

WebServices

.NET J2EE XML JOURNAL

WSJ2.COM

Announcing...



March 18-20, 2003
Boston, MA

HYNES CONVENTION CENTER

SEE PAGE 39 FOR DETAILS

From the Editor

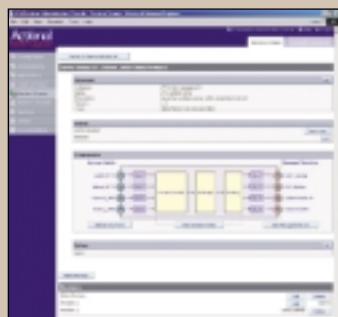
A Rose By Any Other Name

by Sean Rhody pg. 3

Guest Commentary

A Look Forward

by Arthur J. Musgrave pg. 9



First Look

Actional SOAPStation

by Joseph A. Mitchko pg. 16

Emerging Trends

Critical Decisions for Service-Oriented Architecture

by Waller Hurst pg. 36

RETAILERS PLEASE DISPLAY
UNTIL MARCH 31, 2003

\$6.99US \$7.99CAN



SYS-CON MEDIA

INTERNATIONAL .NET DEVELOPER CONFERENCE & EXPO

REGISTER BY FEB. 28 AND SAVE UP TO **\$350!**

web services conference & expo **EDGE**

WSI WEB SERVICES INTEROPERABILITY ORGANIZATION

Web Services Panel Discussion
A Road Map for Web Services Standards

March 18-20, 2003
Boston, MA
Hynes Convention Center

PAGE 47

WEB SERVICES EDGE • JDJEDGE • XMLEDGE • .NET EDGE

FOCUS ON WEB SERVICES MANAGEMENT

- **On the Road to Web Services-Level Management** *Critical components in self-managing* Mark Potts, Kevin Ruthen & Heather Kreger **26**
- **Operational Management: A Web Service Barrier-to-Entry** *The need for service-centric management tools* Franco R. Negri **38**
- **The Promise and Peril of Web Services** *Management will make the difference* Rajiv Gupta **58**
- CRM: Migrating CRM to a Standards-Based Platform** *A future based on integration and interoperability* Ken Heiman & Michael Venturelli **7**

WSJ Feature: A Publish/Subscribe Mechanism for Web Services *Extending an existing event broker* Dmitri Tcherevik **10**

UDDI: Web Services Versioning and Deprecation *An easy-to-implement strategy for success* Jeff Kenyon **18**

WSJ Feature: On-Demand Computing *Incrementally adapting to a new frontier* Bernhard Borges **32**

WSJ Feature: Reaching Back to Unlock Legacy Systems *Finding the secrets that already exist* James Philips & Brian Anderson **42**

Parasoft

www.parasoft.com/ws2

INTERNATIONAL ADVISORY BOARD

Andrew Astor, David Chappell, Graham Glass, Tyler Jewell,
Paul Lipton, Anne Thomas Manes, Norbert Mikula,
Frank Moss, George Paolini, Simon Phipps

TECHNICAL ADVISORY BOARD

Steve Benfield, Bernhard Borges, JP Morgenthal, Andy Roberts,
Ajit Sagar, Michael A. Sick, Simeon Simeonov

EDITORIAL

EDITOR-IN-CHIEF

Sean Rhody sean@sys-con.com

EDITORIAL DIRECTOR

Jeremy Geelan jeremy@sys-con.com

INDUSTRY EDITOR

Norbert Mikula norbert@sys-con.com

PRODUCT REVIEW EDITOR

Joe Mitchko joe@sys-con.com

.NET EDITOR

Dave Rader davidr@fusiontech.com

TECHNICAL EDITORS

David Chappell chappell@sonicsoftware.com

Anne Thomas Manes anne@manes.net

EXECUTIVE EDITOR

Gail Schultz gail@sys-con.com

EDITOR

Nancy Valentine nancy@sys-con.com

ASSOCIATE EDITORS

Jamie Matosow jamie@sys-con.com

Jean Cassidy jean@sys-con.com

ASSISTANT EDITOR

Jennifer Stilley jennifer@sys-con.com

PRODUCTION

PRODUCTION CONSULTANT

Jim Morgan jim@sys-con.com

LEAD DESIGNER

Richard Silverberg richards@sys-con.com

ART DIRECTOR

Alex Bolero alex@sys-con.com

ASSOCIATE ART DIRECTOR

Louis Cuffari louis@sys-con.com

ASSISTANT ART DIRECTOR

Tami Beatty tami@sys-con.com

CONTRIBUTORS TO THIS ISSUE

Brian Anderson, Bernhard Borges, Rajiv Gupta, Ken Heiman,
Walter Hurst, Jeffrey Kenyon, Heather Kreger, Joe Mitchko,
A. J. Musgrove, Franco Negri, James Phillips, Mark Potts,
Sean Rhody, Kevin Ruthen, Dmitri Tcherevik, Michael Venturelli

EDITORIAL OFFICES

SYS-CON MEDIA

135 CHESTNUT RIDGE ROAD, MONTVALE, NJ 07645

TELEPHONE: 201 802-3000 FAX: 201 782-9637

WEB SERVICES JOURNAL (ISSN# 1535-6906)

Is published monthly (12 times a year)

By SYS-CON Publications, Inc.

Periodicals postage pending

Montvale, NJ 07645 and additional mailing offices

POSTMASTER: Send address changes to:

WEB SERVICES JOURNAL, SYS-CON Publications, Inc.

135 Chestnut Ridge Road, Montvale, NJ 07645

©COPYRIGHT

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

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

A Rose By Any Other Name...

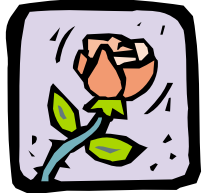
Written by
Sean Rhody



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



What's in a name? A rose by any other name will still smell as sweet. Well, perhaps in the world of horticulture, but in the information technology arena, I'm not sure that aphorism applies. I'm sure you all realize that I'm referring to the recent purchase of Rational Software by IBM for approximately \$2.2 billion dollars. This acquisition leaves me wondering what Rose will smell like a year from now.

Rational as a company helped define an interesting movement and market – that of development by model. Its founders defined various modeling methodologies into UML and codified its usage. But more than just modeling, Rational was the leader in defining an independent movement towards improved quality and process. And increasingly, RUP, Rational's Unified Process, has become a standard for describing, modeling, and running projects. There are several reasons why that was successful, including the quality of the process and the close ties to the software that Rational produced to implement it. But one big reason that Rational was successful was its independence. At the end of the day, no one was suspicious that RUP aligned more with a particular company, or with a specific set of software. And while it always seemed to me that Rational was farther along on Java, it certainly paid close attention to Microsoft and most recently to the .NET platform.

But that's changed now. I'm sure that the purchase makes good financial sense for IBM, and was a benefit to Rational stockholders. From the perspective of implementers, the sense is less clear. For example, does IBM Global Services now use RUP as its project management methodology? If it does, what does that mean for clients who don't want to use it? If it doesn't, what kind of message does that send as a company who won't use its own products? Either way, there's some ambiguity there. Most consulting companies have their own process methodologies. I'm sure we'll hear that IBM's methodology was in line with RUP, or that RUP is complimentary or some such spin, but the bottom line is Rational is going from a company that believed its religion to a company that doesn't always eat its own dogfood. And the good will of the community that drove an independent standard (UML) may not extend to a company that will compete in hardware, software, and services.

Similarly, on the software side it seems a mixed bag. What happens if IBM decides that support for BEA's WebLogic Server is a second-tier concern? Or that an emphasis on Java, in which it has a significant market share and presence, is more important than keeping close synchronization with Microsoft's .NET platform? There's a very real possibility that this useful toolset could become marginalized and proprietyzed out of the mainstream use it receives today.

Of course, Rational is more than just one product, and some of the products are truly agnostic when it comes to technology, so you can't just simplify it and say one thing or another, which also confuses the issue. But to most folks, Rational means modeling, and the question of how agnostic it can be now has to be answered.

What makes this even more interesting is the recent purchase of Rational's main modeling rival, TogetherSoft, by Borland. Over the past few years Borland has dusted itself off, put together a solid marketing program, and won developers with its feature set and concentration on essentials for developers. And it does stand alone as a company that works with, but isn't tied to, either .NET or J2EE. But it's still a software company, albeit now one with a modeling tool, and not a modeling company. It worries me that there are no companies left who are concentrating on modeling and methodology as their sole concerns. Perhaps it's time to turn the methodology question, and the UML, over to an independent standards body. Problem is, methodology is more of a religion than a science. It takes understanding, evangelism, and buy-in in order to be successful. Legislating a standard won't do it – we need the evangelists, but they've been co-opted.

Let me know if the Rose starts to smell a little different. ☺

Mind Reef

www.mindreef.com

Mind Reef

www.mindreef.com

Sitraka

(now part of Quest Software)

www.sitraka.com/jclass/ws

Migrating CRM to a Standards-Based Platform

A future based on integration and interoperability

The next technology evolution – or revolution, depending on how you look at it – is here. Battle lines are being drawn between supporters of Microsoft's .NET and Sun's J2EE platforms for creating Web services. The common ground in this battle: the market will head toward Web services.

Web services provides a standard method through which components can communicate and interact. Similarly to how ODBC/JDBC, COM, CORBA, etc., allow interapplication communication, Web services also allow applications to communicate and interact with each other. The difference between Web services and the previous specifications is that there is a set standard to which all platforms adhere. Communications via the SOAP (Simple Object Application Protocol) and XML standards allow applications to consume Web services regardless of their own development platform. Component interaction is no longer limited to an organization's network; it now encompasses the entire Internet.

The CRM application space has long been an area where universal standards have been seen as a sort of "holy grail" of enterprise application development. The challenge of CRM is to provide a single,

enterprise-level view of customers and their interactions. With the advent of Web services, and the varying platforms that can take advantage of them, CRM applications are presented with a unique opportunity to serve as the true information hub for the enterprise.

