADEMARKS OF SUN MICROSYSTEMS, INC., IN THE UNITED STATES AND OTHER COUNTRIES. SYS-CON PUBLICATIONS, INC., IS INDEPENDENT OF SUN MICROSYSTEMS, INC. ALL BRAND AND PRODUCT NAMES USED ON THESE PAGES ARE TRADE NAMES, SERVICE MARKS, OR TRADEMARKS OF THEIR RESPECTIVE COMPANIES.



## JAXB uses a custom parsing mechanism that builds an in-memory object representation like DOM, but is about as fast as SAX parsing //

SAX-based (event model) parsing, as well as DOM-based (tree model) parsing are supported. This allows developers to utilize whichever parsing paradigm best suits a particular application's requirements. The Simple API for XML (SAX) defines an event-based API that navigates element by element through an XML document, signaling events as particular XML constructs are identified. The Document Object Model (DOM) defines a set of interfaces for constructing an object representation of the parsed XML document, based on a tree data structure. With DOM, the entire document is parsed and the tree structure is stored in memory, allowing repeated, random access to the XML document data. With SAX, the XML data is only accessible when the parser reaches a given element; once it moves on to another element, knowledge of the previous element is lost. These differing parsing styles allow developers to select whichever method best meets the application's requirements. Should the application need to evolve and support a different style, such a change is streamlined via JAXP.

When transforming an XML document, developers can use the JAXP API in conjunction with any JAXP-compliant processor to consume the XML document and transform the data into the desired format (HTML, WML, XSL, or some other flavor of XML). Currently, the only transformation technology supported by JAXP is the extensible Stylesheet Language Transformations (XSLT). XSLT allows the definition

of XSL stylesheets capable of transforming an XML document into any other textual format, markup language, or even foreign language, based upon template definitions.

The JAXP 1.2 reference implementation ships with two open-source components. The Crimson XML parser (which supports SAX and DOM) and the Xalan XSLT processor, both jointly developed by Sun and the Apache Software Foundation.

### 2. JAXB – XML Object Binding

The Java Architecture for XML Binding (JAXB) defines an XML-to-Java mapping architecture. It's a fast, efficient means of creating a two-way mapping between the elements in an XML document and Java classes. Given a DTD that defines the schema for the XML document, and a binding schema that defines how to bind the schema to classes, the JAXB compiler will generate a set of Java classes containing all the necessary code to parse XML documents based on the schema. These classes can be used to build a Java object tree representing an XML document that's valid for the schema. This object tree can be constructed by parsing an XML document, or by directly instantiating objects from the generated classes. JAXB can also be used to dynamically modify the element contents of the object tree, and it can marshal the object representation into an XML document based on the mapping defined within the binding schema. In this way, JAXB defines an architecture that provides a convenient, two-way mapping between Java and XML.

At first glance, it would appear that with the JAXB architecture there's no need for the JAXP API. Although there are some areas in which they overlap, they each have distinct features that make them more or less useful for certain tasks.

Table 1 summarizes the differences between the two technologies. For starters, JAXP provides a one-stop shop for parsing, processing, and transforming XML documents. You can select SAX for high-performance, one-time processing, or when you only want a portion of a very large document. You can select DOM when you need to maintain an in-memory representation of an XML document. The DOM API will even allow you to modify the structure and content of an

object tree. JAXP also provides a transformer API to convert a document from one format to another. From a validation perspective, JAXP can use a previously known schema, or accept a new one dynamically at runtime. This validation paradigm, although a bit costly, is incredibly flexible. JAXP provides all these features from the convenience of a single API.

JAXB provides some similar features and some unique features. JAXB uses a custom parsing mechanism that builds an in-memory object representation like DOM, but is about as fast as SAX parsing. This is because the generated classes contain all the necessary DTD logic, whereas SAX parsers must accommodate a dynamic validation check against a schema. Additionally, the in-memory object tree created and maintained by a JAXB application has a considerably smaller footprint, because, unlike JAXP, it doesn't contain tree manipulation logic. Developers can certainly modify the object properties (which correspond to element values in XML), but JAXB doesn't facilitate the manipulation of the structure of the object tree (adding and removing elements). One additional aspect of JAXB is that a schema is required and only valid XML will be processed by the application. Whether this is a restriction or a feature depends entirely on your application's requirements.

JAXB provides a bridge between the language-neutral world of XML and the platform-neutral world of Java. In the same way that an XML document is an instance of a schema, a Java object is an instance of a class. As Sun's whitepaper, *Web Services Made Easier* explains it: "Thus, JAXB allows you to create Java objects at the same conceptual level as the XML data."

TABLE 1: A comparison of JAXP and JAXB

JAXP	JAXB
<ul style="list-style-type: none"><li>• Accesses data serially (via SAX), or on demand (via DOM)</li><li>• Parses nonvalidated documents</li><li>• Applies XSLT transforms</li><li>• Inserts or removes objects from an object tree that represents XML data</li><li>• Uses same processing code with documents based on different DTDs, or dynamically accepts a DTD</li></ul>	<ul style="list-style-type: none"><li>• Accesses data in memory, but no need for tree manipulation</li><li>• Processes only valid data</li><li>• Generates Java classes based on a DTD</li><li>• Builds object representations of data and marshals that data to a new XML document</li></ul>



Java developers are comfortable with object-based relationships, encapsulation, and the level of granularity that objects provide. JAXB provides a fast, convenient mechanism to access XML in this same manner.

### 3. JAXM – XML Messaging

The Java API for XML Messaging (JAXM), is a full-featured asynchronous messaging technology, representing a standard way to send messages across a network from the Java platform. It's decoupled from the underlying protocol(s) being used, affording developers a standardized messaging API that won't have to be changed to accommodate a different underlying protocol. JAXM is currently based on the SOAP 1.1 and SOAP with Attachments specifications, but it can be extended to work with higher level protocols such as ebXML or bizTalk, or any other protocol that the industry – or a particular business partner – chooses to embrace at a later time.

JAXM messaging requires the use of a messaging provider service, which handles the transportation, routing, marshalling and unmarshalling work that goes on under the hood. The JAXM API shields developers from such details, and provides a clean and simple interface for sending messages, receiving messages, or simply taking part in the processing of a portion of a message (serving as a message intermediate). The messaging provider can offer additional functionality, such as guaranteed messaging, event logging and auditing, message threading, and even security. All such activity is shielded from the sender, receiver, or any participant in a messaging exchange. All that's required is that the JAXM client make method calls against the API. This decoupling allows messaging providers to offer value-added features and allows application developers the freedom to select the provider that best meets their needs. It may be helpful to think of JAXM in the messaging space as being equivalent to what JDBC provides in the database access space. In this analogy, a messaging provider serves a similar purpose as a JDBC driver. Developers write their application code to the API, and vendors develop software components that provide concrete implementations of the API. The primary difference here is that, while JDBC requires that you use a driver that is compatible with the database being used, the messages sent from a JAXM messaging provider conform to industry standards, and can be consumed by any recipient that under-

stands those standards. The JAXM reference implementation uses SOAP 1.1, thus any SOAP-enabled recipient would be able to unmarshal a message sent from it

JAXM is a robust messaging technology supporting a wide-variety of messaging paradigms, including point-to-point messaging, publish-subscribe messaging, intermediate point-to-point, request-response, and even a process-workflow messaging model. (That's not a comprehensive list by any means.) Which specific messaging paradigms are available is entirely dependent on the messaging provider used, but JAXM provides a standard means of defining how a message should be sent and what auxiliary information may need to be passed in order to process the message properly.

In many ways, JAXM is very similar to the Java Messaging Service (JMS). They both provide an abstract messaging API that's decoupled from the back end implementation, and they both require a messaging provider to implement the API. In fact, the current JMS-compliant messaging providers are likely to be the first vendors to incorporate support for JAXM into their tools. The difference, then, between JAXM and JMS, is that JMS ultimately produces a Java-centric message. Whereas JAXM messages comply with industry-standard formats such as SOAP and ebXML.

### 4. JAXR – Registering/Locating XML Services

The Java API for XML Registries (JAXR) provides a registry independent API for accessing XML-based registries. A registry is very much like the yellow pages, or the listings maintained by a search engine. It provides a convenient means to browse available services, gather information regarding those services and then select the desired service. Like search engines, registries support the ability to return a list of search results based upon specified criteria. Like the yellow pages, registries categorize their contents and allow clients to browse by category. In addition to providing the ability to search such registries and retrieve details about available XML services, JAXR enables developers to publish new services to the registry and even to manage the registry data itself (assuming that proper authentication is provided first).

Although the majority of the literature talking about service registries discuss as the concept of global public registries like Xmethods.com, UDDI.org, and the registries

of IBM, Microsoft, and HP a far more immediate use for service registries can be found in intranet and extranet registries. It's unlikely, any time in the near future, that consumers will query public registries trying to find the best deal on their auto insurance, or that businesses will start browsing public registries looking for a service to fill a mission critical need. A much more likely scenario is that businesses may set up private registries to serve as a single point of reference for their various enterprise services. Rather than an application developer combing through outdated documentation, or waiting for a response to a request they sent to management regarding a particular service they need to access, they can simply browse the company's internal registry looking for the service. If they can't find one, the developer can be reasonably sure that one doesn't exist. This tremendously increases the productivity of that developer and of everyone else that might have occasion to locate and/or access the available enterprise services. A similar need can be met with registries in a B2B extranet scenario. A business can expose a private registry to its business partners, providing them with a comprehensive, standards-based mechanism for searching, locating, inquiring about, and binding to whatever services the business has made available. The business could even allow the partner to register new services that would now become available for the business to access.

There are currently two primary specifications for XML registries:



It might be  
helpful to think of  
JAXM in the messaging  
space as being  
equivalent to what JDBC  
provides in the database  
access space //

SAVE 30% off the annual newsstand rate

# JAVA DEVELOPER'S JOURNAL

Offer subject to change without notice

## ANNUAL NEWSSTAND RATE

~~\$71.88~~

YOU PAY

**\$49.99**

YOU SAVE

**30%**

Off the  
Newsstand Rate

## DON'T MISS AN ISSUE!

Receive 12 issues of *Java Developer's Journal* for only \$49.99! That's a savings of \$21.89 off the cover price. Sign up online at [www.sys-con.com](http://www.sys-con.com) or call 1 800 513-7111 and subscribe today!

## In March *JDJ*:

**Extending the J2SE 1.4 Logging Package**  
Monitoring made easy

**Jini Surrogate as a Platform for J2ME Games**  
Surrogate architecture incorporates smaller devices

**Integrating J2ME, GPS, and the Wireless Web**  
Developing location-based applications

**Book Review:**  
*Java Internationalization*

**Java and Wireless**  
Building end-to-end Palm applications using Java

**Mac OS X & Java**  
A perfect marriage



- The ebXML Registry and Repository standard being jointly developed by OASIS and the UN
- The Universal Description, Discovery, and Integration (UDDI) specification being developed by a consortium of over 100 companies.

As is the case with the other Java XML Pack technologies, JAXR represents a flexible mechanism that's not tied to any particular implementation. Thus the same application code can be used for accessing first a UDDI registry, then an ebXML registry. The JAXR 1.0 Early Access Reference Implementation supports both the ebXML registry and the UDDI Registry v2.0 specification. Sun intends to incorporate support for ebXML as that standard is finalized.

### 5. JAX-RPC – XML-based RPC

The Java API for XML-based RPC (JAX-RPC) gives Java developers a simple, convenient mechanism for invoking remote procedures in a neutral, standards-based way. JAX-RPC is a synchronous, pure Java, RPC-invocation mechanism that leverages standard XML protocols such as SOAP. It's pure Java in that it doesn't require the existence of an XML schema or a messaging provider in order to make a remote call.

JAX-RPC provides a simple, object-oriented mechanism for specifying the service location, parameter types and values for a service. As with other RPC mechanisms, a JAX-RPC client simply creates a local object, propagates it with the necessary information, then invokes methods directly on the local object. When the call is made, JAX-RPC marshals the call across the network (as a SOAP message), and unmarshals the response (converts the SOAP response to a Java object). The details of the mapping and reverse mapping of data types and other XML element constructs are abstracted away from the JAX-RPC application developer. Thus, it provides a simple, straightforward mechanism for accessing RPC-style Web services from the Java platform.

As with any RPC mechanism, an interface for the remote procedure needs to be

generated, along with stubs and ties. The JAX-RPC reference implementation accordingly ships with a stub and tie generation tool, xrpcc. Given an appropriate interface, the xrpcc tool generates stubs, ties, and a server configuration file. The tool supports both standard Java RMI interfaces and WSDL documents. In addition to generating the appropriate files to set up the RPC-runtime, the tool can also convert a set of Java RMI interfaces into a WSDL document and vice versa.

Like JAXM, JAX-RPC is a tremendous client technology for accessing Web services. Not only does it leverage Java's platform independency, allowing the client to be run on any Java 2 platform, but the access mechanism used by JAX-RPC is SOAP, an inherently language and platform neutral communication protocol. This provides the optimum in RPC-flexibility and portability.

## Conclusion

The Java APIs and architectures for XML provide a comprehensive suite of XML APIs for the Java platform. JAXP allows developers a flexible means to parse, process and transform XML documents; JAXB defines a two-way mapping architecture between XML and Java; JAXM is a robust asynchronous messaging technology; JAXR provides standardized access to XML registries, and JAX-RPC enables RPC-style invocation of XML Web services.

Just as JDBC has become an expected acronym to see on a Java developer's resumé, the JAX technologies are sure to become a required part of any Web or enterprise developer's toolkit.

## Exploring Further

You can download the pack from Sun's Web site at <http://java.sun.com/xml/java/xmlpack.html> and take it for a spin. Sun recently released the Java XML Pack as a part of the Java Web Services Developer Pack (Java WSDP) which is available for download at <http://java.sun.com/webservices/websservicespack.html>

A tutorial for the WSDP is available at <http://java.sun.com/webservices/docs/ea1/tutorial/index.html> ©



Now in More than 5,000 bookstores worldwide

**FORFAST**  
DELIVERY

subscribe **Now!**

Go  
Online  
and  
Subscribe  
Today!

The World's Leading  
Independent WebLogic  
Developer Resource

FOR WLS DEVELOPERS BY WLS DEVELOPERS  
**WebLogic**  
DEVELOPER'S JOURNAL

Helping  
you enable  
inter-company  
collaboration  
on a global scale

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

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

**WebLogic**

**Journal.com**

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

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

**SYS-CON**  
MEDIA

**EnginData Presents**

# 2002 Developer Market Survey Reports



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

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

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

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

**engindata** ✓  
RESEARCH



Reviewed by Brian Barbash

## About the Author:

Brian R. Barbash is a consultant for the Consulting Group of Computer Sciences Corporation. He specializes in application architecture and development, business and technical analysis, and Web design.