Gone are the days of the complex, custom solutions that were previously necessary to integrate systems. As the Web services architecture enters the mainstream, CRM systems must evolve to take advantage of the integration benefits provided. With Web services, this task will be greatly simplified. Rather than creating static, inflexible synchronization scripts, or asynchronous batch processes, Web services-based solutions can simply and cleanly access and affect each other synchronously. As CRM systems are able to make direct calls to content management, ERP, or

“
Gone are the days of the complex, custom solutions that were previously necessary to integrate systems”

inventory management systems, integration will take on a more API-style development approach. This will allow CRM systems to more easily display real-time data, which is critical in managing customer relationships.

For example, imagine a series of price lists stored in an ERP application that calculate the price of a product based on sales region, quantity purchased, applicable discounts, etc. If the CRM application needs to calculate the current price for a product, without directly accessing the ERP application, the entire logic for this functionality, as well as all the data involved, must be built and stored locally in the CRM system. However, if the ERP application provides a Web service that calculates a price by passing in a certain set of criteria, all the CRM application needs to do is make a single call to the native application. Customer lists, service calls, products owned, and various other data can all be pulled together from their original sources and displayed from within a common CRM portal.

Publishing Web services outside of the enterprise will allow partners and even customers to consume the information/functionality and use it on their extranets/intranets. Partners can access product lists and real-time pricing changes on their Web sites by consuming Web services created as part of a CRM implementation. Customers can view open orders on their intranet to better track their progress.

This ease of interoperability will also enable more loosely coupled CRM systems. CRM vendors will now be able to focus on their core product rather than creating com-



Author Bios

Ken Heiman is a senior associate with Green Beacon Solutions. He has worked as a technical architect and developer in the Microsoft and Java platforms since 1995, and is an experienced CRM consultant. Ken specializes in Web, COM, and .NET CRM and Telephony/SIP solutions. [KHEIMAN@GREENBEACON.COM](mailto:kheiman@greenbeacon.com)

Michael Venturelli is a senior associate with Green Beacon Solutions. He has worked as a technical architect and developer on CRM implementations in the financial services and software industries. Michael specializes in VB and SQL Server development. [MVENTURELLI@GREENBEACON.COM](mailto:mventurelli@greenbeacon.com)

“

New technologies can improve upon existing ones, but you must also be aware of the implementation challenges ”

monly available functionality such as document management, messaging, or workflow. This is made possible by relying on partnerships with ISVs whose core expertise is in developing these components. Customers benefit because they can purchase the key components of the CRM suite. For example, they can purchase the core CRM functions from the vendor and easily integrate added functionality from existing applications or third-party vendors to address the needs of their users. This allows organizations to save in licensing and customization costs for unused or incompatible functionality, and the ability to provide a more tailored, functional system that better meets their users' needs.

New technologies can improve upon existing ones, but you must also be aware of the implementation challenges. Integration design becomes even more complex as additional internal and external groups (partners and customers) are included. Each new interaction between systems must be understood and mapped between the data context of the participating systems. In addition to the same basic data mapping issues that EDI processes have faced for years, one must now deal with providing external functionality as well as data. The benefit of this extra effort is a more complete and usable system.

Another challenge to consider is the impact on security. Any Web services exposed, whether to customers, partners,

or the general public, must be designed with these new security issues in mind. How will data be encrypted? How will access be controlled? These are not new issues, but the audience that must now be considered for these exposed Web services is no longer a controlled internal group, but could potentially include anyone on the Internet.

Technology improvements without a clear business benefit are a difficult sell for IT departments in today's economy. Organizations have a responsibility to define ROI that can justify all costs associated with re-engineering the enterprise application infrastructure. This includes hardware and software infrastructure costs, as well as developer training and costs associated with updating existing interfaces between systems. If the key elements of system interoperability already exist in the enterprise, it can become very difficult to justify such a major change (e.g., "If it ain't broke, don't fix it.").

With the advent of Web services, the competitive landscape for CRM vendors changes significantly. The possibility of loosely coupled systems will force vendors to focus on a core set of CRM capabilities. Modular CRM packaging will lead to more partnerships as ISVs look to one another to provide more complete applications. CRM vendors may replace their own version of fulfillment or collaboration with a partner's version that offers a more robust solution.

The future of standards-based Web services will be focused on integration and interoperability. With the adoption of standards-based platforms that can provide Web services, CRM can give enterprise users a more complete picture of their customers. Collaboration among customers, partners, and vendors can provide all parties with a more functional and efficient system. The future of enterprise applications is Web services, and as CRM follows, it can provide a central point in the enterprise to bring information together in real-time to better manage customers. ©

WebServices JOURNAL

PRESIDENT AND CEO

Fuat A. Kircaali fuat@sys-con.com

COO/CFO

Mark Harabedian mark@sys-con.com

VP, BUSINESS DEVELOPMENT

Grisha Davida grisha@sys-con.com

ADVERTISING

SENIOR VP, SALES & MARKETING

Carmen Gonzalez carmen@sys-con.com

VP, SALES & MARKETING

Miles Silverman miles@sys-con.com

ADVERTISING DIRECTOR

Robyn Forma robyn@sys-con.com

ADVERTISING ACCOUNT MANAGER

Megan Ring-Mussa megan@sys-con.com

ASSOCIATE SALES MANAGERS

Carrie Gebert carrie@sys-con.com

Alisa Catalano alisa@sys-con.com

Kristin Kuhnle kristin@sys-con.com

SYS-CON EVENTS

PRESIDENT, SYS-CON EVENTS

Grisha Davida grisha@sys-con.com

REGIONAL SALES MANAGERS, EXHIBITS

Michael Pesick michael@sys-con.com

Richard Anderson richard@sys-con.com

CONFERENCE MANAGER

Michael Lynch mike@sys-con.com

CUSTOMER RELATIONS/JDJ STORE

CUSTOMER SERVICE REPRESENTATIVE

Margie Downs margie@sys-con.com

JDJ STORE MANAGER

Rachel McGouran rachel@sys-con.com

SYS-CON.COM

VP, INFORMATION SYSTEMS

Robert Diamond robert@sys-con.com

WEB DESIGNERS

Stephen Klimmurray stephen@sys-con.com

Christopher Croce chris@sys-con.com

ONLINE EDITOR

Lin Goetz lin@sys-con.com

ACCOUNTING

FINANCIAL ANALYST

Joan LaRose joan@sys-con.com

ACCOUNTS RECEIVABLE

Kerri Von Achen kerri@sys-con.com

ACCOUNTS PAYABLE

Betty White betty@sys-con.com

SUBSCRIPTIONS

SUBSCRIBE@SYS-CON.COM

1-888-303-5282

FOR SUBSCRIPTIONS AND REQUESTS FOR BULK ORDERS, PLEASE SEND YOUR LETTERS TO SUBSCRIPTION DEPARTMENT

COVER PRICE: \$6.99/ISSUE

DOMESTIC: \$69.99/YR (12 ISSUES)

CANADA/MEXICO: \$89.99/YR

ALL OTHER COUNTRIES: \$99.99/YR

(U.S. BANKS OR MONEY ORDERS)





Arthur J. Musgrove

*Arthur J. Musgrove is the telecom programs director at IONA Technologies.
AJ.MUSGROVE@IONA.COM*

A Look Forward

Web services is useful, but it is too complex to be implemented in many organizations. That's where [insert new technology here] comes in. Its simplification of systems integration will revolutionize the enterprise IT department.

Sorry, I'm just getting a head start on the next wave of technology.

The truth is that hard problems have hard solutions, and industry veterans agree that systems integration is a hard problem. While IT managers want integration solutions, they sometimes think the medicine is worse than the disease. When that view becomes pervasive enough, the industry invents a new technology.

You don't believe me? When the first official version of the Common Object Request Broker Architecture (CORBA) was released in October 1991, it was said to herald a new age of simplified software interoperability and platform independence. The fact was the first CORBA products shipping in the early 1990s were unwieldy, buggy, and unstable beasts. CORBA products were missing more required features than they had, but the industry marched ahead with adoption. After several years, CORBA was useful and the specifications were complete. It had enterprise features, including distributed transactions, security, and publish and subscribe messaging. There was widespread support and most enterprise software that wasn't built on CORBA provided CORBA adapters.

The complaints started softly, then became louder, that CORBA is too hard. It's true that CORBA has so many features that the learning curve is painful. CORBA is a powerful technology, but it is so complex that skilled developers are expensive. CORBA continues its adoption trend in complex, mission-critical applications, but it is out of fashion in mainstream IT, largely in favor of Java 2, Enterprise Edition (J2EE).

J2EE application servers burst onto the scene in 1999 as an immature and often unreliable technology. A key promise of J2EE, according to Sun's J2EE Overview (<http://java.sun.com/j2ee/overview.html>), is "Making Middleware Easier." Easier middleware means lots of low-cost programmers to crank out enterprise applications, instead of the higher-priced programmers required for CORBA.

The original J2EE application servers were missing so many features that an entire industry of J2EE server add-ons and component shops was born. One example is Flashline.com, which based much of its business model on selling J2EE components and establishing an IT market for contracting the development of bespoke functionality. Over time, however, the J2EE specifications have improved. Today, application servers are fairly complete and even useful. As the specifications have become bigger to allow for this additional functionality, development with J2EE application servers is increasingly viewed as too hard.

In the Java press these days, hardly a month goes by without an author declaring J2EE too complex. The author points out that skilled J2EE programmers are expensive because of the complexity of the technology they work in. The author usually fails to mention what portions of the specification they would strike out in the name of simplification. The reason is that features wouldn't be in the specification if they weren't useful to someone. If only there were a simpler middleware technology.

Enter Web services. To be sure, there are some truly unique and useful things about this technology. Most industry observers agree that near universal support by giants including Microsoft, IBM, and Oracle is a key feature. That sort of support, as long as the technology doesn't fracture, has the potential to ease the integration pain of enterprise IT shops.

The adoption pattern of Web services is reminiscent of the Structured English Query Language (SEQUEL), later shortened to Structured Query Language (SQL). Today SQL is ubiquitous, but it hasn't always been that way. Dr. E.F. Codd at IBM is credited as the father of relational databases and SQL for his pioneering work in the early 1970s. Not everyone followed Dr. Codd and IBM exactly by using their original query language. SQL did have its competitors.