BBARBASH@CSC.COM



## eXtend Composer by SilverStream

*An intuitive, organized work environment that's an effective tool for developing Web services*

SilverStream eXtend Composer is part of the SilverStream eXtend family that provides a visual development environment used to create service-oriented applications. Composer allows developers to create services that gather data from multiple disparate sources, apply appropriate business logic, and return the results. The finished components meet J2EE standards and will execute on compliant application servers. Composer explicitly supports SilverStream's eXtend Application Server, IBM's WebSphere, and BEA's WebLogic.

Composer itself consists of three components: eXtend Composer, eXtend Composer Enterprise Server, and eXtend Connect Family. eXtend Composer, the IDE for creating Web services, is the main focus of this review. The Enterprise Server, executing in the application server, provides the runtime environment for Web services created with Composer. The eXtend Connect Family provides connectivity to many types of enterprise data sources and applications, including EDI, CICS, Telnet, JDBC, 3270/5250 terminals, and JMS.

### Tool Overview

Composer applications are objects whose inputs and outputs are XML documents built with basic predefined components. Since XML is the centerpiece of all operations, XML files can quickly proliferate and become difficult to manage. To remedy this, Composer provides a very nice process for managing and organizing the files. In Composer, XML objects, referred to as templates, ultimately define a component's inputs and outputs.

Templates may be grouped into categories that relate to high-level doc-

ument types such as sales orders and inventory requests. Each template is assigned at least one sample XML file that's used to define the data structure for a component's input and output. Composer allows XML templates to be validated at runtime to conform to a DTD or schema. Finally, stylesheets may be added to the templates to automatically transform the XML into a more appropriate format for the developer's needs.

Composer resources are objects that are shared across multiple components



within an application. The basic resources available are code tables and maps, connections, XML Schemas, custom scripts, terminal stylesheets, and WSDL. Additional resources may be available for specific eXtend Connect objects depending on that component's requirements.

The building blocks of all Composer applications are the components, also known as the eXtend Connect family. Components provide the connectivity to enterprise systems to execute business

**SilverStream****SILVERSTREAM SOFTWARE**

Two Federal Street  
Billerica, MA 01821  
Phone: 1 888 823-9700  
Web: [www.silverstream.com](http://www.silverstream.com)  
E-mail: [info@silverstream.com](mailto:info@silverstream.com)

**TEST ENVIRONMENT**

Windows 2000 Professional,  
Pentium III, 256MB RAM



logic and retrieve data. When building a component, XML inputs and outputs are established from library of previously defined templates, and actions are defined to execute the business logic for a given component.

Finally, Composer services are the objects that define the business process to be executed by a collection of components. Service objects are pieces of code (defined and deployed on an application server) that define the external interface to an application.

### Development

The development cycle in Composer is an easy-to-follow process that's consistent throughout the application. Figure 1 is a screen shot of the IDE. The upper-left panel displays the available services, components, resources, and XML templates. The lower left panel lists the available objects for the currently selected item. The right-hand portion of the screen is the edit area for an eXtend object. In this example, the upper panels define the inputs and outputs, while the lower section displays the list of actions a given component will execute. This section will vary depending on the require-

ments and functionality of the selected item. Finally, the lower portion of the screen provides logging and runtime information.

For this review, I'll develop a simple Web service that extracts information from a personal movie and music system. The eXtend Connect family provides access to this information in a variety of ways, including direct database access through JDBC, execution of a Telnet application, interaction with a JMS system, and interaction with a Web-based front end.

The definitions for the XML interfaces must be in place before developing of any service in Composer. In this case, I've defined request and response documents for retrieving the details of an individual movie. These documents are loaded into XML templates and are available to all future components of this service.

For the first method, I'll use a direct JDBC connection. For JDBC, a connection resource to the host database must be created. Any database that supports JDBC will work; in this example, I've used Oracle 8i. The JDBC resource itself is created by defining the inputs and outputs using the predefined XML templates for a request and a response. The next step involves writing the appropriate SQL statement to retrieve the data. Finally, by dragging the necessary identifying elements from the input XML document directly into the Where clause of the SQL statement, an XPath link is established that causes the component to use the data from the incoming XML document as the value for the Where clause.

It's worth noting that the majority of the manipulation of components, actions, and parameters is done via drag-and-drop. This provides an easily understood and intuitive process for development.

As is the case with most services being developed in Composer, the input and output XML structures are more complex than the results from a raw data source. To solve this problem, Composer provides the capability to add a set of temporary XML files to a component. These may be used as a data-staging area to map data from one structure to another. Additionally, Composer includes an ECMA script engine that provides powerful scripting options when working with data.

Accessing the movie database through a Telnet connection highlights another connectivity option Composer provides. Using Telnet, Composer interacts with the host system based on a series of actions defined through a recording process. In Figure 2, the upper-right portion of the screen is a Telnet

Terminal emulator. As the developer interacts with this window, the actions are recorded in the Action list in the lower right. As with all other types of components, the full complement of drag-and-drop operations and ECMA scripts is available to developers during the Telnet session. Composer then "scrapes" the desired results from the screen and maps them to the output or temporary XML documents.

As noted earlier, several other options exist for working with data sources, each of which provides the same level of flexibility offered by the drag-and-drop, scripting, and action mapping.

## Bringing It Together – Services and Deployment

Once the individual component development has been completed, a Web service must be created to deploy to an application server host. Ideally, the Web service should contain only calls to individual components to execute business logic, although Web services share capabilities for actions and scripting of components.

For deployment, Composer provides three options: first, a wizard can be used to deploy code automatically; second, the server administrator can manually deploy code into the environment; and finally, eXtend Workbench can be used for deployment. This option is recommended for large projects involving several types of components. All deployment methods require the appropriate Composer Server to be installed and running in the target environment. In this case I'll use the Composer wizard to deploy to a WebLogic server.

The deployment process involves several steps. First, the developer is required to specify the target server type, the staging directory, the JAR file in which to package the objects, and the deployment context (package structure – e.g., com.syscon.wsjreview) in the JAR file. The next steps involve specifying the triggering mechanism(s) for the Web service.

Services may be triggered by servlets, EJBs, or SOAP-based HTTP requests. For servlets, the request type, output type, and URL path must be specified. Once deployed, all requests to the specified URL will invoke the Web service. EJB service triggers require a JNDI path and a Trans-

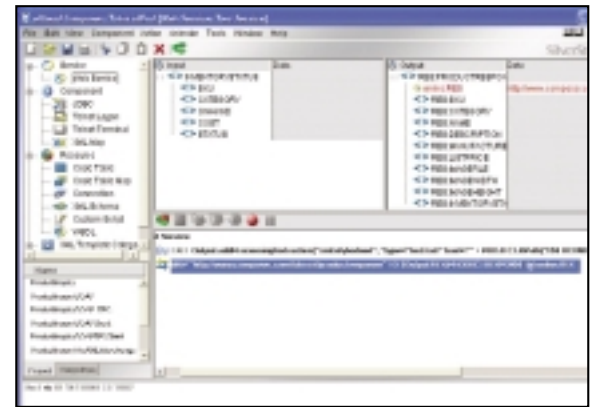


FIGURE 1 | Composer IDE

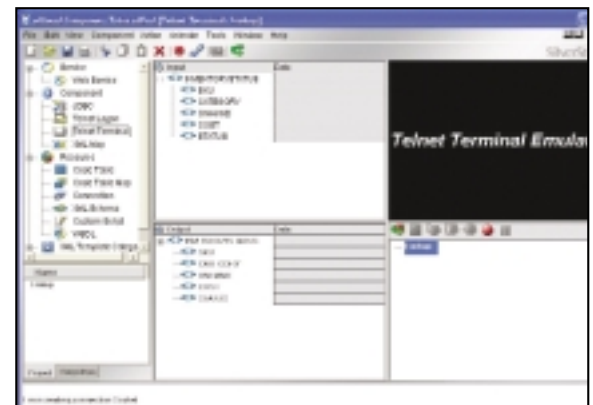


FIGURE 2 | Telnet application

action attribute. The resulting EJB can be called by any Java application to execute the service. To provide an additional interface to the EJB triggers, a second type of servlet trigger can be created to call a previously defined EJB trigger. Finally, a SOAP trigger requires the SOAP binding style, an RPC return namespace (if applicable), and a URL path for execution.

Once all required information is provided, the objects are automatically deployed to the host server and are available for execution.

## Summary

SilverStream's eXtend Composer provides developers with a powerful IDE in which to create service-oriented applications. Through the eXtend Connect Family, Composer enables companies to leverage investments in legacy applications and data while extending them to trusted trading partners. Composer offers an intuitive, organized work environment and is an effective tool for developing Web services. ©



# Getting into the Flow: the Web Services Flow Language

*A first step into potentially  
revolutionary changes*



**T**he goal of the Web services architecture is to enable seamless application integration over the network without regard to programming language or operating environment – one of the key components of the goal to enable inter-and intra-enterprise business processes and workflows to take advantage of the benefits that Web services offer. The Web Services Flow Language (WSFL) is a proposed grammar from IBM that extends the Web services architecture by providing the ability to describe a directed-graph model for defining and processing business processes in terms of the Web services that implement that process, and it defines a public interface that allows business processes to themselves be defined as Web services. This article offers only an initial taste of what WSFL is and what it has to offer; therefore, it may gloss over a few of the more in-depth concepts.

## Business Process Basics

If you're unfamiliar with business process modeling and workflows in general, Figure 1 is an example of a simple workflow process modeled using a directed-edge graph. Each box is an activity (some work that needs to be done); each solid arrowed line (edge) represents the flow of processing control from one activity to another, with the arrow indicating the direction of the flow (thus the term "directed edge"). Decisions are made at various control points to determine whether or not to continue to processing, whether or not the current activity is finished and the graph may continue, whether or not an error has occurred, and so on. Dotted-lines connecting activities indicate the flow of information between activities.

Based on this diagram, we can easily imagine the course of the workflow as processing moves from one activity to the next, and as decisions are made at each control point. WSFL is essentially a tool to model the exact same graph using an XML syntax that can be read by both humans and machines. By consuming WSFL, a workflow engine can walk through the business processes activity by activity, control point by control point. It's not a new concept by any means; however, given the revolutionary power of Web services to bridge cross-platform boundaries, WSFL is enhanced by its ability to model business processes that can easily span not only across business boundaries, but technology boundaries – overcoming a limitation that many workflow engines suffer from.

The workflow diagram illustrated in the

figure is given in Listing 1 using WSFL syntax. Study the syntax and structure of the document well. For reference information about the syntax and the individual elements, see the Web Services Flow Language specification.

## Learning the Lingo

WSFL, like workflow in general, has a language all its own; and, like anything, the key to mastering it is to master the vocabulary. Listed below are just some of the more important concepts within WSFL.

- **Business process:** The business process is any collection of activities that together accomplish some business objective. For example, processing a credit card number, hiring a new employee, and submitting a patent are all examples of business processes.

- **Flow model:** The flow model is the actual XML representation of the directed graph that models the business process. It is the structure used to compose Web services defined by Web Services Description Language (WSDL) documents into workflows. Flow models may be known by several different na-



### Author Bio

James Snell is an architect in IBM's Emerging Technologies group. His primary focus is on the strategic evolution of IBM's Web Services Initiative, specifically as it relates to enterprise development. Prior to joining IBM, James worked for an independent software consulting firm providing custom enterprise development and consulting services for a variety of Fortune 500 and 100 customers. JASNELL@US.IBM.COM

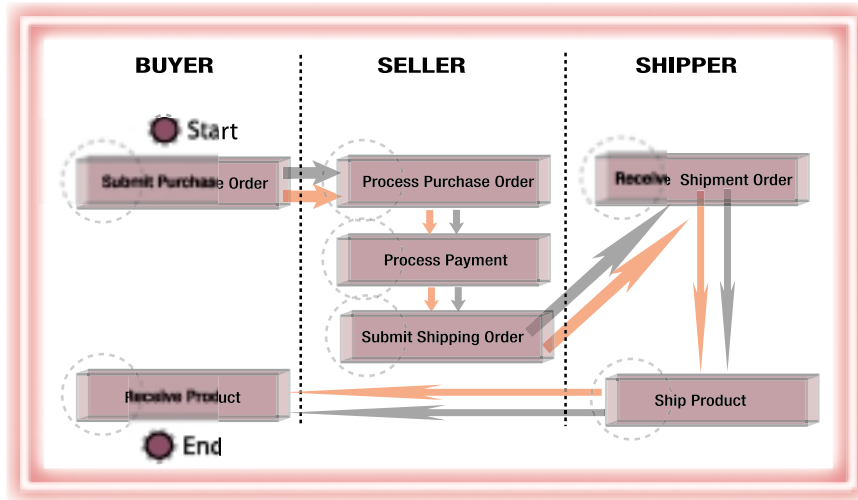


FIGURE 1 | Sample Business Process

mes: flow composition, orchestration, and choreography are just three of the most common synonyms.

- **Global model:** Simply modeling the processing flow between activities within a workflow (which in WSFL terms means modeling the processing flow between Web services) is not enough. In addition to the flow model, there needs to be a way of specifying exactly how the Web services involved in the process are expected to interact with each other. That is where the global model comes in. The global model defines the necessary links between Web services that specify how messages will be sent from one service to another as the processing flow defined by the flow model is executed.
- **Recursive composition:** One of the cool things about WSFL is that once you've defined both the global and flow models for a given business process, it's then possible to define the entire process as a single Web service that may be used within other business processes. In other words, you can recursively compose WSFL business processes using other existing WSFL business processes. This enables a great deal of flexibility and the possibility of fine granularity in the models you define. It also opens up the door to some very cool business possibilities.
- **Service provider:** A service provider is the party responsible for performing a particular activity within a business process. In WSFL, every activity is a Web service; therefore, every service provider is a Web

service provider as defined by the Web Services Architecture Document.

- **Service provider type:** In order to maintain a clear separation between the definition of the business process and its implementation, WSFL's flow and global models define each activity as being implemented by specific types of service providers rather than by the specific service providers themselves. The service provider type is defined by a Web Service Interface document using WSDL. Service providers must properly implement the appropriate Web Service Interface in order to be classified as the appropriate type of service provider to handle a particular activity in the business process.
- **Control link:** A control link is the WSFL equivalent to the directed edge (line) that we discussed earlier; that is, it's the mechanism through which the workflow processor walks through each of the activities in the business process.
- **Data link:** The data link, on the other hand, is the mechanism that the workflow processor uses to control the flow of data through the business process. While in most cases the data flow will closely follow the control flow, it's quite possible that the way that information flows through the process is different from the sequence of activities invoked.
- **Transition conditions:** As a business process is being run, the workflow processor must be able to recognize when a particular activity is finished and when the next activity can be determined and invoked. A transition con-

dition is a true-or-false statement that the processor may use to determine the current state of any particular activity.

- **Life cycle interface:** As mentioned above, WSFL business processes are themselves capable of being defined as Web services. The lifecycle interface is the WSDL-defined Web Service Interface that describes the basic set of operations that all WSFL Web services support. These operations include the ability to invoke, suspend, resume, stop, and terminate the business process as well as to inquire as to its current state.

## Getting into the Flow

Walking through the business process example given above, we can easily see the general processing flow through the graph as each of the activities is performed. The keys to the process are the activity and controlLink definitions; these control the actual flow of the process from one activity to the next. For example, Listing 2 describes the flow of control from the submitPO activity to the processPO activity.

The important concept here is simple: each activity is an individual Web service, described by a WSDL document. The controlLink links these Web services together in a sequential order. The dataLink definition, shown below, defines how data is supposed to flow from one activity to the next.

```
<dataLink source="submitPO"
target="processPO">
  <map sourceMessage="purchaseOrder"
targetMessage="purchaseOrder"/>
</dataLink>
```

The messages that the sourceMessage and targetMessage attributes are referring to reference the WSDL-defined messages for both the source and target Web services. The dataLink element maps the two messages together, essentially stating that the sourceMessage is the same as the targetMessage.

What isn't shown in this WSFL example are the controlling transition conditions that actually determine whether or not each individual activity is currently complete and whether or not the workflow is ready to transition into its next activity.

You should also pay attention to the fact that the sample flowModel references three types of service providers: the buyer, the seller, and the shipper.



```

<serviceProvider name="buyer"
type="buyer" />
<serviceProvider name="seller"
type="seller" />
<serviceProvider name="shipper"
type="shipper" />

```

Each of these represent the significant roles that must be filled to complete the business process. Any Web service provider who properly implements the buyer, seller, and shipper Service Provider Type definitions may fill these roles. Once an appropriate service provider has been identified, a reference to that provider may either be directly referenced with the WSFL document using locator elements, or the workflow engine may decide how exactly the links are resolved. This mechanism has

been designed to give WSFL a great deal of flexibility, allowing business processes to be defined regardless of who is actually going to be responsible for implementing each of the individual activities.

## Looking Ahead

WSFL is just a beginning, a first step into the potentially revolutionary changes that the Web services architecture can make in the way we tie together applications and businesses. My recommendation is that you learn more about WSFL, and the Web services architecture in general. I've provided some basic resources below to help you get started.

## Resources

- If you're new to the area of Web services, I

recommend that you read the Web Services Conceptual Architecture document to get a basic understanding of the fundamental concepts.

- WSFL builds heavily on the Web Services Description Language (WSDL), an XML-grammar for describing individual Web services.
- The WSFL specification represents IBM's input into a growing collection of XML-based workflow definition grammars that include the ebXML Business Process Specification, the Business Process Modeling Language, and Microsoft's proprietary XLANG that's at the heart of their BizTalk Server.
- The Web Services Flow Language specification. ©

### Listing 1

```

<flowModel name="totalSupplyFlow"
serviceProviderType="totalSupply">

```

```

<serviceProvider name="buyer" type="buyer" />
<serviceProvider name="seller" type="seller" />
<serviceProvider name="shipper" type="shipper" />

```

```

<activity name="submitPO">
  <performedBy serviceProvider="buyer" />
  <implement>
    <export>
      <target portType="totalSupplyPT"
operation="submitPO" />
    </export>
  </implement>
</activity>

```

```

<activity name="processPO">
  <performedBy serviceProvider="seller" />
  <implement>
    <export>
      <target portType="receivePO"
operation="receivePO" />
    </export>
  </implement>
</activity>

```

```

<activity name="processPayment">
  <performedBy serviceProvider="seller" />
  <implement>
    <export>
      <target portType="totalSupplyPT"
operation="processPayment" />
    </export>
  </implement>
</activity>

```

```

</implement>
</activity>

<activity name="submitShippingOrder">
  <performedBy serviceProvider="seller" />
  <implement>
    <export>
      <target portType="totalSupplyPT"
operation="submitShippingOrder" />
    </export>
  </implement>
</activity>

```

```

<activity name="receiveShippingOrder">
  <performedBy serviceProvider="shipper" />
  <implement>
    <export>
      <target portType="totalSupplyPT"
operation="receiveShippingOrder" />
    </export>
  </implement>
</activity>

```

```

<activity name="shipProduct">
  <performedBy serviceProvider="shipper" />
  <implement>
    <export>
      <target portType="totalSupplyPT"
operation="shipProduct" />
    </export>
  </implement>
</activity>

```

```

<controlLink source="submitPO" target="processPO" />
<controlLink source="processPO" target="processPayment" />

```

```

/>
<controlLink source="processPayment"
target="submitShippingOrder" />
<controlLink source="submitShippingOrder"
target="receiveShippingOrder" />
<controlLink source="receiveShippingOrder"
target="shipProduct" />

<dataLink source="submitPO" target="processPO">
  <map sourceMessage="purchaseOrder"
targetMessage="purchaseOrder" />
</dataLink>
<dataLink source="processPO" target="processPayment">
  <map sourceMessage="purchaseOrder"
targetMessage="purchaseOrder" />
</dataLink>
<dataLink source="processPayment"
target="submitShippingOrder">
  <map sourceMessage="purchaseOrder"
targetMessage="shippingOrder" />
</dataLink>
<dataLink source="submitShippingOrder"
target="receiveShippingOrder">
  <map sourceMessage="shippingOrder"
targetMessage="shippingOrder" />
</dataLink>
<dataLink source="receiveShippingOrder"
target="shipProduct">
  <map sourceMessage="shippingOrder"
targetMessage="shippingOrder" />
</dataLink>
</flowModel>

```

#### Listing 2

```

<activity name="submitPO">
  <performedBy serviceProvider="buyer" />
  <implement>
    <export>
      <target portType="totalSupplyPT"
operation="submitPO" />
    </export>
  </implement>
</activity>

<activity name="processPO">
  <performedBy serviceProvider="seller" />
  <implement>
    <export>
      <target portType="receivePO"
operation="receivePO" />
    </export>
  </implement>
</activity>

<controlLink source="submitPO" target="processPO" />

```

Download the code at  
[sys-con.com/webservices](http://sys-con.com/webservices)

SAVE 30% off the annual newsstand rate

BUSINESS & TECHNOLOGY  
**wireless**

Offer subject to change without notice

ANNUAL NEWSSTAND RATE
<del>\$71.88</del>
YOU PAY
<b>\$49.99</b>
YOU SAVE
<b>30%</b> Off the Newsstand Rate

## DON'T MISS AN ISSUE!

Receive 12 issues of **Wireless Business & Technology** for only \$49.99! That's a savings of 30% off the cover price. Sign up online at [www.sys-con.com](http://www.sys-con.com) or call 1 800 513-7111 and subscribe today!

## Wireless Business & Technology:

### Wireless Checks In to Hospitality

Wireless is transforming the hotel experience for guests and employees alike.

### Is Worldwide Wireless Broadband Barrelling Our Way?

As wireless broadband connectivity spreads around the globe, what technologies are taking us there?

### Making Money from Messages

SMS might be the solution to the growing debt problem faced by companies that massively overbid for 3G licenses.

### Mobile Operators Seek Incremental Revenue from the Mobile Internet

Leveraging infrastructure for ROI.



# Easy As Apple Pie

*Focus on building new applications instead of old infrastructures*

**C**arl Sagan once said, "In order to make an apple pie from scratch, you must first create the universe."

Although Carl wasn't especially known for witty quips, that one should strike a chord with developers creating applications for the Internet. Because that's the approach many developers are taking – creating a lot of ancillary (but necessary) programs in order to get to the apple pie – what they originally set out to do.

You can get there from here, but it's a long way to go, and in today's economy who has the time or money? There is a shortcut, though, one that's helped youknowbest build our Web commerce services faster, easier, and with greater security. Microsoft's .Net My Services provided us with ready-made solutions to a number of basic issues that allowed us to focus on increasing our value to consumer users, manufacturers, and online retailers.

## The Center

.NET My Services represents a core set of Web services built around the idea of centralizing and protecting user information. Users have complete control over access to their data, including the ability to specify the length of time a third party can access their data. It includes services such as authentication, authorization, alerts, and data storage, ultimately allowing users to create a

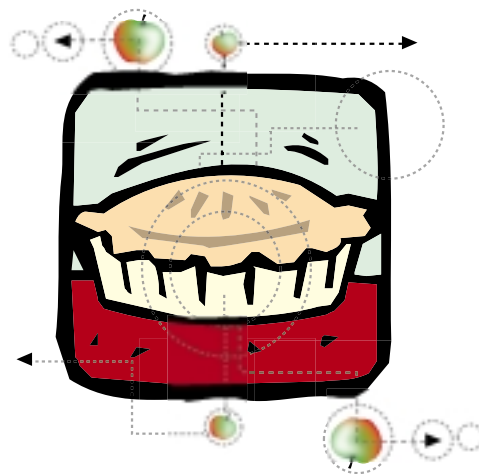
single online presence that they control. Applications like youknowbest's myList can request information from these services instead of asking the user for it. This allows the user to control a single instance of their information and simply grant permission for other applications to access it.

While .NET My Services is centered on users and their data, an amazing amount of work has been put into supporting companies that wish to offer services to the user and access this data. We've been able to leverage this framework to help build our universal shopping list faster, with fewer resources and more features.

Our universal shopping list (myList.com) allows users to save products from anywhere and get information, such as pricing and availability. As with most things, it's easy to say, not quite so easy to do. It isn't just about saving product names in a database and regurgitating them back to the user. It's about creating a single instance of a product that understands information such as its manufacturer, facts, retailers, and offers, and allowing people to access this information through their lists.

In developing myList, we found that our business has two sides. One side centers on the users, the information they store, and the functionality required in making our service compelling. The other side is completely focused on gathering and maintaining information on products, manufacturers, and vendors. Both of these two extremely different tracks are required for our product to be successful.

So how does a small startup work on two totally different businesses at once? The answer: Web services; well, more specifically Microsoft's .NET My Services. We found that .NET My Services provides functionality that simplifies and speeds up application development. With any user-oriented application, there are some standard issues that



need to be addressed. How will users authenticate themselves? How will user data be collected and where will it be stored? How will this data be protected? We'd seen these problems before, but were facing them again. Without .NET My Services, myList would be just another application that requires a user to have a user name and password. It would be another application that would ask for information like the user's e-mail address or name. Our service would be yet another application storing user-centric data in yet another place. We would spend time and money building a system that had been built a hundred times before.

## The Framework

The .NET My Services framework is being developed to solve these issues and we have found that it does. A good example is .NET Passport (formerly known as Microsoft Passport). Instead of our making users create new accounts consisting of another user name and password to remember, we simply allow them to sign in using their .NET Passport account.

This works from the users' point of view because they don't have to create an account on yet another Web site or give us any personal information. Users are able to use the same user name and password they use at sites such as MSN and eBay. All authentication happens through secure



### Author Bio

Tim Tryzbiak is the lead software developer for youknowbest, a Celebration, Florida-based developer of Web commerce services for manufacturers. He has extensive experience in programming Web services, along with component-based programming and XML. Previously, he served as a computer consultant and an Army Intelligence analyst.  
TIM.TRYZBIAK@YOUKNOWBEST.COM



# WebServices

.NET J2EE XML JOURNAL

COMING IN THE  
April ISSUE

- e** Using Web Services with J2EE  
*Moving into a technology-independent world*
- e** Asynchronous Web Services  
*Deployment based on JMS that extends SOAP-over-HTTP*
- e** Dynamically Converting Existing Java Code to a Web Service  
*A robust and comprehensive framework for delivering Web services*
- e** Web Services Over P2P Networks  
*The technology is ready for an interesting intersection*
- e** Knowing the Score - Web Services and Business Processes  
*The promise of a new era in e-commerce*

## WSJ ADVERTISER INDEX

ADVERTISER	URL	PHONE	PAGE
BaseBeans Engineering	www.basebeans.com	866-460-2320	65
BEA eWorld	www.bea.com/events/eworld/2002	404-240-5506	21
engindata Research	www.engindata.com		37
HP Bluestone	www.hpmiddleware.com/download	856-638-6000	67
Java Developer's Journal	www.sys-con.com	800-513-7111	39
JavaOne	http://java.sun.com/javaone	888-886-8769	23
JDJ Store	www.jdjstore.com	888-303-JAVA	55
Loox Software	www.loox.com	800-684-LOOX	33
Mongoose Technology	www.portalstudio.com		25
New Atlanta Communications	www.newatlanta.com		31
Object Management Group	www.omg.org/mda	781-444-0404	17
ParaSoft	www.parasoft.com/ws2	888-305-0041	29
Quintessence	www.in2j.com		15
Sams Publishing	www.samspublishing.com	800-571-5840	13
Sonic Software	www.sonicsoftware.com		2, 3
SpiritSoft	www.spiritsoft.net/downloads		4
Sun Microsystems	www.sun.com/sunoneinfo		10, 11
SYS-CON Events	www.sys-con.com	201-802-3069	52, 53
SYS-CON Media	www.sys-con.com	800-513-7111	41, 55
Web Services Edge Conference & Expo	www.sys-con.com	201-802-3069	56, 57
Web Services Journal	www.sys-con.com	800-513-7111	45
WebSphere Developer's Journal	www.sys-con.com	800-513-7111	65
Wireless Business & Technology	www.sys-con.com	800-513-7111	35
Wireless Edge Conference & Expo	www.sys-con.com	201-802-3004	62, 63
XML Global Technologies	www.xmlglobal.com/newangle	800-201-1848	68
XML-Journal	www.sys-con.com	800-513-7111	60

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.