Relational Technology's Ingres RDBMS, a contemporary competitor of Relational Software's Oracle and IBM's System/R, in the 1970s and 1980s, used a language called QUEL. The SQL language was, however, so successful that Ingres was forced to adopt it in 1986. SQL was given a big boost toward standardization by the American National Standards Institute (ANSI) and the International Standards Organization (ISO).

SQL has evolved through several versions and is today the universal language of relational databases. Some people believe that, if allowed to grow and mature, Web services may become the universal language of application integration, forever ridding the world of such things as Protocol Bridges and Enterprise Application Integration (EAI) Adapters. CORBA and J2EE will continue their roles as application execution environments, but they may lose favor as integration tools. Web services is already following the path of SQL by being accepted as a standard by a recognized industry body, the World Wide Web Consortium (W3C).

There is still quite a lot of work before Web services is universal. It needs a single security standard. It needs a standard for distributed transactions. It needs an enterprise-class messaging system. It desperately needs complete specifications for mapping Web services requests into native execution environments. As these features are added, the body of specifications that is Web services will become bigger and more complex. It will also become useful.

When that happens, let's not kill it. When Web services is complete and useful enough to be derided as too complex – or better still, as dead – you will know it is ready to be the solution for your truly hard problems. ☺

A Publish/Subscribe Mechanism for Web Services

Extending an existing event broker

Software platforms traditionally offered a publish/subscribe mechanism as one of the core platform services. With help from this mechanism, an application could raise events or express interest in events produced by other applications. The Internet and Web services are emerging as the next generation platform for distributed applications. While the new platform enables applications to communicate synchronously or asynchronously over standard Internet protocols, it is still lacking a service that would resemble a traditional publish/subscribe mechanism. In this article, I'll show how such a service can be developed by extending the functionality of an existing framework.

Background

There are many types of systems that could make use of a publish/subscribe service. Figure 1 presents one of them. This system consists of a number of agents and a

few management applications. The agents monitor their environment and send status messages to the management applications. The applications process information collected through agents and make system management decisions either automatically or with help from a human operator. The decisions are mapped to actions performed by the agents.

Let's take a minute to examine the types of communication taking place between the agents and the management applications. First, there is a stream of system status messages flowing from the agents to the management applications. Second, there is a stream of control messages flowing back from the management applications to the agents.

The case of control messages is relatively straightforward. A control message has a well-

defined source and a well-defined destination. Therefore, it can be delivered easily using a traditional, one-to-one request/reply protocol. All of the available RPC mechanisms, such as IIOP, Java RMI, COM, or SOAP-RPC, satisfy this requirement.

The case of status messages is much more interesting. Messages sent by agents are caused by changes in the environment and are therefore asynchronous by nature. In addition, an agent sending a message is decoupled from the management application and rarely knows who will be receiving the message on the other end. Finally, there can be many management applications receiving messages from a single agent. Therefore, what we have here is an asynchronous one-to-many type of messaging that cannot be carried over a traditional

AUTHOR BIO:



Dmitri Tcherevik is a technology strategist in the Office of the CTO at Computer Associates (CA). After extensive development experience in Ingres and the Jasmine object database, he led the development of CleverPath Enterprise Content Manager and Advantage Integration Server. Before joining CA, Dmitri developed a computer chess program under the leadership of Mikhail Botvinnik, the renowned world chess champion. He graduated with honors from the Department of Cybernetics at the Moscow Institute of Physics Engineering (MIFI) specializing in distributed systems, databases, and logic programming.
DMITRI.TCHEREVIK@CA.COM

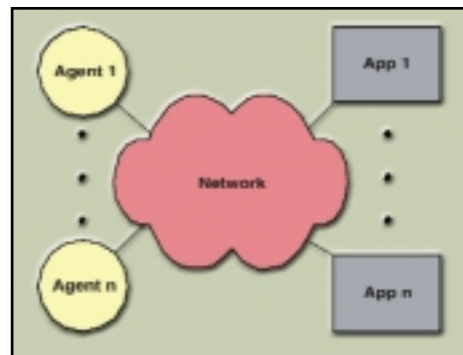
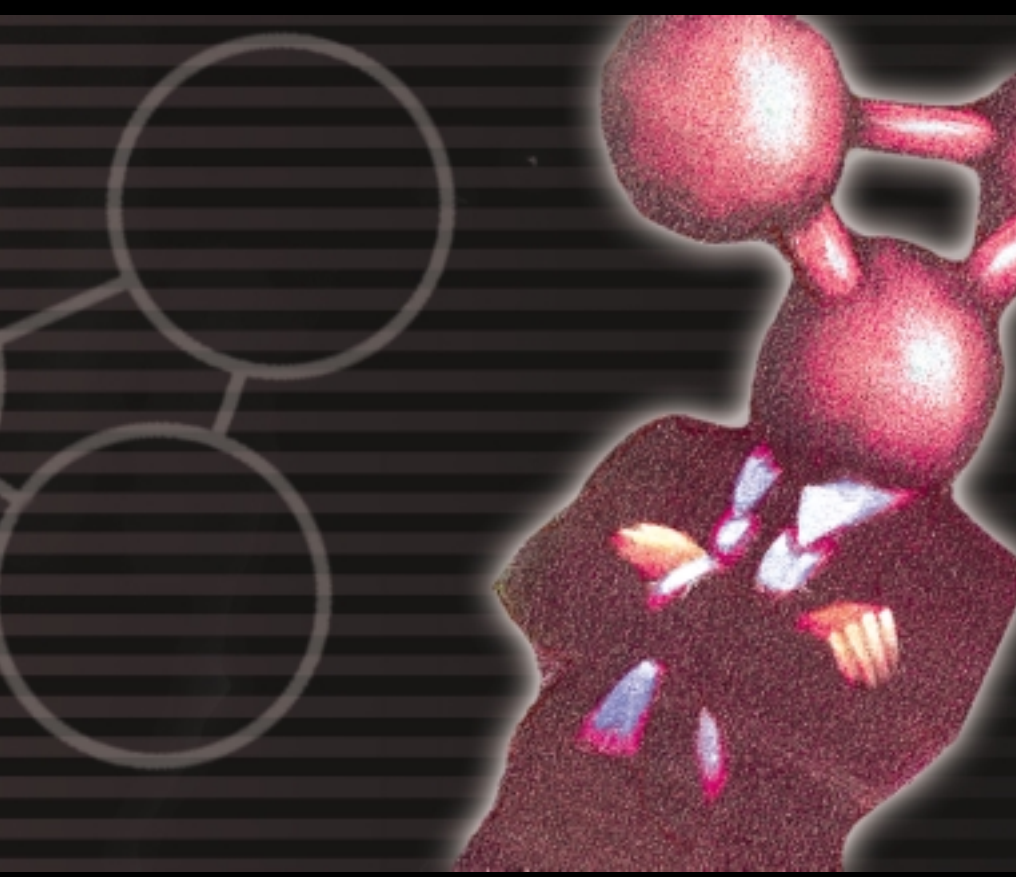


FIGURE 1 | A sample publish/subscribe scenario



RPC link. Instead, a publish/subscribe mechanism is required.

This problem has been successfully solved in many products and middleware frameworks. Typically, it is addressed by introducing a stand-alone publish/subscribe service, sometimes called an event broker or an event intermediary. From the point of view of the event broker, the world is divided into two types of entities: event producers and event consumers. Event producers advertise event types and raise events by sending messages to the event broker. Event consumers express interest in events by registering event subscriptions with the event broker. The event broker matches the two parties by forwarding events sent by the producers to endpoints registered by the consumers. In the context

of the management application described earlier, this can be seen in Figure 2.

In addition to forwarding events, the event broker can implement a slew of useful functions such as event filtering, event logging, guaranteed event delivery, event correlation, protocol translation, and others. In this article, however, I'll focus not on the functionality of the event broker but on the communication mechanism used by consumers and producers to access its services. In particular, we want to understand what it would take to deploy an existing event broker on the Internet as a Web service that could be easily accessed by other Web services and applications.

An Event Broker Overview

Let's begin with an overview of a hypo-

thetical event broker that offers a basic set of services such as event advertising, event filtering, and event forwarding.

Advertising Events

The event broker features a catalog repository that contains metadata describing events exposed by the various event producers. A management agent, a relational database, and a file system folder are all examples of event producers. Metadata stored in the catalog is structured as a collection of namespaces. A catalog namespace contains a schema and a collection of other namespaces. The schema of a namespace consists of classes that describe events available from event producers that chose to advertise their events in that namespace.

A fragment of the catalog namespace tree is shown in Figure 3. The catalog namespace that is expanded corresponds to a file system event producer. The schema of this namespace contains three classes, the names of which are self-explanatory. Each class may have several properties describing the corresponding events, e.g. the file system path of the file, its MIME type, and others.

Subscribing for Events

Given the path of a namespace and the name of an event class, an application can subscribe for events described by this class. First the application must retrieve a handle to the namespace from the catalog. Second, it must create an event filter. An event filter is a query defined in terms of properties of the event class. Third, it must register the event filter and an object that implements an event listener interface with the event broker (see Listing 1).