# WebServices

.NET J2EE XML JOURNAL

The world's leading independent Web Services information resource  
Get Up to Speed with the Fourth Wave in Software Development

- Real-World Web Services: Is It Really XML's Killer App?
- Demystifying ebXML: A Plain-English Introduction
- Authentication, Authorization, and Auditing: Securing Web Services
- Wireless: Enable Your WAP Projects for Web Services
- The Web Services Marketplace: An Overview of Tools, Engines, and Servers
- Building Wireless Applications with Web Services
- How to Develop and Market Your Web Services
- Integrating XML in a Web Services Environment
- Real-World UDDI
- WSDL: Definitions and Network Endpoints
- Implementing SOAP-Compliant Apps
- Deploying EJBs in a Web Services Environment
- Swing-Compliant Web Services
- and much, much more!

www.wsj2.com

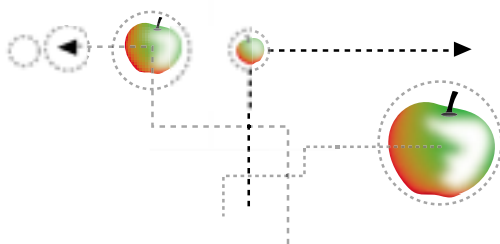
\*OFFER SUBJECT TO CHANGE WITHOUT NOTICE

Only \$69.99 for 1 year (12 issues)  
Newsstand price \$83.88 for 1 year

Special  
Introductory  
Offer  
SAVE \$13.89\*







means and is hosted by Microsoft. Using a server-side component known as the "Passport Manager," youknowbest simply gets a Passport Unique ID (PUID) and a confirmation that this PUID is authenticated. Any data we store or actions we take are on behalf of this PUID. In truth, we have no idea who this PUID belongs to. Users can prevent any application from accessing their e-mail or even their name. To users, this means privacy. For us, .NET Passport is a re-lia-ble and secure way to allow our users to access our application without having to support this functionality ourselves. There are many nuances in an authentication system: registration screens, secure connections, encryption, support pages, support personnel, and so on. When you think that we were able to deliver this in two weeks, you begin to see the benefits of .NET Passport.

While we had to make some compromises on the look of our sign-in page, we felt that the benefits overwhelmed the small user interface issues we had. In the end, we expect to see larger adoption rates because users don't have

to give us any personal information before they try our application. They can sign in to our service with their .NET Passport account and never be asked for anything more.

### .NET Alerts

.NET Passport is really just the beginning. By leveraging the other services, we're able to spend less time building infrastructure and more time adding features or focusing on other aspects of our business. A good example of how we leveraged Microsoft's .NET My Services is our integration with .NET Alerts.

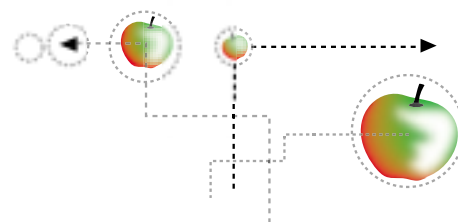
We're currently working on a feature that will allow our users to request notification when certain product conditions are met. For example, let's say you've saved the latest PDA into your list and you want to be notified when the price drops or when a particular retailer begins selling the PDA. When the conditions have been satisfied, we use .NET Alerts to notify you. youknowbest is still responsible for determining when a retailer has met the target price or has come online, but we don't have to worry about how to send the notification. We simply call the .NET My Services Web service and pass it the message. Based on criteria you have set with .NET Alerts, the message could come to your computer as a Windows Messenger alert or an e-mail, or be sent to another device. The point is that youknowbest doesn't have to architect and implement this infra-

structure. As long as the user gives us permission to send an alert, we simply send it to the right Web service and allow .NET My Services to worry about delivery mechanisms.

Combining .NET My Services with myList has made a huge impact on our development time. Not having to architect and build something like .NET Alerts allowed us to provide advanced functionality in half the time. While .NET My Services is still in development, we have found it to be incredibly easy to work with. Sending a message, like a request for an alert, to .NET My Services is just a matter of creating an XML packet, sending it to the right Web service, and verifying the response. With tools like Microsoft Visual Studio .NET, it's even easier than that.

### Privacy

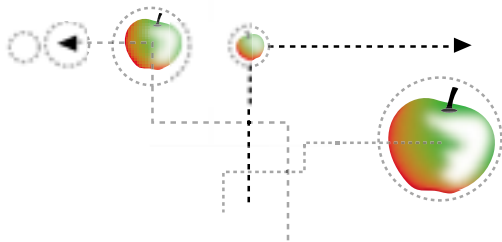
Understanding how to send a request to a Web service is different from being authorized to act on the user's behalf. A core tenet of .NET My Services is privacy, which was one of the deciding factors for us in determining whether to work with



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

**web services** **EDGE**  
world tour 2002

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



Microsoft. Privacy is a huge issue and trusting another company with your users' data is not to be taken lightly. What we found is that Microsoft and .NET My Services are serious about privacy and that users are truly in complete control of their data.

When an application like myList requests or sends data to one of the .NET Services, a check is made to determine if the application is authorized to perform the transaction on behalf of the user. If not, the user can consent on the spot. Like .NET Passport, it's handled over a secure line and hosted by Microsoft. If users consent, they can set an expiration or allow the application to have unlimited access to their data. Either way, users can manage these consents and revoke them at any time. This puts users in control of their data and who has access to it. The paradigm shifted from us owning and controlling the users' data to users sharing their data with us using .NET My Services as the broker.

Since users can grant or block access as they see fit, we were forced into a different mindset when developing our application. After evaluating our options, we made the decision to not store any user-oriented data on our servers. While there may be support for caching of this data, we decided to request information from .NET My Services as needed. Although this may not be in our best interest, it *is* in our users' interests and would ensure that any data they share with us is fresh. If the user decides not to share their information, we simply won't perform the particular task. Pricing and availability for a product is a good example of this.

We're working on a feature for our service that uses the user's location in determining pricing and product availability. If a user elects to share their location (zip code or city and state), we'll be able to find stores in their area that sell the products they're interested in. Prices from online retailers could also vary depending on where the products have to be shipped. However, this location-sensitive data requires us to pull information from the users .NET Profile and we can do this only if they have given us their consent. It lets users determine if sharing their data with myList is worth the feature they want to use.

Pulling the zip code may not seem like a big deal, but imagine if the user moves and updates their .NET Profile. If we weren't using .NET My Services, we'd have to hope they updated their zip code on

our Web site through some account management utility. With everything a user has to do, this isn't likely. By subscribing to Microsoft's service, we get this update automatically (with the user's permission). The next time they request product pricing, we'll use the right zip code and the user never had to change it on our site. Expand this to e-mail addresses, shipping addresses, billing addresses, or any other information that needs to be accessed. We'll always have the latest information without having to ask users to enter it over and over again.

## Conclusion

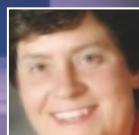
A tremendous amount of work is already done for us simply by using Microsoft's .NET My Services. We gain by not having to create a user support system or add infrastructure to support things like notifications. This has made development time shorter and allowed us to take resources from the myList side of our business and put them against other things, like understanding products. Have we had to make some concessions? Sure, but none of them outweigh the fact that we have enriched our users' experience, given them complete control over their data, and done it all in less time and with less effort than if we had to do it ourselves.

In short, using .Net My Services means the mechanics for creating myList become easy as apple pie – and allows us to avoid any Big Bangs along the way. Carl would be proud. ©

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

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

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



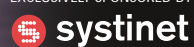
## PRESENTERS...

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



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

EXCLUSIVELY SPONSORED BY



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

**BOSTON, MA** (Boston Marriott Newton) **SOLD OUT!** **JANUARY 29**  
**WASHINGTON, DC** (Tysons Corner Marriott) **SOLD OUT!** **FEBRUARY 26**  
**NEW YORK, NY** (Doubletree Guest Suites) ..... **MARCH 19**  
**SAN FRANCISCO, CA** (Marriott San Francisco) ..... **APRIL 22**

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



# Making Second-Generation Web Services Secure

First-generation Web services use existing technologies like SSL, PKI, and access control products...  
what's next?

**W**hen you hear the word *security*, what comes to mind? SSL? Firewalls? Authentication? Authorization? B-52 bombers? Security means different things to different people, but in the context of securing applications, we can think of security in two parts: access control and secure communication.

## Existing Measures

*Access control* encompasses a number of concepts, including identity management, authentication, authorization, and auditing. It includes knowing who your users are, knowing

what they can do in your applications, and keeping a record of what they have done. There are a variety of products on the market today that provide cost-effective and manageable



solutions for securing Web-based applications. Access control products work by integrating with a directory of user identities, typically stored in an LDAP directory or a relational database, and use the user identity and entitlement information to drive a set of shared services including user authentication and authorization. By centralizing access control as a set of shared services, security for all applications, regardless of the language or protocol, can be centrally managed in one place rather than in each application.

*Secure communication* involves keeping the conversation between two parties private. For example, when you purchase something at your favorite online store, your browser and the Web server it is interacting with is (hopefully) using an SSL connection to communicate. This technology prevents others on the Internet from "spying" on the transaction, ensures that your credit card information is transmitted securely, and guarantees that the order you enter in your browser can't be manipulated on its way to the online store.

Access control and secure communication are probably the two most important aspects of security that need to be addressed for Web services. While the concepts are the same, new technologies and products need to be developed to meet the requirements of this new Web services world. We'll take a closer look at the requirements that Web services imposes on these two areas and explore some of the solutions available by looking at a case study of an office supply company, PENS, Inc. PENS wants to create a set of Web services for its customers to use to place orders. Customers will then be able to submit purchase orders in the form of XML documents and send them electronically to the PENS Web service.

## Authenticating and Authorizing Users of a Web Service

The first generation of the Web service will utilize PENS' existing access control infrastructure, invested in browser-based Web applications. These applications are typically protected by asking the users to identify themselves and provide some sort of a shared secret, such as a password. Because Web service requests aren't going through a browser, asking the user for their user name and password after receiving the XML

### Author Bio

Jim Ducharme is a director of development at Netegrity, Inc., the leader in securely managing e-businesses. He is responsible for the development of new product initiatives at Netegrity aimed at securing Web services and XML-based B2B infrastructures.  
JDUCHARME@NETEGRITY.COM

**// The most important thing you can do to prepare for securing Web services in the future is to ensure that you have a scalable and flexible security infrastructure in place that will continue to grow with them as new technologies emerge"**



# THE INSIDER INTELLIGENCE YOU NEED...

## TO KEEP AHEAD OF THE CURVE

**FREE** E-Newsletters  
SIGN UP TODAY!

Go to [www.SYS-CON.com](http://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](http://www.sys-con.com).

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



Don't Delay!  
Subscribe  
for **FREE!**

at [www.sys-con.com](http://www.sys-con.com)

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

**SYS-CON**  
MEDIA



document isn't possible. The information needed about the user has to be available when we receive the XML document.

### SSL

This could be accomplished with an existing access control product, using client-side certificates for authentication. In order to send their XML purchase order, all of PENS' customers would require a certificate that they could attach to an SSL connection to the PENS server. The user's certificate information would be passed along with the XML purchase order. When the access control product intercepts the XML message, it authenticates users based on their certificate information. This solution isn't feasible for PENS, however, because it doesn't have a PKI infrastructure in place nor does the company want to start handing out or managing certificates for all its customers.

### SOAP

Another solution might be to have the user's credentials inside the XML message itself. Utilizing SOAP, PENS could require that the sender put their user ID and password in the SOAP header. SOAP was designed to allow for this additional information without having to change the schema or DTD of the purchase order. This information can easily be taken out of the message once it has been processed. When the message arrives at PENS, the credentials would be extracted from the SOAP headers and used to authenticate and authorize the user.

Sounds simple enough, but extra care needs to be taken to ensure that the XML message is securely transmitted to the server. If someone were to intercept the message they'd now have the user's ID and password. In order for this to be a truly secure transaction, purchase orders must be transmitted directly from the user placing the order to the PENS server. This is not an acceptable long-term solution since that scenario may not always be true.

### XML Signatures

A possible third solution is based on a new technology developed specifically for binding user identity to an XML document: XML signatures. XML signatures do just what their name implies, they allow someone to electronically sign an XML document, thereby binding their identity to it. This is no different conceptually what you do when you sign a paper purchase order: you give your approval by putting your identity on the document. PENS can now require that all purchase orders sent to the Web service be digitally signed with XML signatures. When the purchase order arrives, the signature can be used to determine who sent the document and prove that the document hasn't changed since the person signed it.

XML signatures depend on private and public key pairs. The signature itself is created using the sender's private key information and can be verified by the recipient of the message using the sender's public key. The only requirement for this solution is that PENS has to know the user's public key. While there are several ways to do this, one of the simplest is that the user's public key could simply be part of the user's profile in the PENS directory. (While XML signatures allow for the public key information to be passed along with the message, there still needs to be some mechanism to identify the holder of this public key with the user record in the PENS, Inc. directory.)

### Moving Beyond Existing Measures

The three examples above represent what will most likely be the prevalent security models for the first generation of Web services. The element they share is that the provider of the Web service maintains the user identity information and knows how to authenticate each of its users. It requires that anyone who wishes to use the Web service establish an identity and provide some information so that

the Web service can authenticate them. But as Web services evolve, these assertions may not be true or practical.

### Hosted Identity Services

As PENS, Inc., becomes more popular, it becomes more difficult for the company to manage the thousands or millions of user identities in its directories. Most of the information maintained about each of its users is identical to the information that other Web sites and Web services maintain about their users. If there were a single user profile somewhere that could be shared across all these Web services, it would not only make it easier for users to manage their own profile information, it would also mean that each of the individual sites would no longer have to manage all the information. In addition, if this service were to provide the ability to authenticate those users, you would have a very valuable service for PENS' Web service.

### Passport

This model is known as a hosted user identity and authentication service and is the goal of Microsoft's Passport technology. It's already in use in many Web sites to provide a single user identity and authentication service. While today it's limited to browser-based applications, the Passport technology holds the promise of working with XML-based Web services as well. With Passport, users create and manage their user profile in one place only. Sites or Web services that integrate the Passport service are able to access this profile information once the user has been properly authenticated by Passport. This means that the Web service provider no longer has to provide profile management or authentication services for these users.

The requirements for this model to work are straightforward. First, the Web service provider will need to integrate the shared service software into their Web service infrastructure. Second, the Web service provider must trust the hosted service to perform this functionality on their behalf, and have confidence that the information provided is accurate and secure. Finally, the Web service provider needs to believe that users are willing to use the Passport service to store and manage their information. Since this service would be the only source of user identity, only users of that service would be able to place orders at PENS. Because of these requirements, these solutions tend to not completely replace a site's own identity and authentication service, but

**Access control, security and infrastructure vendors incorporating SAML into their products and usable solutions should become available throughout 2002"**

THE LARGEST INTERNATIONAL

# XML

## CONFERENCE & EXPO IN THE WORLD!

**WIN A  
\$35,000  
LUXURY CAR!**



ATTENDEES WILL BE INVITED TO TAKE A GOLF SWING  
TO WIN AND RIDE OFF IN A \$35,000 LUXURY CAR!

## XML-NEXT G OF ENTERPRISE DEPLOYMENT

### REGISTER ONLINE TODAY

FOR LOWEST CONFERENCE RATES  
EARLY SELL-OUT GUARANTEED!

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

### Focus on XML

XML is today's essential technology to develop and deploy Web services. Here's your chance to learn from expert practitioners how XML is making the next phase of the Web a reality.

Focus on standards, interoperability, content management, and today's new quest for more efficient and cost-effective Internet and intranet-enabled business process integration.

Focus on XML during information-packed days while you enjoy an XML/Web Services Keynote panel, comprehensive conference sessions, and an unparalleled opportunity to exchange ideas with peers and industry leaders.



### A Sampling of XML-Focused Sessions

- XML STANDARDS - AN OVERVIEW
- OASIS STANDARDS UPDATE
- UBL - A UNIVERSAL BUSINESS LANGUAGE
- BRINGING XML TO PKI
- LINK MANAGEMENT WITH XLINK
- PRACTICAL XSLT AND XPATH
- ENTERPRISE CONTENT MANAGEMENT WITH XML
- XML IN THE ENTERPRISE AND INTER-ENTERPRISE WORLD

NORBERT MIKULA  
XML CHAIR  
BOARD OF DIRECTORS, OASIS  
INDUSTRY EDITOR  
WEB SERVICES JOURNAL

### Featuring...

- UNMATCHED KEYNOTES AND FACULTY
- THE LARGEST INDEPENDENT JAVA, WEB SERVICES, AND XML EXPOS
- AN UNPARALLELED OPPORTUNITY TO NETWORK WITH OVER 5,000 I-TECHNOLOGY PROFESSIONALS

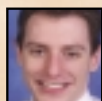
### Who Should Attend...

- DEVELOPERS, PROGRAMMERS, ENGINEERS
- I-TECHNOLOGY PROFESSIONALS
- SENIOR BUSINESS MANAGEMENT
- SENIOR IT/IS MANAGEMENT/C LEVEL EXECUTIVES
- ANALYSTS, CONSULTANTS

Hear these thought leaders in interactive, cutting-edge keynote addresses and panels...



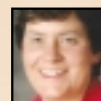
JEAN FRANCOIS ABRAMATIC  
SENIOR VP R&D, FORMER  
CHAIRMAN, W3C • ILOG



TYLER JEWELL  
PRINCIPAL TECH.  
EVANGELIST • BEA



DAVID LITWACK  
CEO  
SILVERSTREAM



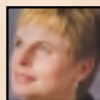
ANNE THOMAS MANES  
CTO  
SYSTEMET



BARRY MORRIS  
CEO  
IONA



RICK ROSS  
FOUNDER  
JAVA LOBBY



PATRICIA SEYBOLD  
FOUNDER & CEO  
SEYBOLD

#### For Exhibit information

CONTACT: RICHARD ANDERSON  
135 CHESTNUT RIDGE RD.  
MONTVALE, NJ 07645  
201 802-3056  
[RICHARD@SYS-CON.COM](mailto:RICHARD@SYS-CON.COM)

**XML**EDGE  
conference & expo

**JUNE 24-27**

JACOB JAVITS  
CONVENTION CENTER  
NEW YORK, NY

**OCTOBER 1-3**

SAN JOSE  
CONVENTION CENTER  
SAN JOSE, CA

#### SPONSORED BY:



#### MEDIA SPONSORS



#### OWNED AND PRODUCED BY



JAVA AND JAVA-BASED MARKS ARE TRADEMARKS OR REGISTERED TRADEMARKS OF SUN MICROSYSTEMS, INC., IN THE UNITED STATES AND OTHER COUNTRIES. SYS-CON PUBLICATIONS, INC. IS INDEPENDENT OF SUN MICROSYSTEMS, INC. ALL BRAND AND PRODUCT NAMES USED ON THESE PAGES ARE TRADE NAMES, SERVICE MARKS, OR TRADEMARKS OF THEIR RESPECTIVE COMPANIES.

rather augment it to reduce the overall burden and provide a convenience feature for the users willing to use the shared service.

Hosted identity services like Microsoft Passport hold great promise but they don't eliminate the need for a Web service provider to centrally maintain user identities. They're not geared towards addressing enterprise identity management. Companies need to maintain full control over their employee identities. They need to be able to decide who, if anyone, gets to see what data. Combine that with the fact that many companies have already made major investments in identity management systems and we see that these hosted identity solutions are not a complete solution for sharing user identity.

### *Sharing User Identity*

Companies will also need the ability to exchange user information with their partners. How can this be achieved securely if the information is stored inside an enterprise's identity management system?

Let's illustrate this. PENS has created a corporate account program for its Web service. This enables authorized users from member companies to place supply orders directly using the Web service. The decision as to who is an authorized user and who isn't, and what spending limit each authorized user has, needs to be managed by the member company and not by PENS. The member company is the one that truly knows the users within their organization and the roles they should have. PENS needs a solution that will allow their corporate customers to manage their own user information while at the same time still allowing the Web service to authenticate and authorize the users of the Web service.

One of those companies, Widgets, Inc., has 35,000 employees and wants every employee to utilize this Web service when ordering office supplies. It wouldn't be practical or secure to copy the 35,000 employee records from the Widget user directory to the PENS user directory. PENS doesn't really need to know each of the 35,000 people personally, all it needs to know is that the Web service request is coming from a valid employee of a company with a corporate account.

### *SAML*

A new XML standard called SAML (Security Assertions Markup Language) is being developed to help solve this problem. The SAML specification is currently being finalized within the OASIS XML-Based Security Services

Technical Committee. SAML allows two entities to share identity, authentication, and authorization information. SAML is based on the concept of assertions. These assertions effectively allow one entity, an authentication authority, to state facts they know about a user to another entity, in our case the Web service provider. With SAML and Web services, the trust relationship is between the Web service provider and the authenticating entity, and not the user of the Web service directly.

The way SAML would work in our use case is that the employee of Widgets who wants to place an order with PENS would authenticate himself or herself using the Widgets authentication provider (using whatever mechanisms are in place). Once authenticated, the user would ask the authentication provider for a set of SAML assertions to send to PENS, along with the purchase order. These assertions, which are nothing more than XML fragments, would state that Widgets, Inc., authenticated the user. They would also contain basic information about the user, along with entitlement information, such as what their authorized spending limit is. To ensure that these assertions can't be modified and to prove who generated them, they're digitally signed by the authentication engine. These assertions would be sent along with the XML purchase order to PENS.

When the message arrives at PENS, the SAML assertion is found in the message and used for authentication. Rather than authenticating the end user, PENS is really ensuring that the SAML assertion was created by a trusted partner, in this case Widgets, Inc., and that the assertion hasn't been modified. Once authenticated, the spending limit information about the user can be extracted from the SAML assertion to determine if the user is authorized to make this purchase.

This example covers only one aspect of SAML's power. Along with identity sharing, SAML in addition seeks to address issues like remote authentication, remote authorization, single sign-on, and other business-to-business security issues. Access control, security, and infrastructure vendors are actively incorporating SAML into their products and usable solutions should become available throughout 2002. A toolkit called the JSAML toolkit is available from Netegrity, Inc., (see references at the end of this article for the URL). Downloading this will

allow you to test this new technology and get a better understanding of its features and power.

### *Liberty Alliance*

Another solution is the Liberty Alliance project. Started in September 2001, this organization is working to deliver a solution to the concept of "federated" user identity. While there appears to be a lot of muscle behind the effort, at the time of this writing there was little more than a set of white papers, marketing slides, and a press release available. Hopefully the alliance will work well together to quickly deliver a solution.

### *XML Key Management*

Finally, another new effort originating from the W3C to address sharing identity information is the XML Key Management working group. Based on an early effort known as XKMS (XML Key Management specification), this group is chartered with creating a solution that will simplify the integration of PKI solutions with Web services. Many of the new XML technologies dealing with security, such as XML Signatures, SAML, and XML Encryption, leverage public/private key pairs as part of their implementation. XKMS provides a convenient and standard mechanism for transporting, locating, validating, and registering key information. There are various toolkits available that illustrate the use and functionality of XKMS. It's not clear, however, how much of this specification will survive the W3C process.

### *Keeping Content Secure*

Another aspect of security is how to keep the conversations between a client and a Web service secure. Given that these XML messages may contain confidential information, it's critical that there be mechanisms in place to keep the content private.

Just as SSL has been the primary means for securing communication between a browser and a Web service, it can provide a similar service for Web services. Enabling server-side-only SSL on the Web service platform ensures that the messages between the client and Web service are protected from prying eyes. The only limitation to the use of SSL with Web services is that SSL secures a single client-to-server communication channel. While this is acceptable for securing browser-to-Web-server communication, it may not be enough



for Web service-based applications. As these applications evolve, they will go beyond a simple request to a single service. Web service-based applications will consist of a number of disparate Web services aggregated together to work over the same XML message. These Web services may be distributed across the Internet, hosted by different providers, or have different rights over the information within the message. While SSL could certainly be used to secure the communication from one service to another, once the message has arrived at a particular Web service, the service has equal rights over the content of the message and is expected to keep the content of the message secure in its environment. What's really needed is the ability to secure the content – not just the communication.

### XML Encryption

To address some of these issues, a working group within the W3C is working on a new XML-based standard called XML Encryption. This standard will provide the ability to encrypt digital content such as entire XML documents or selected portions of them. The encrypted document could be sent through unsecured protocols, handed to untrusted intermediate services, or stored indefinitely in unsecured repositories. Only the intended recipient of the document will be able to decrypt it and see the protected content. Different portions of the document can also be encrypted for use by different recipients. This would be very useful for protecting sensitive documents like medical records. Your entire medical record could be captured as digital content but only certain portions of the record would be made viewable to authorized medical or insurance personnel. XML Encryption will protect the content being transmitted rather than just securing the communication channel between two parties, as SSL does.

### Conclusion

The biggest issue with the various technologies and solutions described above is their immaturity and the present lack of commercial products to

support these technologies and ease the process of integrating them. To take advantage of any of these technologies today requires a fairly significant integration effort. This situation will certainly improve as vendors integrate these technologies into their products and as the standards solidify.

The good news is that the first generation of Web services currently in development doesn't necessarily require all of these solutions to be in place in order to deploy them. These first-generation Web services will utilize existing technologies like SSL, PKI, and access control products for security.

The most important thing you can do to prepare for securing Web services in the future is to ensure that you have a scalable and flexible security infrastructure in place that will continue to grow with you as new technologies emerge. Centralized access control and identity management products have proven themselves to be invaluable in securing Web applications and reducing maintenance costs of the current generation of Web-based applications. These products must continuously evolve to support the new application models and provide the same level of protection, if not greater, and cost savings for these next generation applications. The model of how to secure applications doesn't need to change with the arrival of Web services, but there just may need to be new technologies to help support the model.

### References:

SAML: [www.oasis-open.org/committees/security](http://www.oasis-open.org/committees/security)  
 JSAML Toolkit: [www.netegrity.com/products](http://www.netegrity.com/products)  
 JSAML Whitepaper: [www.netegrity.com/files/JSAMLwhitepaper.pdf](http://www.netegrity.com/files/JSAMLwhitepaper.pdf)  
 Securing Web Services Whitepaper: [members.netegrity.com/access/files/TransactionMinder.pdf](http://members.netegrity.com/access/files/TransactionMinder.pdf)  
 Microsoft Passport: [www.passport.com](http://www.passport.com)  
 Liberty Alliance project: [www.projectliberty.org](http://www.projectliberty.org)  
 XML Signature: [www.w3.org/Signature](http://www.w3.org/Signature)  
 XML Encryption: [www.w3.org/Encryption/2001](http://www.w3.org/Encryption/2001)  
 XKMS: [www.w3.org/2001/XKMS](http://www.w3.org/2001/XKMS) ©

# SUBSCRIBE AND SAVE

## XML JOURNAL

Offer subject to change without notice

ANNUAL NEWSSTAND RATE

~~\$83.88~~

YOU PAY

**\$77.99**

YOU SAVE

**\$5.89** Off the Newsstand Rate

## DON'T MISS AN ISSUE!

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

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

## Here's what you'll find in every issue of XML-Journal:

- Exclusive feature articles
- Interviews with the hottest names in XML
- Latest XML product reviews
- Industry watch





# Web Services' Impact on Business Process Management

*The promise of a single solution for integration across multiple enterprises*

**W**eb services have been advertised as the one-size-fits-all solution for all sorts of integration problems. But like any other innovation in the software industry, Web services require a new generation of tools and infrastructure to help companies overcome the adoption hurdle and take full advantage of the business benefits this technology promises.

In this article I take a closer look at Web services' impact on business process management (BPM) and identify the requirements for a new generation of BPM products that fully leverage Web services and address the resulting architectural changes.

## BPM: Approaches and Trends

Over the past 10 years, different generations of business process management systems have emerged in the marketplace to solve different problems. These systems include:

- Workflows within packaged applications
- Document management work flows
- EAI-based BPM
- B2B-based BPM

There are a number of trends in the marketplace today that alter the landscape for application integration and business process management including:

- Convergence of application development and application integration
- Convergence of B2B and back-end integration
- Emergence of standards-based back-end integration
- Maturity of J2EE application servers
- Web services innovation

The worlds of application integration and application development were separated because traditionally, different vendors provided these two solutions. The pure integration vendors, in either the EAI or B2B markets, focused their efforts on integrating existing systems and applications, investing little effort in supporting new application development. They relied on application platform vendors such as BEA, Microsoft, and IBM to address application development.

According to Gartner (December, 2001): "Through 2006, at least 75 percent of Web services deployed by Global 2000 enterprises will have been implemented through integration of new developments and pre-existing applications (0.7 probability)."

Developers will be looking for a single platform to develop new applications and integrate existing ones. Web services adoption will help accelerate the convergence of these two worlds by providing a standards-based

method to wrap existing applications and business objects for maximum reusability and integration. As a result, several new requirements for BPM products have emerged:

- Ability to call back-end and external Web Services
- Ability to expose a business process as a Web service
- Seamless integration and reusability of existing business objects (e.g., EJBs)
- Support for the development and execution of in-line application code.

Organizations that invested in the J2EE platform will seek a smooth ramp between the underlying J2EE APIs, the application logic stack, and the business process layer. Companies will also look at the deployment aspects

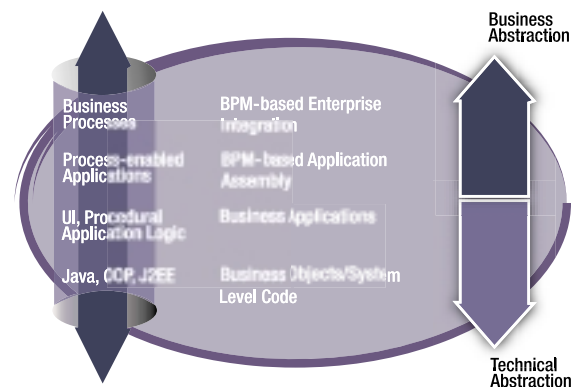


FIGURE 1 Application development and integration continuum

of the platform and evaluate the BPM/integration solution in terms of scalability, reliability, recoverability, and manageability. In the context of an integrated platform, the challenge will be to provide the proper level of abstraction and the tools to the right users (see Figure 1).