The listener object implements the `EventListener` interface defined as follows:

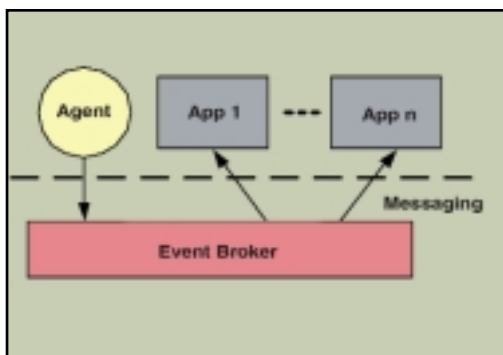


FIGURE 2 | Unicenter publish/subscribe framework

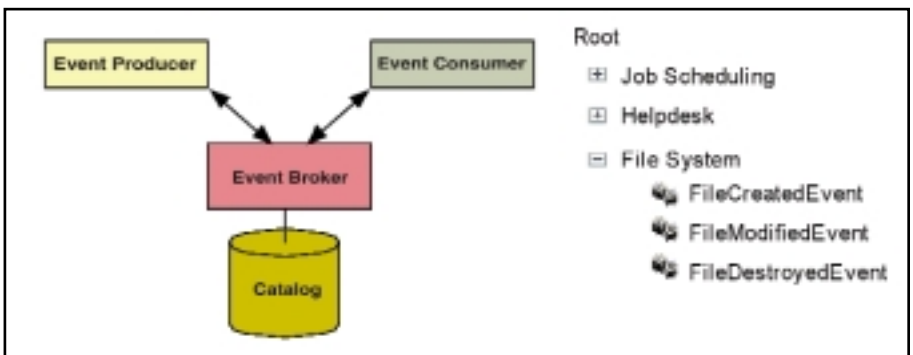


FIGURE 3 | Event Broker Catalog

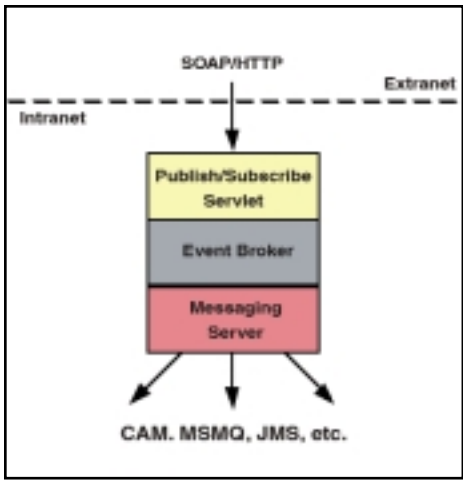


FIGURE 4 | Intranet/extranet event gateway.

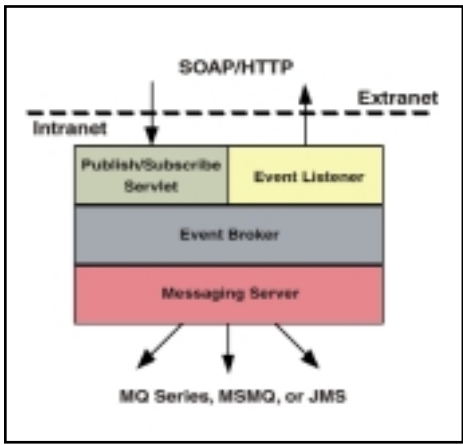


FIGURE 5 | Publish/subscribe service without subscription management

```
public interface IEventListener
{
    public void handleEvent(IEvent
event);
}
```

When an event matching the filter is detected by the event broker, the `handleEvent()` method of the listener object is invoked. It receives the event object as an argument. The event object contains properties describing the event.

Sending Events

An application sending an event must first create an event object and then use the `raise()` method of the event broker API to broadcast this event to event consumers (see Listing 2).

I make no assumptions here about the mechanism used by event consumers and event producers to access services of the

event broker. In this hypothetical example, they use these services by invoking methods of interfaces that are implemented by local or remote Java objects. In the next section we show how these interfaces can be mapped to a publish/subscribe Web service that can be easily invoked from a program written in essentially any programming language and running on practically any platform.

Publish/Subscribe Web Service

Two Styles of SOAP Encoding

There are two ways to encode information in the body of a SOAP message – either as an RPC call or as an XML document. The style of encoding is controlled by a switch in the WSDL contract for the service. The body of a SOAP message sent to an operation declared as

```
<soap:operation soapAction="uri"
style="rpc">
. . . . .
</soap:operation>
```

is encoded as an RPC call while the body of a message sent to an operation declared as

```
<soap:operation soapAction="uri"
style="document">
. . . . .
</soap:operation>
```

is encoded and interpreted as an XML document. There are strict rules governing how parameters of an RPC request shall be encoded in the body of a SOAP message. Document-style encoding, on the other hand, is very nonrestrictive and allows virtually any type of XML documents. When flexibility or extensibility is desired, document-style encoding is generally preferred.

Event broker methods used to manage event subscriptions are static in nature. Therefore, we can safely map them to operations with RPC encoding. The `raise()` method, on the other hand, is very dynamic. Event objects passed to this method are defined by classes stored in the catalog repository. We cannot afford to recompile an application every time someone adds a new event class to the catalog. Therefore, we choose document-style encoding for the `raise()` operation.

Format of Event Messages

As the first step, we define the XML format for encoding events in a SOAP message (see Listing 3). In order to reduce the number of network round trips, multiple events are encoded as part of a single SOAP message. Each event is mapped to an XML element whose name corresponds to the name of the class describing the event. Different namespaces in the event broker catalog may contain event classes with identical names. Therefore, we must properly qualify names of event elements by placing them in different XML namespaces. We derive the URL of an XML namespace containing the name of an event element from the path of the catalog namespace containing the corresponding event class.

The mapping is very simple. If the path of a catalog namespace is “Root/File System,” then the URL of the corresponding XML namespace is “<http://eventcentral.com/Root/FileSystem>”. Properties of event objects are mapped to elements nested in the event element. All property values are encoded as strings. The original type of a property is indicated in the value of the “`xsi:type`” attribute.

Event Processing

We can now develop the portion of the public/subscribe service that deals with events submitted over SOAP/HTTP by external applications. It will be implemented as a servlet extending the `JAXMServlet` class defined in JAXM, one of the standard Java APIs for Web services. The `onMessage()` method of this servlet can be written as shown in Listing 4.

The `SOAPCodec.decodeEventList()` method is used here to convert event objects from XML elements to Java objects that can be forwarded to the event broker runtime via a Java interface.

Once we implement this portion of the service, we can compose some very interesting scenarios. Consider, for instance, the configuration presented in Figure 4.

With the help of a servlet capable of processing events encoded as SOAP messages, we can build an event gateway that allows applications located outside of a company firewall to send events to subscribers deployed on the internal company network. When an outside application submits a SOAP message to the gateway, these sub-

Altova

<http://xmlj.altova.com/xmlspy5>

scribers will receive MQSeries, MSMQ, or JMS messages.

Event Forwarding

At this point we can receive events over SOAP/HTTP and translate them to events in the event broker runtime system. As the next step, we enable the publish/subscribe service to forward events received from the event broker to external Web based event subscribers. We can accomplish this by implementing an event listener class (see Listing 5).

The publish/subscribe service is now virtually complete. Web-based applications can use it to send and receive events. In addition, it can play the role of a gateway translating event messages between SOAP/HTTP and messaging protocols used on local networks such as MSMQ, MQSeries, or JMS (see Figure 5). Subscription management is the only building block that is still missing.

Subscription Management

As I mentioned earlier, we decided to use RPC encoding for the subscribe() operation of our service. We can now define a WSDL contract for this operation. Once the contract is ready, we can use a WSDL compiler to generate client-side and server-side stubs for automatic encoding and decoding of SOAP messages.

Writing a WSDL contract by hand is not

for the weak of heart, however. A typical WSDL document is written in cryptic XML and consists of many interrelated sections. It's easy to make a mistake while preparing it manually. Fortunately, most Web services development kits are capable of generating a WSDL contract from an interface defined in a conventional programming language such as Java or C#. We will take advantage of this capability and define the subscription management interface in Java as shown in Listing 6.

An application can subscribe for events from the publish/subscribe service by submitting a collection of subscription objects. Every subscription object carries a number of properties:

- An endpoint where the application is listening for events, e.g. "http://myapp.org:8080/listener"
- The path of a namespace in the event broker catalog, e.g. "Root/File System"
- The name of an event class defined in this namespace, e.g. "FileCreatedEvent"
- A query expression used to filter events, e.g. "path like '%.doc'"

Implementation of the subscription management service is very straightforward. Once the communication stubs are generated with a WSDL compiler, the IEventManager interface can be implemented (see Listing 7).

When a subscription request arrives from a

Web client, we use its endpoint to create a new event listener object. We then register this listener object and the event filter provided as part of the subscription request with the event broker using a method of the broker's API. When an event matching the subscription criteria occurs in the system, the handleEvent() method of the event listener is invoked by the event broker. The method encodes the event in SOAP/XML and posts it to the endpoint provided by the subscriber.

Conclusion

In this article I presented the design and implementation of a publish/subscribe Web service that can be used by applications deployed on the Internet to subscribe for, send, and receive events using ubiquitous Internet protocols. In addition, the service can work as a gateway between SOAP/HTTP and traditional messaging protocols such as JMS, MSMQ, MQSeries and others. We implemented the service as an extension of an existing event broker with help of an off-the-shelf Web services development kit.