The enterprise developer who implements business objects and system-level code will be working at the J2EE API layer. The developer will need mechanisms to expose these objects to the layer above, poten-



### Author Bio

Vittorio Viarengo is director, product management with BEA Systems, a leading application infrastructure company with more than 11,800 customers around the world. Viarengo is responsible for the direction of BEA WebLogic Integration, business process management, and Web Services development framework. [VIVI@BEA.COM](mailto:VIVI@BEA.COM)

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

**BUY THOUSANDS  
OF PRODUCTS AT  
GUARANTEED  
LOWEST PRICES!**

**GUARANTEED BEST PRICES  
FOR ALL YOUR  
WEB SERVICES  
SOFTWARE NEEDS**



## MICROSOFT

**\$2,238.99 Visual Studio .NET Enterprise Architect**

Visual Studio .NET provides developers with the most productive tool for building next-generation applications for Microsoft Windows® and the Web. Visual Studio .NET Enterprise Architect (VSEA) builds on the power of Visual Studio .NET Enterprise Developer by including additional capabilities for designing, specifying, and communicating application architecture and functionality. It enables software architects and senior developers to provide architectural guidance and share best practices across the development team.



## ALTOWEB

**\$3,540.00 Application Platform Release 2.5**

The AltoWeb Application Platform lets you build, deploy, and manage J2EE applications and Web services up to 10x faster without requiring extensive J2EE or Web services expertise. How? By replacing lengthy, custom and complex J2EE, XML and Web services coding with rapid component assembly and reuse.



## MICROSOFT

**\$1,629.99 Visual Studio .NET Enterprise Developer**

With Visual Studio .NET Enterprise Developer, developers can securely version and share their source code, share best practices, target scalable .NET Enterprise Servers, choose from a wide range of third-party tools and technologies, and easily tune the performance of their Web applications and Web Services through the extensive performance-testing tools in Visual Studio .NET.



## WHIZLABS

**\$39.95 WebSphere@Whiz Certification Simulator**

3 Mock Tests (159 Questions). It comes with a test engine and a question bank of 159 questions on the latest pattern of the IBM WebSphere Certification Exam. It also consists of a diagnostic test which will help you know your strengths and weaknesses so you can plan your preparation accordingly.



## SITRAKA

**\$3,999.00 Sale Price \$3,799.00 JClass ServerChart V1.1**

If you've used JClass Chart in your client-side applications, you already know the value of data visualization to your end-users. With JClass ServerChart, you can bring a wide range of charts and graphs right to their Web browsers, giving them real-time access to business-critical data in an intuitive environment.



## SILVERSTREAM

**\$495.00 Extend Application Server Developer Edition (5 User)**

SilverStream eXtend is the first comprehensive, real-world development environment for creating Web Services and J2EE applications. The seamless integration of our proven eBusiness engines and designers gives you the benefits of XML-based, enterprise-wide integration and the power to create, assemble and deploy service-oriented applications.



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

OFFERS SUBJECT TO CHANGE WITHOUT NOTICE

**SAVE UP TO \$100 ON MULTIPLE SUBSCRIPTIONS**

**Special  
online offers**

Pick **4** or **5** and Subscribe  
for one **special low price**



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

Wireless Business & Technology • Java Developer's Journal

Web Services Journal • WebLogic Developer's Journal

XML-Journal • WebSphere Developer's Journal

ColdFusion Developer's Journal • PowerBuilder Developer's Journal

**SYS-CON  
MEDIA**

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

**WWW.SYS-CON.COM/SUBOFFER.CFM**

tially as Web services. At the business application layer, the application developer will need an easy way to assemble these Web services – typically accomplished by writing some procedural application logic or by using a BPM state machine. That developer will have to be sheltered from the complexity of the underlying J2EE layer and from the technical details of assembling Web services. At the integration level, the business analyst defines coarse-grained business processes that use the services provided by the underlying layers as well as back-end resources within the enterprise, and services provided by partners over the Web.

## Convergence of Back-End Integration and B2B

The problem of integrating internal applications on the one hand, and business partners on the other hand, has grown out of two different environments. The first is characterized by a controlled environment with full access to back-end resources, based on messaging systems, short-running transactions, fine-grained access to data, and binary-level transformations. The second environment is characterized by support for XML standards, Internet protocols such as the communication channel, long-running processes and transactions, public processes exposed to partners, and trading partner management. A number of lessons can be learned from the implementation of the first generation of B2B integration solutions.

First, systems and applications developed by different groups on different platforms cannot be tightly coupled; otherwise, changes in the implementation of any of the systems involved will propagate throughout the architecture, making it unmanageable. This breakdown is one of the reasons why the EAI paradigm is insufficient for integration across partners. With Web services, you can integrate applications based on a public contract that describes the XML messages for applications to exchange, while leaving the underlying implementation details to each application. As long as applications honor their contract they can change at will without breaking the integration.

Second, the communication paradigm must be coarse-grained because of the high cost of communicating among loosely coupled systems using WAN or the Internet. Applications ought to maximize return on the

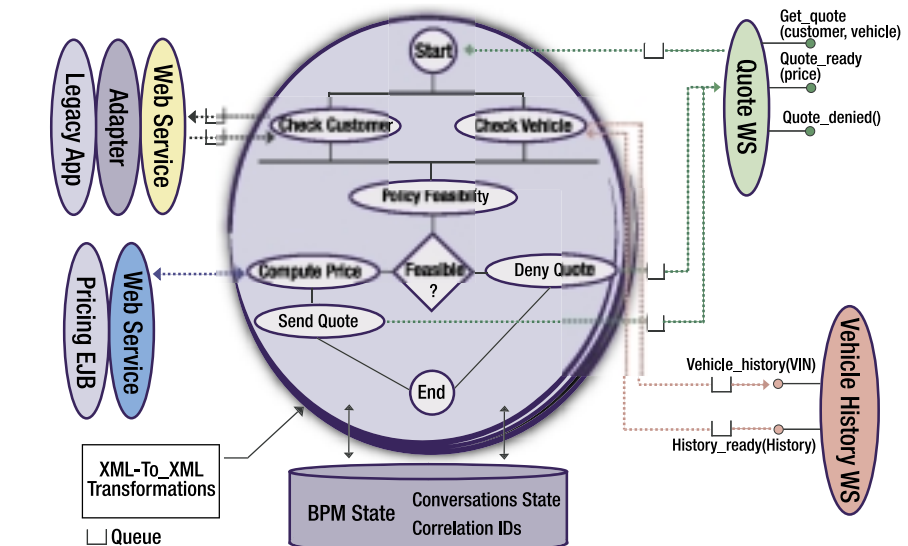


FIGURE 2 | Web service-based BPM example

cost of opening and using the communication channel by passing around larger pieces of information in the form of an XML business document. By integrating at a business level, Web services will allow greater flexibility when the underlying implementation changes.

Third, the communication ought to be asynchronous because you can't rely on other systems, especially legacy applications, to be 100% reliable. Moreover, the application shouldn't be dependent on the response time of another system whose response time may be inconsistent. These architectural changes are profound and will require a new generation of BPM to be designed around them – rather than simply being evolved from previous models.

## Emergence of Standards-Based Integration

Integration is one of the most expensive entries by far in any CIO's ledger, often due to the proprietary nature of back-end systems and applications. XML was the first step toward standardizing this space as it unleashed data from proprietary binary formats into a standards-based data representation. Unfortunately, XML as a data representation alone is not enough. Access to application functionality requires a way to describe the methods available, a mechanism to discover these methods, and a mechanism to access the resulting data. Web services holds this promise.

When XML emerged and started to gain credibility in the marketplace, BPM and in-

tegration companies rushed to add XML support to their products. Most BPM solutions provided a way to consume and produce XML. Many of these products were architected long before XML was introduced; therefore, the architecture wasn't designed to handle its extensible nature. This approach falls short as the number of different XML standards increases. Moreover, the scope of XML standards is quickly moving beyond the mere description and exchange of data.

XML is now pervasive in the architecture and is used to describe services (WSDL), Web service registry (UDDI), business processes (BPML, Xlang), and sequences of public business events and processes (ebXML). For these reasons, BPM engines must provide mechanisms to cope with XML extensibility at their core, so it's possible to support multiple XML standards without requiring changes in the product. At the periphery, BPMs must support XML transformations, definition of public business processes, and interaction with multiple Web services in both synchronous and asynchronous fashions.

The new generation BPM requires mechanisms to integrate with enterprise-class Web services, such as the transformation of XML messages, introspection of WSDL definitions, processing and dispatching of SOAP calls, management of correlation IDs, and the state associated with multiple conversations with multiple Web services. The combination of Web services and BPM provides developers with the business-level programming para-



digm that allows organizations to build what analysts describe as “composite business applications.”

## Example

To better understand the architectural and technical implications of the execution of Web services within a business process engine, let's consider an example in which an insurance company aims to expose an underwriting business process to its subsidiaries using Web services. We'll use a fictitious, simplified version of the actual business process (see Figure 2).

1. The quote request comes in through a Web service interface.
2. The request is queued; when ready, BPM starts the proper business process.
3. BPM starts two parallel tasks: it sends an asynchronous request to a back-end application to check if the customer has any history of interaction with the insurance company and it sends a request to a Web service offered by the Department of Motor Vehicles to check the history of the vehicle to be insured (see Figure 3).
4. When the responses come back asynchronously BPM decides if the request can be accepted by executing in-line Java logic.
5. If the request for quote is accepted, BPM invokes an EJB that computes the policy price based on the input parameters.
6. BPM sends the quote back to the requesting client.

There are two main requirements for the BPM design environment in this example. The first is to provide the user with a business-level

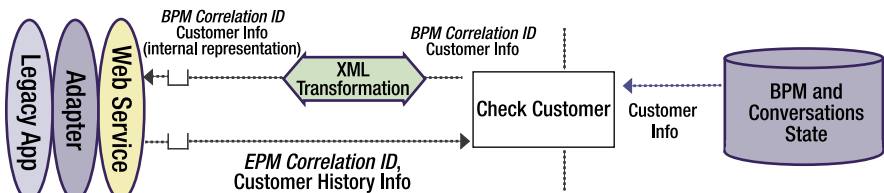


FIGURE 4 | Back-end Web services and BPM interaction (information flow and state management)

independent from the actual Web services with which it interacts. Over time, the same business process will have to interface with alternative Web services implementations (e.g., the credit check could be performed via a cheaper or more efficient Web service offered by a different credit agency), making it possible to swap Web services without changing the business process definition, as long as the semantic of the services methods and schemas are equivalent. At design time, the BPM tool must be able to load the WSDL definition of a Web service and allow the user to select which Web service methods to call. The tool must be able to generate the WSDL definition for those services that are exposed to external entities. It should also allow the user to introspect the schema of the document payload and provide a mechanism to transform it into the schemas that other Web services require.

There will be many cases when Web services won't be transactional, which requires the BPM to provide the user with mechanisms to model compensating transactions when something goes wrong in between multiple invocations of non-transactional Web services. In this example, an adapter is used to get at the

access the various entities involved in the business process (see Figure 4).

Beyond the design time, there are several high-level implications for the BPM at run time. The BPM must provide a mechanism to accept SOAP calls, marshal the SOAP headers and content into the internal data representation, potentially apply a data transformation, and start the proper business process(es). In this example, the interactions with the various Web services are asynchronous. Therefore, each conversation started by BPM must generate a correlation ID and use it to send the proper response to the proper requester later. If an external Web service initiates the request, then BPM needs to store the external correlation ID to correlate the response properly later.

For most business scenarios, the conversations between BPM and other entities via Web services will probably involve multiple exchanges of messages that have state associated with them. For each conversation instance, the BPM run-time will have to transparently and recoverably manage the associated state. For each invocation of a Web service within the boundaries of a defined long-running transaction, the BPM engine has to store the information it needs to compensate for the effect of the Web service invocation. When a message has to be transformed from one format to another as it's passed across different Web services, the BPM engine needs to provide a highly efficient transformation engine.

## Conclusions

Web services offer the promise of a single solution for integration across multiple enterprises. However, there are still many areas where there is opportunity for growth and enhancement. This article has identified and discussed the requirements that would ensure a fully successful new generation of BPM systems using Web services. ©

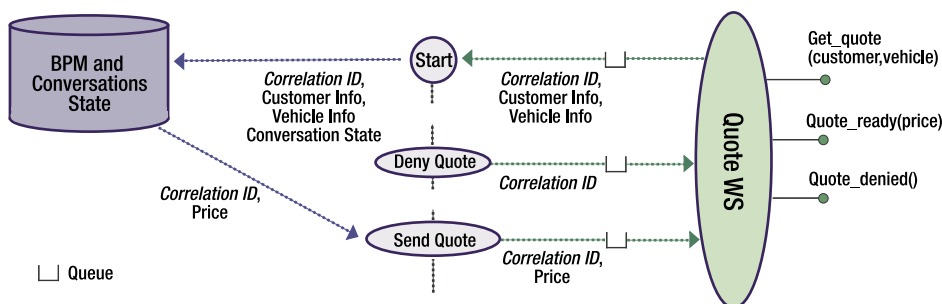


FIGURE 3 | Public Web service and BPM interaction (information flow and state management)

abstraction of the services offered by the Web services involved while hiding the low-level implementation details. The second requirement is to keep the business process definition

data and functions of the back-end application. The adapter interfaces are wrapped with a Web service so that the BPM has a consistent metaphor and interfaces to



**WebServices**  
JOURNAL

**XML** JOURNAL

THE FIRST & ONLY  
**WEB SERVICES**  
RESOURCE CD

# WEB SERVICES RESOURCE CD

THE SECRETS OF THE WEB SERVICES MASTERS

INCLUDES EXCLUSIVE .NET ARTICLES

MORE THAN

**400**  
EXCLUSIVE

WEB SERVICES  
& XML  
ARTICLES



EDITED BY  
SEAN RHODY

**\$119**  
CD  
**VALUE**

FROM  
WEB SERVICES  
JOURNAL

Web services **EDGE** \$100  
conference & expo coupon inside

EVERY ISSUE OF WSJ & XML-J EVER PUBLISHED

# THE MOST COMPLETE LIBRARY OF EXCLUSIVE WSJ & XML-J ARTICLES ON ONE CD!

## "The Secrets of the Web Services Masters"

CD is edited by well-known WSJ Editor-in-Chief  
Sean Rhody and organized into more than 40 chapters  
containing more than 400 exclusive WSJ & XML-J articles.

**Easy-to-navigate HTML format!**

### Bonus:

Full .PDF versions of every WSJ & XML-J published  
since the first issue

XML in Transit  
XML B2B  
Java & XML  
The XML Files  
XML & WML  
Voice XML  
SYS-CON Radio  
XML & XSLT  
XML & XSL  
XML & XHTML  
2B or Not 2B  
XML Script

XML Industry  
Insider  
<e-BizML>  
XML & Business  
XML Demystified  
XML &  
E-Commerce  
XML Middleware  
XML Modeling  
CORBA & XML  
Data Transition  
XML @ Work

XML &  
Databases  
Electronic Data  
Interchange  
Ubiquitous  
Computing  
Information  
Management  
Objects & XML  
XML Pros & Cons  
Java Servlets  
XML Filter

UML  
Integration  
WSDL  
Beginning  
Web Services  
Web Services  
Tips & Techniques  
Frameworks  
Management  
Security  
UDDI  
.NET

3  
YEARS  
25  
ISSUES  
400  
ARTICLES  
ONE CD



Special Limited Time Price

Now  
Shipping

\$79

+ S&H

**ONLINE**  
ORDER AT  
**JDJSTORE.COM**  
**SAVE**  
**\$40**

[WWW.JDJSTORE.com](http://WWW.JDJSTORE.com)

OFFER EXPIRES JUNE 30, 2002



# Concurrently Accessing Multiple Web Service Instances

## Web service toolkits make short work of large searches

There are a number of application areas in which it's useful to access multiple Web services that implement the same interface. For example, consider buying a book from an online bookstore. Suppose several bookstores implement a Web service that provides information about the price and availability of books. A user can employ a software agent that contacts the Web services of these bookstores to find the one with the lowest price for a particular book. Another example is a user who wants to buy a digital camera with a particular set of features such as 2 megapixel resolution and a battery life of two hours. The user's software agent can contact the Web services of several online vendors to find those that sell cameras with the desired features. Yet another example is a user who is checking online job listings. The user's agent can contact the Web services of several online job posting sites and combine the results to form a single comprehensive list of job postings.



The situations in which it can be beneficial for a software agent to access multiple Web service instances can be categorized as follows:

- **Choosing a vendor with the best value for some feature:** Examples include: the bookstore with the lowest price for a book, a car dealer with a car that gets the most miles per gallon, the vendor that sells sleeping bags with the lowest temperature rating.
- **Finding vendors that carry items with particular features:** Examples include: vendors selling digital cameras with 2 megapixel resolution and a battery life of two hours, an airline with flights from New York to San Diego on March 21.
- **Combining information from several sources:** Examples include: searching several online job posting sites to form an aggregate listing of jobs, getting book reviews from multiple sites, gathering

news about a company from several sources.

- **Achieving fault tolerance:** Examples include: getting stock quotes from an alternate service if the primary service is down, verifying credit card data from a backup service.
- **Dividing up work:** Examples include: a chess program that employs multiple service instances to examine different portions of its search tree, a genetic algorithm that distributes computation among multiple service instances.
- **Improving response time by concurrently calling several service instances and using whichever returns a result first:** An example is getting real-time stock quotes by sending requests to several services.

One approach to handling these situations is to access Web services sequentially. For example, when looking for a bookstore with the best price an agent could first contact a service at Fatbrain.com. After it has received a price, it would then request a price from Amazon.com, and so on.

This sequential approach results in unnecessarily long delays for the end user. A better approach is to send requests concurrently to all the bookstores. This article develops a reusable Java component designed for this task.

The approach taken here makes use of the IBM Web Services Toolkit (IBM WSTK), although the same concepts apply to other Web service toolkits such as GLUE and Idoox. The IBM WSTK provides a utility to convert a WSDL interface for a Web service into a Java proxy class. The proxy class acts as a client to the Web service and converts Java method calls to SOAP calls. The class has methods corresponding to the Web service methods described in the WSDL. A user of the class simply calls the appropriate method and waits for it to return a value.

The proxy class generated by the IBM WSTK is intended for use with a single Web service instance. When a method on the class is called, it blocks the client's thread until the Web service returns a result. This article presents a class called MultiServiceProxy that makes use of the IBM WSTK proxy classes internally. The MultiService Proxy class provides its clients with a straightforward way to concurrently access mul-

**Author Bio**  
Joe Verzulli is an independent consultant in the New York area specializing in Java and XML. He has worked with Java since JDK 1.0.2.  
JVERZULLI@HOTMAIL.COM

TABLE 1: Primary classes and interfaces

Name	Type	Responsibility
MultiServiceProxy	Class	Allows application programs to concurrently call methods on multiple Web service instances.
IBatchServiceListener	Interface	Implemented by application programs to receive a single notification after all Web service instances have returned results.
IncrementalServiceListener	Interface	Implemented by application programs to receive a notification each time another Web service instance returns a result.
ServiceResultEvent	Class	Encapsulates a result from a Web service along with any error information.
Bookstore_ServiceProxy	Class	Generated by IBM WSTK to call methods on a single instance of the bookstore Web service.
IBookstore Interface	Interface	exposed by the bookstore Web service.
MethodCallContext	Class	Accumulates results returned from multiple Web service instances.
ServiceAccessThread	Class	Each instance of this class runs in a separate thread and uses a Bookstore_ServiceProxy instance to make a blocking call to a Web service.

multiple Web service instances. MultiServiceProxy doesn't block its client's thread while waiting for services to return results. Instead, it allows the client to register event listeners that will be invoked when the Web services return results.

### Using the MultiServiceProxy Class

The MultiServiceProxy class allows clients to concurrently call multiple Web service instances using the same methods they use to call a single Web service instance. Listing 1 shows an example.

The key statement in this example is the call to MultiServiceProxy.newInstance().newInstance() returns an object that allows clients to call multiple Web service instances concurrently. The statement multiBookstore.getPrice("OOSC2") causes two threads to be spawned. One thread calls getPrice() on the Web service at <http://host1.com:8080/soap/servlet/rpcrouter> and the other thread calls getPrice() on the Web service at <http://host2.com:8080/soap/servlet/rpcrouter>. multiBookstore.getPrice("OOSC2") returns to the client immediately without waiting for the threads to finish their tasks. At some later time the threads will receive results from the Web service instances. At that time the threads will pass the results back to the client by calling event listeners that the client registered with MultiServiceProxy.

The Bookstore\_ServiceProxy class passed to newInstance() is a proxy class generated by

the IBM WSTK for accessing the bookstore Web service. It contains a method, getPrice(), that requests a price from a single Web service instance and doesn't return until a reply is received from the service. The MultiServiceProxy class creates one instance of Bookstore\_ServiceProxy for each Web service it contacts. These instances are created in separate threads so that the multiple Web service instances can be contacted concurrently.

As mentioned above, the MultiServiceProxy class passes results back to the clients via event listeners. Clients can implement the IncrementalServiceListener interface to be notified when Web service results arrive. The code fragments in Listing 2 illustrate this.

In this code fragment the statement multiBookstore.getPrice("OOSC2") will cause asynchronous calls to be made to multiple Web service instances. In the typical case multiBookstore.getPrice("OOSC2") will return before the Web services have returned values and the message "Back from multiBookstore.getPrice()" will be displayed. At some later time one of the Web services will return a value. That will cause SampleIncrementalListener.result() to be called. Some time after this, another Web service will return a value and SampleIncrementalListener.result() will be called again.

The ServiceResultEvent class contains the result from the Web service and the URL identifying the Web service that the result came from. It also contains error

information that can be queried to see if there was an error while contacting the service.

The above example uses the IncrementalServiceListener interface to receive separate result notifications for each Web service. There is also an IBatchServiceListener interface to receive notifications of Web service results. IBatchServiceListener contains a results() method that is invoked just once after all Web service instances have returned results. The results are passed as a Collection of ServiceResultEvent instances to the results() method. Table 1 summarizes the primary classes and interfaces.

### Internals of the MultiServiceProxy Class

The MultiServiceProxy class uses Java's dynamic proxies to present clients with the same interface as the proxy classes generated by the IBM WSTK. (Note that the proxy classes generated by the IBM WSTK are proxies in the traditional sense of the word and have nothing to do with Java's dynamic proxies.) The classes generated by the IBM WSTK work with a single Web service instance at a time while the dynamic proxies created by the MultiServiceProxy class work with multiple Web service instances concurrently.

Listing 3 shows the newInstance() method of MultiServiceProxy. It calls Java's Proxy.newInstance() method to create a dynamic proxy class implementing the same interface as the class from the IBM WSTK (which is passed in as the WebServiceProxyClass argument). The last argument to newInstance() is a MultiServiceProxy instance whose invoke() method will be called whenever one of the methods of the dynamic proxy class is called.

An earlier example contained the statement multiBookstore.getPrice("OOSC2") where multiBookstore is a dynamic proxy returned by newInstance(). We never explicitly wrote a client-side class with a getPrice() method. The Bookstore\_ServiceProxy class generated by the IBM WSTK has a getPrice() method but multiBookstore is not an instance of Bookstore\_ServiceProxy so multiBookstore.getPrice() must be calling some other method.

multiBookstore is an instance of a dynamic proxy class created behind the scenes by Java when Proxy.newInstance() was called. After creating the class, Proxy.newInstance() then created



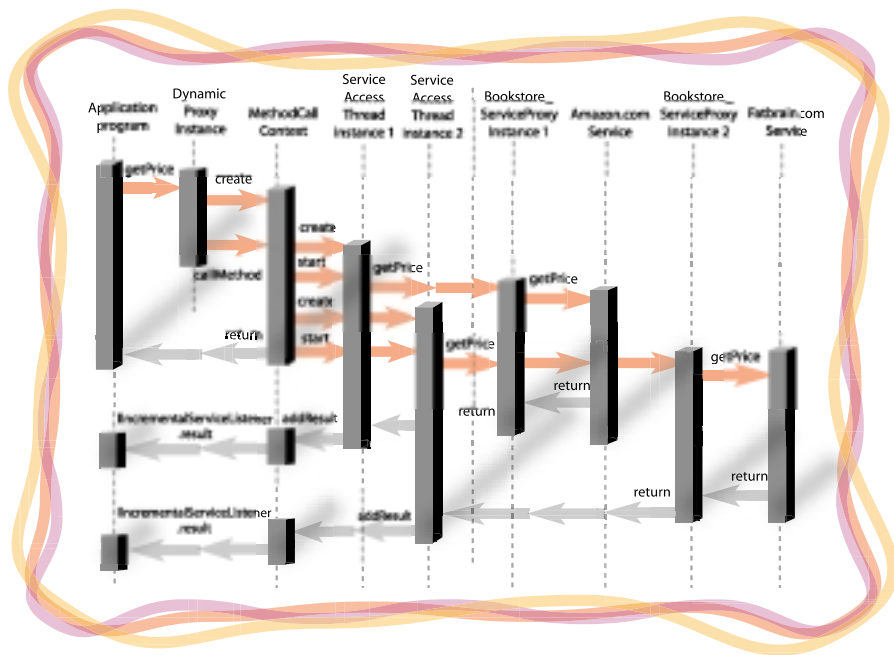


FIGURE 1 | Sequence diagram showing Web services instances

an instance of the class and returned it. The dynamic proxy class implements the same interface as `Bookstore_ServiceProxy` because `Bookstore_ServiceProxy` was passed as an argument to `Proxy.newProxyInstance()`.

The dynamic proxy class's `getPrice()` method doesn't do any real work itself; it just delegates to `MultiServiceProxy.invoke()`. `MultiServiceProxy.invoke()` can tell that the original method call was to `getPrice()` by examining its `Method` argument.

## The Invocation Handler

The invocation handler `MultiServiceProxy.invoke()` creates an instance of the `MethodCall Context` class. `MethodCall Context` stores the parameters and results for a

particular Web service method invocation. `MethodCall Context.call Method()` spawns a thread (via a `Service AccessThread` class) for each Web service to be called. Each thread creates an instance of `Bookstore_ServiceProxy` and makes a blocking call to `Bookstore_ServiceProxy.getPrice()`. When `Bookstore_ServiceProxy.getPrice()` returns, the thread adds the return value to an internal collection in the `MethodCall Context` instance. When a value is added to the collection a check is made to see if we have received results from all Web service instances. If so, the results will be sent to all registered `IBatchServiceListeners`. In addition the current result will be sent to all registered `IncrementalServiceListeners` regardless of whether we have received results from all services yet.

Figure 1 is a sequence diagram showing an example of calling `getPrice()` with two Web service instances.

## Sample Client and Web Service

This section describes a sample client that contacts multiple instances of a Web service. We will use the bookstore Web service mentioned earlier. A GUI client will use the `MultiServiceProxy` class to call the `getPrice()` method of several instances of the service. Figure 2

shows the GUI client. It allows the URLs of the Web services to be entered. There are radio buttons for selecting either an `IncrementalServiceListener` or an `IBatchServiceListener`.

Clicking the "Call Web services" button uses the `MultiServiceProxy` class to call `getPrice()` on the Web services whose URLs were entered in the listbox. If the `IncrementalServiceListener` radio button was selected then prices returned by services will be added to the table grid as they arrive. The sample bookstore Web service has a random delay in its `getPrice()` method to simulate the delay that would be experienced calling a real service across the Internet. This random delay causes values to be added to the results table one at a time with a noticeable delay between values.

If the `IBatchServiceListener` radio button is selected and the "Call Web services" button is clicked the result table will remain empty for some random time period and then all values will appear at once.

## Creating the Proxy

In this section we will briefly describe the high-level steps used to create the Web service proxy class `Bookstore_ServiceProxy`. We also mention a minor change that needs to be made to the `Bookstore_ServiceProxy.java` file after it is generated.

The first step is to write the class `Bookstore`, which implements the bookstore Web service. Then the `wsdlgen` utility from the IBM WSTK is used to create WSDL files describing the service. At this point we have the files needed for the server side of the Web service.

The next step is to run the `proxgen` utility from the IBM WSTK on the WSDL files. `Proxgen` creates the file `Bookstore_ServiceProxy.java`.

The `MultiServiceProxy` class (and the dynamic proxies it employs) requires that `Bookstore_ServiceProxy` implement an interface containing the `getPrice()` method. Since the IBM WSTK doesn't generate such an interface, we manually create an `IBookstore` interface with the `getPrice()` method. The `Bookstore_ServiceProxy.java` file must then be edited to add "implements `IBookstore`" to the declaration of the class.