The publish/subscribe service can be used in a large number of applications that require asynchronous one-to-many messaging: system monitoring and management, information replication, instant messaging, peer-to-peer computing, and others. ☺

Listing 1

```
// Retrieve the namespace handle from the catalog
// INamespace eventSource = (INamespace) root.find("File
// System");
// Create an event filter
IQuery filter =
(IQuery)eventSource.create("FileCreatedEvent",
ItemType.IT_Query);
filter.setQueryExp("path like '%.doc'");
// Create an event listener object
MyEventListener listener = new MyEventListener();
// Register the filter and the listener object with the
runtime system
EventBroker.subscribe(listener, filter);
```

Listing 2

```
// Retrieve the namespace handle from the catalog
INamespace eventSource = (INamespace) root.find("File
System");
// Create an event object and set its properties
```

```
IEvent event = (IEvent)
eventSource.create("FileCreatedEvent", ItemType.IT_Event);
event.setPropertyValue("path", "/Docs/Important Paper.doc");
// Broadcast the event to local and remote listeners
EventBroker.raise(event);
```

Listing 3

```
<env:Envelope
xmlns:env="http://schemas.xmlsoap.org/soap/envelope"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
  <env:Body>
    <broker:raise
xmlns:broker="http://eventcentral.com">
      <ev:FileCreatedEvent
xmlns:ev="http://eventcentral.com/Root/File System">
        <path
xsi:type="xsd:string"/>Docs/Important Paper.doc</path>
      </ev:FileCreatedEvent>
```

```

        <ev:FileDestroyedEvent

xmlns:ev="http://eventcentral.com/Root/File System">
        <path xsi:type="xsd:string">/Docs/Junk
Paper.doc</path>
        </ev:FileCreatedEvent>
        </broker:raise>
    </env:Body>
</env:Envelope>

```

Listing 4

```

public SOAPMessage onMessage(SOAPMessage requestMsg)
{
    ...
    SOAPEnvelope requestEnv =
requestMsg.getSOAPPart().getEnvelope();
    SOAPHeader requestHeader = requestEnv.getHeader();
    SOAPBody requestBody = requestEnv.getBody();
    SOAPElement raiseElem = (SOAPElement)
requestBody.getChildElements().next();

    // Decode events and forward them to the runtime system
    IEvent [] events =
SOAPCodec.decodeEventList(requestEnv, raiseElem);
    for (int k = 0; k < events.length; k++)
    {
        // Raise the event in the runtime system
        EventBroker.raise(events[k]);
    }
    // Compose the response message
    SOAPMessage responseMsg =
messageFactory.createMessage();
    ...
    return responseMsg;
}

```

Listing 5

```

public class EventListener implements IEventListener
{
    private String endpoint; // end point of a web based
listener
    public void handleEvent(IEvent event)
    {
        ...
        SOAPConnection connection =
connectionFactory.createConnection();
        SOAPMessage requestMsg =
messageFactory.createMessage();
        SOAPEnvelope soapEnv =
requestMsg.getSOAPPart().getEnvelope();
        SOAPBody soapBody = soapEnv.getBody();
        SOAPBodyElement command = soapBody.addBodyElement(
            soapEnv.createName("raise", "broker",
"http://eventcentral.com"));
        SOAPCodec.encodeEventList(soapEnv, command, new

```

```

IEvent [] { event });
        requestMsg.saveChanges();
        SOAPMessage responseMsg =
connection.call(requestMsg, endpoint);
        connection.close();
        ...
    }
    ...
}

```

Listing 6

```

public class Subscription
{
    public String endpoint; // an endpoint used to receive
events
    public String nsppath; // path of a namespace contain-
ing the event class
    public String classname; // name of an event class
    public String filter; // a query expression for fil-
tering events
}
public interface IEventManager
{
    public void subscribe(Subscription [] subscriptions);
    public void unsubscribe(String endpoint);
}

```

Listing 7

```

public class EventManager implements IEventManager
{
    public void subscribe(Subscription [] subs)
    {
        ...
        for (int k = 0; k < subs.length; k++)
        {
            // locate the event source namespace in the
catalog

            INamespace nsp = root.find(subs[k].nsppath);
            // create an event filter
            IQuery q = (IQuery) nsp.create(subs[k].class-
name, ItemType.IT_Query);
            q.setQueryExp(subs[k].filter);
            // create an event listener object
            EventListener lsn = new
EventListener(subs[k].endpoint);
            // register the listener object with the run-
time system
            EventBroker.subscribe(lsn, q);
        }
    }
    ...
}

```

Download the code at
sys-con.com/webservices



Written by Joseph A. Mitchko

**Author Bio:**

Joe Mitchko is a technology specialist for a leading Internet service company and is product review editor and a contributing writer for Web Services Journal.

JOE@SYS-CON.COM

Actional SOAPstation

Finding order and scale in complexity

A favorite dot-com-era TV commercial of mine depicts several young entrepreneurs eagerly watching a ticker for their first e-commerce transaction. Their initial cheers and high-fives quickly faded into dead silence as the transactions started mounting into the millions. How could they fulfill all of those orders? Now, imagine corporate IS managers watching the growing number of specialized Web services spring up across the enterprise. How can they centrally manage all those services?

Following SOAPswitch, its successful initial Web services product, Actional has announced a new product aimed at bringing order to the complex back-end configurations required to service a wide variety of specialized business partner-based Web services. SOAPstation is an entirely configurable, rule-based engine designed to address many corporate IS concerns when a SOAP request traverses the network.

To begin, SOAPstation is a Web service proxy, which means that it will handle a SOAP request on behalf of the actual service point, do some processing, and then forward the request to the real service.

To better understand SOAPstation's capabilities, let's follow the path of a Web service SOAP request as SOAPstation processes it through a series of message-processing blocks called a service group.

SOAPstation starts by authenticating the SOAP request, using a variety of configurable security methods such as HTTPS certificates and SAML assertions. Once the SOAP request is authenticated, it will go through a transformation-processing block, where you might

have some predefined data transformation take place so that the message complies with an internal service format.

Next, an access control processing block will determine what information, if any, the SOAP request is authorized to access. If not authorized, SOAPstation will kick back the appropriate SOAP response. Next, the SOAP request will pass through an interception block, a kind of configurable event driver that allows certain actions to take place when certain rules are met in the SOAP request message.

Finally, the SOAP message moves through a routing processing block. Here, depending on the data contained in the request, or other factors such as date and time, the SOAP request will be routed to the appropriate managed service. Before this happens, the SOAP request may go through an additional transform that is specific to the managed service. The SOAP response message will take a similar path through SOAPstation on its return trip.

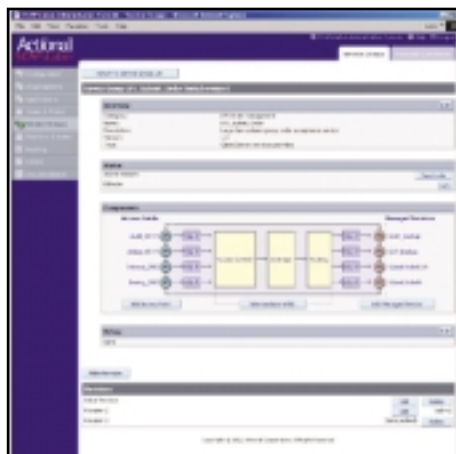


FIGURE 1 Actional SOAPstation's intuitive user interface and wizard-driven configuration simplify the management of the many and complex relationships between service providers and consumers.

Actional

COMPANY INFO

Global Headquarters
701 N. Shoreline Blvd.
Mountain View, CA 94043
Tel: 650 254-4100
Web: www.actional.com
E-mail: sales@actional.com

EVALUATION DOWNLOAD

For evaluation software, contact sales@actional.com

LICENSING INFORMATION

Licensed on a per CPU basis. For pricing information, contact sales@actional.com

TESTING ENVIRONMENT

OS: Windows-XP, Solaris
Hardware: Dell Inspiron 8000

In addition to being able to monitor Web service traffic, the SOAPstation Admin Console provides you with the capability to set up and configure a wide variety of rules using simple wizard-based configuration screens. In fact, the entire process of setting up a managed Web service is very straightforward and can be achieved in most cases in a few minutes. The exception will be for some of the more complex rules or transformations where you may need to throw in some custom Java code or XSL transformations.

SOAPstation also provides built-in facilities for adding instrumentation and logging to your managed services. For instance, you can easily set up rules for a service that allow you to e-mail an administrator when certain events take place in the message flows, such as increases in error counts or network timeouts.

Conclusion

SOAPstation is designed to bring order to the complex decisions that are required when processing Web service requests, and provides the ability for corporate IS to scale Web service offerings across the enterprise in an organized and controlled manner. It was generally available in late 2002. ©

Edge Web Hosting

www.edgewebhosting.com

Web Service Versioning and Deprecation

An easy-to-implement strategy for success



Current standards for SOAP, WSDL, and UDDI have no explicit support for the versioning and deprecation of Web services. This article introduces a means for Web service versioning and deprecation that is lightweight and flexible, and requires minimal development effort.

This approach is intended primarily for use within a corporation using one or more internal UDDI registries. It may be applied in the UDDI public registry “cloud,” but only for controlling versioning and deprecation for Web services controlled by a single provider.

Requirements

The following is a preliminary set of requirements that a successful strategy for Web service versioning and deprecation should support.

1. The ability for a consuming application to automatically determine and consume the most recent version of the Web service that supports the required functionality
2. A standard means through which the Web service version implementation may be discovered.
3. The ability to have multiple versions of the same Web service running simultaneously (to avoid the need for all applications to perform a hard cutover)

4. A standard means through which an application may discover that a Web service has been deprecated
5. The ability to support an arbitrary “sun-setting” length for a given Web service
6. The ability for an application to check, at any time, whether it should be using a more recent version of a Web service (i.e., once an hour, once a day, once a week, etc.)
7. Integration with source code control

Versioning

One approach to these requirements is a Web service versioning and deprecation strategy centered on applications searching for and binding to Web services by searching for services that support one or more required interfaces (tModels). The following

.NET and WSDL

The Microsoft .NET Framework generates the WSDL file dynamically from the service as an associated function of that service (i.e., the user retrieves the WSDL by calling `http://localhost/Service1.asmx?wsdl`). In the .NET environment, the developer should save that generated WSDL as a static file, and point the Overview URL for the tModel to that static file. Binding a static tModel to a dynamic WSDL file appears to invite problems, as when a future programmer changes the Web service, and unwittingly causes the dynamically generated WSDL to change.

scenarios illustrate the strategy, first from the Web service developer side, and then from the application developer side.

Scenario: Web Service Developer

The scenario begins with the Web service developer developing a new Web service, or modifying an existing Web service. One message supported by the Web service is `getVersion()`, which returns a complex data structure of three integers (major version, minor version, and patch level); the developer codes this function to return the appropriate values for the Web service (the reference implementation code distributed with this document shows an example of the Version class, and the `getVersion` Web method).

As part of the Web service development or deployment, the developer creates a WSDL (Web Service Description Language) file, describing the interface of the Web service. The developer registers this WSDL file as a TModel in an appropriate UDDI registry, with the **Overview URL** field in the TModel data structure pointing to the URL of the WSDL file.

If the deployed service is a new version of an existing service, it is registered as a new binding on the existing service, and that binding is linked to all the tModels it supports.