## Conclusion

This article has developed a Java component that allows application programmers to easily call methods on multiple

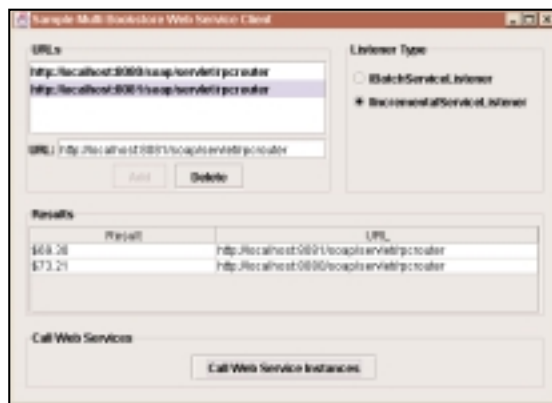


FIGURE 2 | GUI Client

Web service instances concurrently. This component can be used in a variety of applications such as finding the best price from several vendors or merging information from several sources.

The MultiServiceProxy component builds on the proxy classes generated by the IBM WSTK. This reduces the complexity of MultiServiceProxy but introduces a dependency on the IBM WSTK. However, the same general approach can be used with proxy classes generated by other Web service toolkits (such as GLUE and Idoox) with minor modifications.

MultiServiceProxy allows clients to use the same interface for accessing multiple Web service instances that they use for

accessing a single instance. This is done with Java's dynamic proxies so that application programmers don't have to write custom code for each Web service. MultiServiceProxy also hides most of the threading issues involved in calling multiple Web service instances from the application programmer.

## References

- Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.
- GLUE: [www.themindelectric.com](http://www.themindelectric.com)
- IBM WSTK: IBM Web Services Toolkit: <http://alphaworks.ibm.com/tech/webser>

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

- Idoox: [www.idoox.com](http://www.idoox.com) ©



### Listing 1

```

IBookstore multiBookstore;
String urls =
    {"http://host1.com:8080/soap/servlet/rpcrouter",
     "http://host2.com:8080/soap/servlet/rpcrouter"}

multiBookstore =
    (IBookstore) MultiServiceProxy.newInstance(
        Bookstore_ServiceProxy.class,
        urls,
        listener);

multiBookstore.getPrice("OOSC2");

```

### Listing 2

```

class SampleIncrementalListener
    implements IIncrementalServiceListener
{
    public void result(ServiceResultEvent r)
    {
        System.out.println("Got result "
            + r.getValue()
            + " from Web service "
            + "at URL " + r.getURL());
    }
}

```

```

// The following code is in the client that will
// contact multiple web services.
IIncrementalServiceListener listener =
    new SampleIncrementalListener();

multiBookstore =
    (IBookstore) MultiServiceProxy.newInstance(
        Bookstore_ServiceProxy.class,
        urls,
        listener);

multiBookstore.getPrice("OOSC2");
System.out.println("Back from "
    + "multiBookstore.getPrice()");

```

### Listing 3

```

Listing 3

public static synchronized Object newInstance(
    Class webServiceProxyClass,
    String urls[],
    IServiceListener listeners[])
{
    return Proxy.newProxyInstance(
        webServiceProxyClass.getClassLoader(),
        webServiceProxyClass.getInterfaces(),
        new MultiServiceProxy(
            webServiceProxyClass,
            urls,
            listeners));
}


```

Download the code at

[sys-con.com/webservices](http://sys-con.com/webservices)

## Microsoft Launches Visual Studio .NET And .NET Framework

(San Francisco) – Microsoft Corp. has launched Visual Studio .NET and the .NET Framework, the

 application development tool and platform for .NET applications. The two products were hailed as key to the development experience that will drive the phenomenon of XML Web services. Visual Studio .NET and the .NET Framework are the cornerstones of .NET and represent a major milestone in bringing Microsoft's vision of XML Web services to reality.

With Visual Studio .NET and the .NET Framework, developers can create and deploy XML Web services on the .NET Platform. XML Web services provide businesses with a new way to employ the Internet as a development platform and enable them to seamlessly interoperate across disparate systems and platforms. With support for more than 20 programming languages, these products enable developers to extend their existing skills to embrace the new world of XML Web services. [www.microsoft.com](http://www.microsoft.com)



## World Wide Web Consortium Issues XML Signature as a W3C Recommendation

The World Wide Web Consortium (W3C) has issued XML-Signature Syntax and Processing (XML Signature) as a W3C Recommendation,

 representing cross-industry agreement on an XML-based language for digital signatures. A W3C Recommendation indicates that a specification is stable, contributes to Web interoperability, and has been reviewed by the W3C membership, who favor its widespread adoption.


Digital Signatures are essential to Web services. While there are technologies you can use to sign an XML file, XML Signature brings two additional benefits. First, XML Signature can be implemented with and uses many of the same toolkits used for XML applications. No additional software is required. Second, XML Signature can process XML as XML, instead of a single large document. This means multiple users may apply signatures to sections of XML, not simply the whole document. As more commercial applications are used to send XML

documents through a series of intermediaries, the ability to sign sections of a document without invalidating other portions is invaluable.

[www.w3.org](http://www.w3.org)

## Sun's Forte Developer 7 Simplifies Extension of C and C++ Applications Into Sun ONE Architecture

(Santa Clara, Calif.) – Sun Microsystems, has announced an early access release of its Forte


 Developer 7 suite of products, which simplifies the incorporation of C, C++, and Fortran applications into the Sun Open Network Environment (Sun ONE) platform. The Forte Developer suite represents Sun's highest performing, most tightly integrated development products for creating new functionality in the Solaris Operating Environment on both the SPARC and Intel architecture platforms.

With this release, the Forte Developer suite has been migrated to the NetBeans open source platform, which is the foundation for Sun's Forte for Java integrated development environment (IDE). Sun's family of development tools now shares a unified IDE, creating the first tool that works seamlessly for all major languages and across platforms. This approach helps improve productivity, enabling developers to work in multiple languages with one environment.

[www.sun.com/forte/developer](http://www.sun.com/forte/developer)

## ArtinSoft to Offer Automated Conversion Tool From Java/J2EE to C#

(San José, Costa Rica) – ArtinSoft, an enterprise software migration and upgrade specialist, has

 announced its new automated conversion solution, Java Language Conversion Assistant Enterprise Edition (JLCA EE), to support conversion from Java and J2EE to C#. JLCA EE is a superset of the Microsoft JLCA. Microsoft's JLCA converts applications from Visual J++ to C# and was built on ArtinSoft's Freedom technology.

ArtinSoft's JLCA EE provides customers with an investment in the J2EE platform with a safe and economical conversion to C#, one of the key programming languages used on Microsoft's .NET platform. JLCA EE extends and enhances Microsoft's JLCA by including: J2EE language features, Java Beans, Enterprise JavaBeans, SWING, XML, and RMI among others. It will enable enterprises to maximize the benefits of the .NET Framework, Visual Studio .NET, XML Web services, and XML-based applications.

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

## webMethods Sets Agenda for Enterprise Web Services

(Fairfax, VA) – webMethods, Inc., a leading provider of integration software, has defined its approach for implementing Web services.


The webMethods integration platform provides a foundation for the deployment of Enterprise Web Services – enterprise-class Web services coupled with business process management capabilities and a robust integration platform.


webMethods differentiates Enterprise Web Services by providing an ability to reuse existing applications, offering transaction management capabilities, and building on standards.

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

## Oracle Sponsors Technical Committee To Define Portlet Web Services Standards

(Redwood Shores, Calif.) – Oracle Corp. has announced its sponsorship of the OASIS Web Services for Remote Portals (WSRP) Technical Committee. The

 committee will define open Web services standards for "remote portlets" – portlets that can be written once

 and deployed on any portal application. As a result, portlets are expected to be freely accessible to end users regardless of the portal application they are using. The standardization of WSRP will not only expand and enhance the Web services content available, but help increase developer productivity by enabling them to focus on designing compelling content rather than implementing code changes.

[www.oracle.com](http://www.oracle.com), [www.oasis-open.org](http://www.oasis-open.org)



# WEB SERVICES JOURNAL

## Readers'

**Vote Now**  
to Select the Best  
Products of the Year  
in 17 Award  
Categories...

Winners to be announced at  
"Web Services Edge 2002  
International Web Services  
Conference & Expo"

Jacob Javits Center,  
New York City, New York

June 24-27

[www.sys-con.com](http://www.sys-con.com)

- ✓ Best App Server for Web Services
- ✓ Best Framework for Web Services
- ✓ Best Integrated Services Environment
- ✓ Best Mass Market Web Service
- ✓ Best Web Service Site
- ✓ Best Web Services Automation Tool
- ✓ Best Web Services Book
- ✓ Best Web Services Class Library
- ✓ Best Web Services Foundation Platform
- ✓ Best Web Services IDE
- ✓ Best Web Services Integration Tool
- ✓ Best Web Services Legacy Adapter
- ✓ Best Web Services Middleware
- ✓ Best Web Services Platform
- ✓ Best Web Services Testing Tool
- ✓ Best Web Services Training Tool or Program
- ✓ Best Web Services Utility

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



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

**WebSphere**  
DEVELOPER'S JOURNAL



## Introductory Charter Subscription

**SUBSCRIBE NOW AND SAVE \$31.00  
OFF THE ANNUAL NEWSSTAND RATE**  
ONLY \$149 FOR 1 YEAR (12 ISSUES) REGULAR RATE \$180

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

## Do You Have Access to the Internet?

## Then Subscribe Online and Save \$31!

**It's that easy**

The  
World's  
Leading  
Independent  
WebSphere  
Developer  
Resource



# And the Winner Is...

WRITTEN BY



**Anne Thomas  
Manes**

Anne Thomas Manes is the CTO of Systinet, a Web services infrastructure company. Anne is a recognized industry spokesperson and has published on a range of technology issues.  
ATM@SYSTINET.COM

I was quite amused by a series of articles talking about the battle between Java and .NET that appeared in mid-January. One article said that Java has a two-to-one lead over .NET based on an informal online poll. Meanwhile, in an article entitled "Outlook: Java tech trends through 2004," Mark Driver at Gartner claimed, "Microsoft's emerging .NET platform will continue to garner most of the vision and mind share for Web-services-based development efforts." And in an article entitled, "Enterprise Java Bulks Up," Thomas Murphy of META Group said, "The lack of standards support will not enable Java to compete as effectively with the challenge raised by Microsoft .Net."

With such drastically differing opinions out there, I thought it would be entertaining to conduct my own investigation. Since I don't have the resources to conduct a statistically significant survey, I decided to base my research on newsgroup and discussion list traffic. The way I figure it, traffic on these forums should be a good indication of the actual use of the technology.

I conducted my research in early January, and here's what I found. Developmentor hosts a number of popular discussion groups at <http://discuss.develop.com>. The most popular one is the DOTNET group, which has over 2,800 members and usually exchanges 100–200 messages a day. The traffic on this one group exceeds the traffic on all other Web services-related discussion groups combined. Based on this, one might assume that .NET is the clear winner – but only if you're comparing .NET with the entire Java platform. When I looked inside the DOTNET discussion list, I found that most of the discussions focus on the .NET framework, C#, ASP.Net, and ADO.Net. I wanted to limit my study to Web services development – and by that I actually mean SOAP development. So, using a loose statistical process (I analyzed five weeks worth of messages), I estimated that approximately 15% of the DOTNET discussion list traffic pertains to Microsoft SOAP development, which is an average of about 675 messages per month.

Another very popular forum hosted by Developmentor is the SOAP discussion group. This group has over 1,700 members, who exchange an average of 180 messages per month. This is a nondenominational

group, which happily discusses SOAP issues based on any implementation or language. You'll find discussions relating to Perl, PHP, Python, and PocketSoap, as well as .NET and Java. Using my loose statistical process, I estimated that 50% of traffic pertains to .NET, 35% pertains to Java, and 15% pertains to other languages.

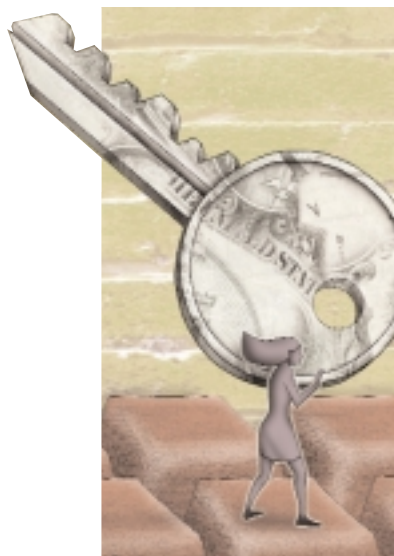
I next examined Microsoft's discussion lists at <http://msdn.microsoft.com/newsgroups/>. There appear to be three active discussion lists related to Web services: xml.soap, xml.soapSDK, and msdn.webservices. The combined traffic on these three lists adds up to approximately 990 messages per month.

Then I started looking for Java discussions. Most Java SOAP discussions are sponsored by the SOAP implementation providers forums. Based on that traffic, the three most popular Java SOAP implementations are Apache SOAP, Systinet WASP, and The Mind Electric GLUE. Apache sponsors two user discussion lists (soap-user and axis-user) with more than 1500 members. Combined, the two lists exchange an average of 600 messages per month. The archives are available at <http://marc.theaimsgroup.com>. Systinet provides a newsletter list and a developer discussion forum. There are more than 3000 subscribers to the newsletter and more than 350 members in the discussion forum. The Systinet discussion group, at <http://list.systinet.com>, exchanges an average of 550 messages each month. The Mind Electric uses Yahoo Groups to host its discussion list at <http://groups.yahoo.com/group/MindElectricTechnology>. There are more than 900 members in this group, who exchange an average of 360 messages each month.

In addition to the major players, I also found a little bit of traffic at each of the other Java-based SOAP vendor sites, including Oracle, Cape Clear, HP, Iona, IBM, BEA, and Borland (in order by volume). Oracle averages approximately 75 messages per month; the others between 30 and 45 messages per month.

I found two other popular SOAP discussions. Borland's Web services discussion list for Delphi at [www.borland.com/newsgroups](http://www.borland.com/newsgroups) gets traffic comparable to that for Apache. And Simon Fell's PocketSOAP discussion at [http://groups.yahoo.com/group/pocket\\_soap](http://groups.yahoo.com/group/pocket_soap) has more than 100 members who exchange an average of 80 messages per month.

So, let's take a moment and tally up the results. According to my calculations, Java wins by a nose with an average of 1,868 messages per month, capturing 43% of discussion traffic. .NET is a close second with 1,755 messages per month, or 41% of traffic. And "other" (Delphi, C++, PHP, Perl, Python, Frontier, etc.) rounds out the landscape with 16% of traffic, averaging 707 messages per month. In other words, interoperability is what's really important. ☺



# **Hewlett-Packard Company**

**[www.hpmiddleware.com/download](http://www.hpmiddleware.com/download)**

# **XML Global**

**[www.xmlglobal.com/newangle](http://www.xmlglobal.com/newangle)**