Scenario: Application Developer

At application design time, the developer discovers an appropriate Web service to reuse. The developer codes his or her application to use the messages specified in the WSDL (tModel).

The developer incorporates code to find a service binding dynamically, at application startup time (in lieu of looking up the access point from a configuration file, or hard-coding the value), at predetermined intervals, and/or in exception handling (i.e.,



Author Bio

Jeff Kenyon is a staff software development engineer at Qwest, where he was a technical lead on the UDDI effort. He holds an MS in information science and is a Sun Certified Programmer and Web Component Developer.
JKENYON@INDRA.COM

BEA eWorld

www.bea-eworld.com

when a Web service call fails). This code, taking as its argument a tModel key corresponding to the expected interface, searches for all services supporting the specified interface.

Special Cases

The basic versioning scenario may not be applicable in all instances. However, since it is a straightforward use of the existing UDDI standards, we believe that the strategy can be easily extended to accommodate more unusual cases.

- **Application requires a Web service supporting multiple tModels:** Since bindings may support any number of tModel bindings, an application may eventually require a Web service that supports multiple tModels. In this event, the code provided in this article may be used to search for bindings supporting each of the needed tModels, and then select a binding from the intersection of the individual binding lists.
- **Identical versions of a service that report different data:** Multiple services, each supporting the same interface, may return different data sets. As an example, consider a sales reporting Web service, implemented to provide headquarters with a simple means of pulling sales data from a wide variety of servers around the world, in a standard format. All are registered as separate services (e.g., Western Region

Sales Data, Eastern Region Sales Data, etc.), but all support the identical interface. In this event, the application developer would need to limit the search for bindings to those bindings within one specific service (identified by the service key), rather than searching for bindings across all services.

Implementation

All Web services implementing the strategy must support a getVersion() message. The message returns a complex data type, Version, consisting of three integer values (major, minor, and patch).

Should an application developer require a Web service binding, he or she searches by tModel key for the service bindings supporting that tModel (while it is possible to search by name, there is nothing preventing multiple tModels from all having the same name). For each service implementing that interface, the application retrieves the version information (using getVersion()) and selects the highest (i.e., most up-to-date) version of the Web service.

For example, take the following UDDI entry shown in Table 1.

An application programmed to use the interface described in Service 1 Interface would be able to use Service 1 v1.01 initially; when Service 1 v1.5.0 was released, the application would automatically begin using it (since it is the highest version supported by the required interface). When Service 1 v2.0 is released, the application will ignore it; even though it is the same service, it does not support the required interface.

Public Registry Applications

As mentioned above, while this strategy works within the bounds of an internal registry, it must be applied with more caution for Web services in the public UDDI “cloud,” for a number of reasons.

One issue is that of trust. In the world of public Web services, trust should be established prior to using a service:

- **To guarantee the source and accuracy of the data:** While it may be that a tModel specifies source, accuracy, and other characteristics of the data returned, it cannot be automatically

assumed that the service provider has fully or correctly implemented the tModel.

- **To avoid incurring unknown service charges, or inadvertently entering into a service contract:** Although UDDI business models have not yet been established, it is conceivable that one model might be that by using a particular provider’s version of a service, you are automatically agreeing to be billed for that service (at a very unfavorable rate).

Another issue that could arise when applying this strategy blindly within the public registry is that of multiple distinct services, from one or more providers, supporting the same interface. As a simple example, take an interface for retrieving stock quotations. On January 1, 2003, CompanyA releases version 1.0 of a service implementing the interface, and in April of the same year, follows up with version 1.5. In July, CompanyB releases version 1.0 of their service implementing the interface. Under the versioning strategy described, applications would continue to use CompanyA’s version 1.5 service, despite the fact that CompanyB’s service is actually the most recent version of the service implementing the interface. Version numbers are, and always have been, meaningless between software providers.

Versioning Code Walkthrough

In this section, we’ll examine some aspects of the versioning code. The code examples used are Java and UDDI4J, but the concepts are easily transferred to .NET.

First, assume the existence of a Web service supporting a getVersion message (see Listing 1). The values populated for major, minor, and patch are under the control of the Web service. Thus, the ability to allow the source code control system to supply the values returned by getVersion is totally under the control of the developer, and is dependent upon the programming language and source code control system chosen.


The code required to locate the binding(s) corresponding to the most recent version of the service is, unfortunately, rather complex when implemented directly in UDDI API v1.0 calls. The complexity is due to the API

TABLE 1: UDDI entry

• Company 1 (Service Provider)

• Service 1

- Binding: Service 1 v1.0.1, supports:
 - tModel: Service 1 Interface
- Binding: Service 1, v1.5.0, supports:
 - tModel: Service 1 Revised Interface
 - tModel: Service 1 Interface
- Binding: Service 1, v2.0.0, supports:
 - tModel: Service 1 Expanded Interface
 - tModel: Service 1 Revised Interface



not supporting searches for services, across all businesses that support a given tModel. As a result, the code is somewhat clunky: we must retrieve a list of all businesses, then for each business, look at each service to discover whether there is a binding supporting the given tModel (represented below by the String tModelKey, containing a valid TModel key with the prefix "uuid:"). This complexity goes away with the UDDI API v2.0 (release version, not draft).

If the application developer is quite sure that he or she is interested only in services from a specific provider, he or she may optionally limit the search for bindings to that single provider (or to a pool of potential providers). While this would make the code considerably faster, the disadvantage is that if the service is shifted to a different provider, the service bindings would no longer be found. This particular disadvantage may be overcome by performing the wider search across all businesses only if the limited search fails (see Listing 2).

The BindingObj class holds the UDDI BindingTemplate instance, as well as storing the results of the getVersion() call on the access point named in the BindingTemplate (by storing the BindingObj in a Java TreeSet with a custom comparator used to sort based on version numbers, we avoid the need to do an explicit sort later).

The call to the Web service's getVersion() message is handled in the same way as a call to any other message in the service to be consumed. The developer should keep in mind that getVersion() returns a complex data structure of three integer values. As an example, when handling the response from a .NET service in Apache SOAP, it is necessary to explicitly deserialize the structure (see Listing 3).

The reference implementation provided with this article contains the full code examples (source code for this article can be found online at www.sys-con.com/webserver/sourcecec.cfm).

Deprecation

If service versioning is implemented using the method described above (i.e., searching by bindings based on tModel),

deprecation of a Web service may be implemented by deprecating the specific tModel(s) supported by that Web service.

When a specific tModel is to be taken out of service, its UDDI registry entry is modified to add the identifier DEPRECATED, with a value of the date (in YYYY/MM/DD format) after which the tModel is no longer supported. Optional identifiers are DEPRECATED_MSG, containing additional details on the deprecation, and DEPRECATED_LINK, a URL to which users may refer for information that is more detailed.

To support deprecation, the application developer inserts a check into the service binding code; prior to looking for services that support the tModel, the tModel "identifierBag" data structure is checked for the DEPRECATED flag and value. There are three possible outcomes to this check:

- **The DEPRECATED flag is not set:** The code continues to try to find the most recent service version.
- **The DEPRECATED flag is set with a date greater than the current date:** The result is a console message warning of the deprecation (optionally populated with information from DEPRECATED_MSG and DEPRECATED_LINK). The code continues to look for the most recent service version.
- **The DEPRECATED flag is set with a date less than or equal to the current date:** The result is a fatal exception (optionally populated with information from DEPRECATED_MSG and DEPRECATED_LINK).

To implement this deprecation method, developers only need to insert their own code when encountering the aforementioned conditions, and direct the information appropriately within their application (e.g., to the console, to an alarming system, e-mail to the administrator, etc.).

The Web service should be kept in service for some locally defined minimum period of time following the deprecation date of its last tModel(s), and its access logs checked on a daily basis. Continued access implies that an application has bound directly to the Web service (or that deprecation warnings have been ignored), and it will be necessary to immediately follow up

with the application developers to directly notify them of the Web service's impending demise. The application architect should also be contacted, to provide remedial education on the recommended method of binding to a Web service.

After this period, the Web service may be removed from service. The owner of the tModel should then delete the registry entry for the tModel from any UDDI registries to prevent any inadvertent future use of that TModel.

Deprecation Code Walkthrough

Assuming the existence of a string tModelKey, containing a valid tModel key with the prefix "uuid:", the Java code in Listing 4 may be used to retrieve a tModel object.

Now that we have a tModel object (tm), we can examine the identifierBag in the deprecation-related tags (see Listing 5).

If the DEPRECATED identifier is not present, the deprecation check is complete. If it is present, we must compare the deprecation date to the current date to see whether the error produced should be nonfatal (deprecation has not yet occurred) or fatal (deprecation has occurred).

In either case, when we generate the message explaining the deprecation, we should include the content of DEPRECATED_MSG (a brief explanation of the deprecation) and DEPRECATED_LINK (a link to a more complete explanation of the deprecation; see Listing 6).

Development Roadmap and Final Notes

In summary, to implement versioning and deprecation, the following roadmap is recommended:

1. Web service developer builds support for getVersion message into his or her service.
2. Web service developer registers Web service in UDDI.
3. Web service consumer programs to interface, rather than implementation.
4. Web service consumer checks for deprecation of interface on a regular basis.

The process outlined in this document is reasonably lightweight. By limiting the require-

ments to placing versioning and deprecation data in standard locations, the developers are free to determine the best approach for implementation within their own application context. For an application that carries out a startup procedure at regular intervals, it may mean retrieving the appropriate binding upon startup and performing a deprecation check as a maintenance process once a month. For an application intended to run continuously, it may mean checking for more recent versions and deprecations once a week.

To be successful, Web service and application developers must follow the processes outlined here. While I believe that this strategy is logical and sound in its own right, it is further supported by the design of the UDDI businessService and bindingTemplate data structures. The absence of identifiers on these structures requires that versioning and depre-

cation information be stored either in the free-text description fields, or in some other data structure as dictated by some internal process.

One very important principle that Web service developers must follow is that any time they modify the WSDL interface, they must create a new tModel. A given tModel represents a version of a specification (in this case, an interface and implementation specification). Just as you cannot go back and modify an existing specification and keep the original version number, you cannot modify a tModel (or the underlying versions of the documents it refers to) once released. Developers will code their applications to the interface described in the WSDL file attached to the tModel but if the Web service developer changes that interface without issuing a new tModel, application Web service calls will suddenly start breaking.

This approach to versioning takes advantage of the features of UDDI, but relies upon the participation of both application and Web service developers in equal measure (sort of the Web services version of Mutual Assured Destruction). Application developers rely on Web service developers to implement the getVersion message, to maintain proper UDDI registry entries, and to continue supporting specific interfaces until they are officially deprecated. Web service developers rely on application developers to bind to a service based on the tModel required, and to use UDDI to find the binding for the latest version of a service supporting that tModel.

Programming to interfaces, rather than to specific implementations of Web services, ensures that application developers will always use the most up-to-date Web service, and will be notified in a timely manner of all (and only) relevant interface deprecations. ©

Listing 1: Web service supporting a getVersion message

```
<%@ WebService Language="C#" Class="SomeService" %>
using System.Web.Services;
public class SomeService {
    [ WebMethod ]
    public Version getVersion() {
        Version result = new Version();
        result.setMajor(1);
        result.setMinor(2);
        result.setPatch(3);
        return result;
    }
}

public class Version {
    private int major = 0;
    private int minor = 0;
    private int patch = 0;
    public void setMajor(int x) {
        major = x;
    }
    public int getMajor() {
        return major;
    }
    public void setMinor(int x) {
        minor = x;
    }
    public int getMinor() {
        return minor;
    }
}
```

```
public void setPatch(int x) {
    patch = x;
}
public int getPatch() {
    return patch;
}
}
```

Listing 2: Performing the wider search

```
UDDIProxy proxy = new UDDIProxy();
Collection results = new TreeSet(new VersionCompare());

// Find all businesses

Vector businessInfoVector = null;
BusinessList bl =
    proxy.find_business("%", new FindQualifiers(), 0);

try {
    businessInfoVector =
        bl.getBusinessInfos().getBusinessInfoVector();
} catch (NullPointerException e) {
    // do nothing; let the function return an empty collection
}

// Find all services

if (businessInfoVector != null) {
    for (int i = 0; i < businessInfoVector.size(); i++) {
        BusinessInfo businessInfo =
```



```

        (BusinessInfo)businessInfoVector.elementAt(i);

// Retrieve list of service
Vector serviceInfos;
try {
    serviceInfos =
businessInfo.getServiceInfos().getServiceInfoVector();
} catch (NullPointerException e) {
    // may not have any service details
    continue;
}

Enumeration enum1 = serviceInfos.elements();
while (enum1.hasMoreElements()) {

    ServiceInfo si =
(ServiceInfo)enum1.nextElement();

    // set up tModel bag
    Vector vTModels = new Vector();
    vTModels.add(tModelKey);

    TModelBag tmBag = new TModelBag(vTModels);

```

```

// Find all bindings matching tModel

BindingDetail bd =
    proxy.find_binding(new FindQualifiers(),
                        si.getServiceKey(),
                        tmBag,
                        0);

Vector vBindings =
    bd.getBindingTemplateVector();
Enumeration enumBindings = vBindings.elements();

while (enumBindings.hasMoreElements()) {
    BindingTemplate bt =
        (BindingTemplate)
enumBindings.nextElement();
    // add to TreeSet
    results.add(new BindingObj(bt));
}

return results;

```

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

BUY THOUSANDS
OF PRODUCTS AT
GUARANTEED
LOWEST PRICES!

GUARANTEED BEST PRICES
FOR ALL YOUR
WEB SERVICES
SOFTWARE NEEDS



N-ARY

\$299.00

n-ary Ticket System v2.0

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



GREENPOINT INC.

\$1,350.00

WEB CHARTS 3D VER. 4.7

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



CHUTNEY TECHNOLOGIES

\$755.25

Chutney SOAP+ Toolkit

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



ALTOWEB

\$2,500.00

Application Platform Release 2.8

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.



ZION

\$895.00

**JMessageServer(TM) Workgroup
(up to 50 users)**

An open Instant Messaging & Presence Server written in Java, with a Web-based account manager and database buddy list and presence. Zion supports JSuperChat IM clients and custom clients enabled with JBuddy SDK for real-time communications and Enterprise Application Integration (EAI).



POINTBASE

\$353.00

**PointBase Server 4.4
Windows Installer**

PointBase® Server is a 100% Pure Java RDBMS for client-server networked applications. PointBase Server provides a reliable and scalable database for workgroup servers, application servers, Web servers and other Internet-based applications needing storage of metadata.



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

OFFERS SUBJECT TO CHANGE WITHOUT NOTICE

Listing 3: Deserialize the structure

```
SOAPMappingRegistry smr =
    new
    SOAPMappingRegistry(Constants.NS_URI_2001_SCHEMA_XSD);
BeanSerializer beanSer = new BeanSerializer();
IntDeserializer id = new IntDeserializer();
smr.mapTypes(Constants.NS_URI_SOAP_ENC,
    new QName("http://tempuri.org/", "getVersionResult"),
    Version.class, beanSer, beanSer);
smr.mapTypes(Constants.NS_URI_SOAP_ENC,
    new QName("http://tempuri.org/", "major"), null, null,
    id);
smr.mapTypes(Constants.NS_URI_SOAP_ENC,
    new QName("http://tempuri.org/", "minor"), null, null,
    id);
smr.mapTypes(Constants.NS_URI_SOAP_ENC,
    new QName("http://tempuri.org/", "patch"), null, null,
    id);
call.setSOAPMappingRegistry(smr);
```

Listing 4: Java code to retrieve a tModel object

```
UDDIProxy proxy = new UDDIProxy();

try {
    // Point to UDDI server
    proxy.setInquiryURL
        ("http://localhost/uddi/api/inquire.asmx");
} catch (Exception e) {
    e.printStackTrace();
}

// Task 1A: Retrieve tModel identifiers
Vector vTM = new Vector();
try {
    vTM =
    proxy.get_tModelDetail(tModelKey).getTModelVector();
} catch (Exception e) {
    // Proxy threw a SOAP or UDDI error.
    // Throw fatal exception.
    System.err.println("UDDI Registry access error: " + e);
}

// take the TModel off the Vector.
TModel tm = null;
if (vTM.size() > 0) {
    tm = (TModel) vTM.elementAt(0);
} else {
    // No TModel returned. Hmmm. That shouldn't happen.
    // Exit with empty collection
    return bindings;
}
```

Listing 5: identifierBag

```
// Grab its identifiers
String deprecated = null;
String deprecatedLink = null;
String deprecatedMsg = null;

Enumeration enumIdentifiers =
```

```
tm.getIdentifierBag().getKeyedReferenceVector().elements();
while (enumIdentifiers.hasMoreElements()) {
    KeyedReference kr =
    (KeyedReference)enumIdentifiers.nextElement();
    if (kr.getKeyName().equals("DEPRECATED")) {
        deprecated = kr.getKeyValue();
    } else if (kr.getKeyName().equals("DEPRECATED_MSG")) {
        deprecatedMsg = kr.getKeyValue();
    } else if (kr.getKeyName().equals("DEPRECATED_LINK")) {
        deprecatedLink = kr.getKeyValue();
    }
}
```

Listing 6: Generated message

```
if (!(deprecated.equals(null))) {
    // convert deprecated value to date
    SimpleDateFormat formatter = new
    SimpleDateFormat("yyyy/MM/dd");
    Date today = new Date();
    Date deprecationDate = new Date();
    try {
        deprecationDate = formatter.parse(deprecated);
    } catch (ParseException e) {
        // Unparseable date string: set to 3000/12/31
        try {
            deprecationDate = formatter.parse("3000/12/31");
        } catch (ParseException e1) {
            System.err.println("Very serious parsing problem: " +
            e1);
        }
    }

    if (deprecationDate.after(today))
    {
        // value >= today, exception
        // include values of DEPRECATED_MSG and DEPRECATED-
        LINK in
        // any exception.
        System.err.println
        ("TModel will be formally deprecated after " +
        formatter.format(deprecationDate) +
        " (" + deprecatedMsg + " [" +
        deprecatedLink + "]).");
    } else if (deprecationDate.equals(today) ||
        deprecationDate.before(today)) {
        // value < today, fatal exception
        // include values of DEPRECATED_MSG and DEPRECATED-
        LINK
        throw new WebServiceDeprecationException
        ("TModel has been formally deprecated as of " +
        formatter.format(deprecationDate) +
        " (" + deprecatedMsg + " [" +
        deprecatedLink + "]).");
    }
}
```

Download the code at

sys-con.com/webservices

introductory
subscription offer!

A TRULY INDEPENDENT VOICE IN THE WORLD OF .NET

.NET Developer's Journal is the leading independent monthly publication targeted at .NET developers, particularly advanced developers. It brings .NET developers everything they need to know in order to create great software.

Published monthly, *.NET Developer's Journal* covers everything of interest to developers working with Microsoft .NET technologies – all from a completely independent and nonbiased perspective. Articles are carefully selected for their prime technical content – technical details aren't watered down with lots of needless opinion and commentary. Apart from the technical content, expert analysts and software industry commentators keep developers and their managers abreast of the business forces influencing .NET's rapid development.

Wholly independent of both Microsoft Corporation and the other main players now shaping the course of .NET and Web services, *.NET Developer's Journal* represents a **constant, neutral, expert voice** on the state of .NET today – the good, the bad, and the ugly...no exceptions.



SYS-CON
MEDIA



SUBSCRIBE ONLINE!

www.sys-con.com/dotnet/

or Call

1 888 303-5282

Here's what you'll find in
every issue of .netdj:

Security Watch

Mobile .NET

.NET Trends

Tech Tips

Standards Watch

Business Alerts

.NET News

Book and Software
Announcements



.NET Developer's Journal is for .NET developers of all levels, especially those "in the trenches" creating .NET code on a daily basis:

- For beginners:
Each issue contains step-by-step tutorials.
- For intermediate developers:
There are more advanced articles.
- For advanced .NET developers:
In-depth technical articles and columns written by acknowledged .NET experts.

Regardless of their experience level, *.NET Developer's Journal* assumes that everyone reading it shares a common desire to understand as much about .NET – and the business forces shaping it – as possible. Our aim is to help bring our reader-developers closer and closer to that goal with each and every new issue!

SAVE 16% OFF

THE ANNUAL COVER PRICE

Get 12 issues of .NETDJ
for only \$69⁹⁹!

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

ANNUAL
COVER PRICE:

~~\$83.88~~

YOU PAY

\$69⁹⁹

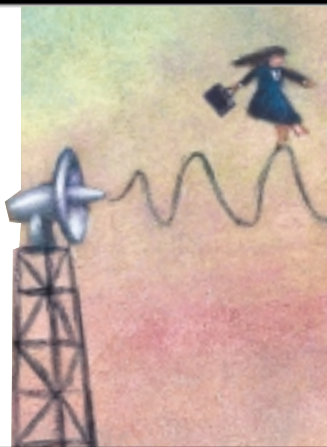
YOU SAVE

\$13.89

OFF THE ANNUAL
COVER PRICE

On the Road to Web Service-Level Management

Critical components in the creation of self-managing computer resources



Web services is now delivering on the promise of interconnecting systems, within and between organizational boundaries. But the benefits of open interoperability of such distributed resources only increase the complexity of the computing environment that has to be managed.

AUTHOR BIOS:

Mark Potts is the chief technology officer at Talking Blocks (www.talkingblocks.com), a U.S.-based software development company. As CTO, he is defining the vision and direction for products enabling the next generation of management surrounding Web Services. Mark is Talking Blocks' representative for the W3C Web Services Architecture Working Group and the OASIS Business Transaction Protocol Technical Committee.
MARK.POTTS@TALKINGBLOCKS.COM

Kevin Ruthen is with IBM Business Consulting Services where he acts as a senior architect and manager within Financial Services. He has strong expertise in e-business integration technologies, such as Web services, to create and enable business strategies with strategic alignment of technology processes. Kevin focuses on leadership of teams for the design, architecture, and development of business solutions leveraging e-business technologies.
RUTHEN@US.IBM.COM

Heather Kreger is the Web services lead architect for IBM's Emerging Technologies. She is responsible for helping Web service integration, manageability, and adoption across IBM. Heather is colead of JSR109, which specifies Web services deployment in J2EE environments. She is IBM's representative and an editor for the W3C Web Services Architecture Working Group and the OASIS Web Services Management Protocol Working Group. Heather is coauthor of Java and JMX: Building Manageable Systems (Addison Wesley).
KREGER@US.IBM.COM

Earlier this year Gartner published a report defining a Web services management platform as one of four platforms required for successful Web service deployments. In this article we'll look at the role traditional systems management has played in the enterprise, requirements specific to managing a Web services environment, and how Web services in a managed environment can improve service levels while reducing the overhead and costs associated with managing complex distributed environments.

Systems Management

Since Web services are WSDL-described network-accessible software components, some of the basic requirements for the management of Web services are the same as those for the management of applications. Traditionally, application management has meant the monitoring and control of an application throughout its life cycle (installation through startup, execution, configuration, to final shutdown and decommissioning).

Monitoring includes collecting metrics

and events relevant to the application from the application's execution environment (hardware, operating system, etc.), as well as the application itself. Control could include installation, configuration, startup, shutdown, and general health and status, as well as real-time tuning to ensure optimal performance. In order for an application to be "monitored" and controllable, it needs to expose basic management information including identification, status, metrics, configuration, operations, and events. This set of information is referred to as the manageability model for the application. The information populating this model may be supplied explicitly by the application, implicitly through the application's environment, or through both channels.

The typical architectural model for enterprise systems management is the manager-agent. In this model, the management system communicates with an agent using a predetermined protocol that may be proprietary or based on standards. The agent is usually local to the application and responsible for communicating with the

managed applications and the management system. The agent forwards events from the application to the management system, and forwards requests from the management system to the application.

Management Standards

A number of management models and protocols were developed to standardize the manager-agent architecture, protocol, and information model. Even so, the most popular system management products deployed today, CA Unicenter and IBM/Tivoli, rely on proprietary protocols between their managers and agents. Fortunately, they also consume information from managed resources using standardized management technologies like the Simple Network Management Protocol (SNMP) and the Common Information Model (CIM).

SNMP, from the IETF, was developed as a stopgap solution for accessing management information on a device. SNMP, however, lacks native support for operations, secure authorization, and relationships. At the same time, it became obvious that the information model needed to be

Now in More

than 5,000

bookstores

Worldwide

Subscribe Now!

FOR FAST DELIVERY

SPECIAL
INTRODUCTORY OFFER
SAVE \$31*
HURRY, DON'T DELAY! OFFERS SUBJECT TO CHANGE WITHOUT NOTICE

Helping
you enable
intercompany
collaboration
on a global scale

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

Go Online and Subscribe Today!

WebLogicDevelopersJournal.com

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

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

**SYS-CON
MEDIA**

standardized independently of how the information was expressed or accessed.

The Distributed Management Task Force (DMTF) has been defining a standard manageability model, CIM for IT resources, for over five years. One of the most important benefits of CIM is the common vocabulary for simple concepts like status and description. The information in a CIM model can be accessed using Web Based Enterprise Management (WBEM), which defines a protocol using XML-encoded CIM meta-schema, classes, and instances over HTTP. While many of the systems, devices, and network models are very mature and complete, the application model is still being developed.

Nearly all the existing management technologies, protocols, and models have been focused on managing the configuration and status of specific resources rather than business views of distributed applications and systems, in essence, business services. End-to-end views of resources used by a business system are hard to develop and understand and the status of those systems even harder to infer. The traditional coarse granularity of management status (up, down, degraded, etc.) does not provide enough context to tell a busy operator or business administrator whether a system, or service, needs attention. More importantly, it doesn't tell them if the system is behaving *as expected* by its users, namely their business partners, suppliers, customers, or other internal personnel.

Web Service Management

The emergence of service-oriented architectures (SOA) and industry commitment to Web services and Grid computing requires new management capabilities to be available in order to truly achieve the

visions and goals of these initiatives. Both Web services and Grid services leverage an environment based upon deploying services that are loosely coupled across heterogeneous, dynamic environments. To effectively manage such a dynamic environment, it is critical that management decisions be based on messages (context and payload), service descriptions, and service provider information, and decoupled from traditional management facilities. Management of this new environment extends past the traditional systems and applications to include administration of services and provide management visibility and control at the service level.

Gartner defines the Web Services Management Platform (WSMP) as "a set of software services that is designed to help coordinate the activities of services while they are being used." Different service provider platforms (e.g., WebSphere, .NET, etc.) will not be able to manage services deployed on other platforms due to disparate management components. Gartner views a WSMP as the bridge to enable and provide interoperability for managing services across platforms.

Service-Based Management

Today most businesses are not investing in IT without a clear return on investment, lower total cost of ownership, and clearly demonstrated cost savings. Investments made in Web services and future Grid service initiatives offer the opportunity to realize these requirements, but need to be deployed in a consistent, repeatable, and manageable fashion. Traditional operations management has not been able to offer the unique management functionality that can help achieve these requirements as compared to service-based management.

In order to achieve optimal IT investment, there must be strategic alignment between business requirements and IT investment and management to support that alignment, i.e., service-level management (SLM). SLM is achieved through the proper definition of services, relationships between services, and their correlation and representation as business processes. Traditional operations management platforms have been narrowly focused on specific systems and applications as opposed to a service-based dynamic environment.

Service-based management requires the provision and consumption of services in a nonintrusive manner while maintaining the loosely coupled nature of SOA. The WSMP does precisely this by acting as a transparent intermediary, or broker, between consumers and providers of services. The broker handles requests and manages the runtime provisioning of service endpoints to the requests dynamically. It finds the most appropriate service for service requests on demand. The broker then supports the interaction between consumer and provider with management facilities for availability, versioning, provisioning, configuration management, logging, auditing, alerting, error management, transformation, and integration with security facilities for authentication and authorization.

A WSMP that transparently mediates between Consumers and Providers to resolve service requests on demand offers two important advantages:

1. Messages received can be intercepted or inspected for additional information pertaining to the consumer or the request and taken into consideration; for example, identity, geography, time, price, etc., offering opportunities for differentiated service offerings or intelligent routing based on context.
2. Management can be proactive in resolving error conditions such as service failures or potential service-level breaches by intelligently routing messages to best available resources.

WSMP should not be seen as a replacement to systems management, but rather as a conduit and extension to external, broader management facilities such as those offered by Tivoli, CA, BMC, that can correlate and add management to the

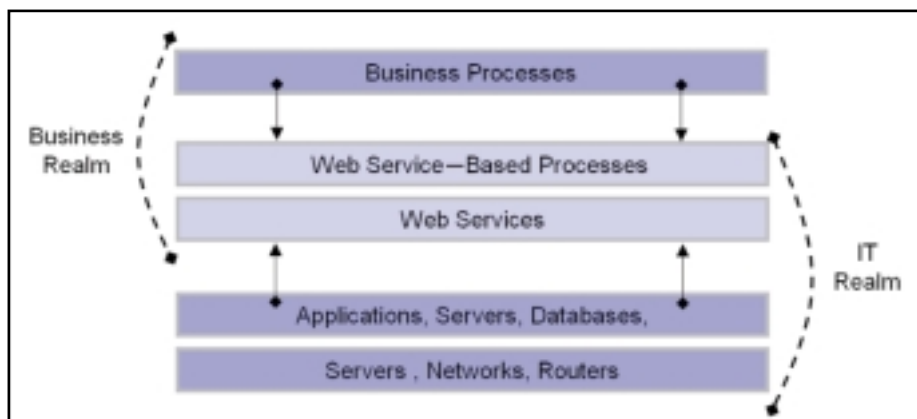


FIGURE 1 | Alignment surrounding Web services and Web service-based processes

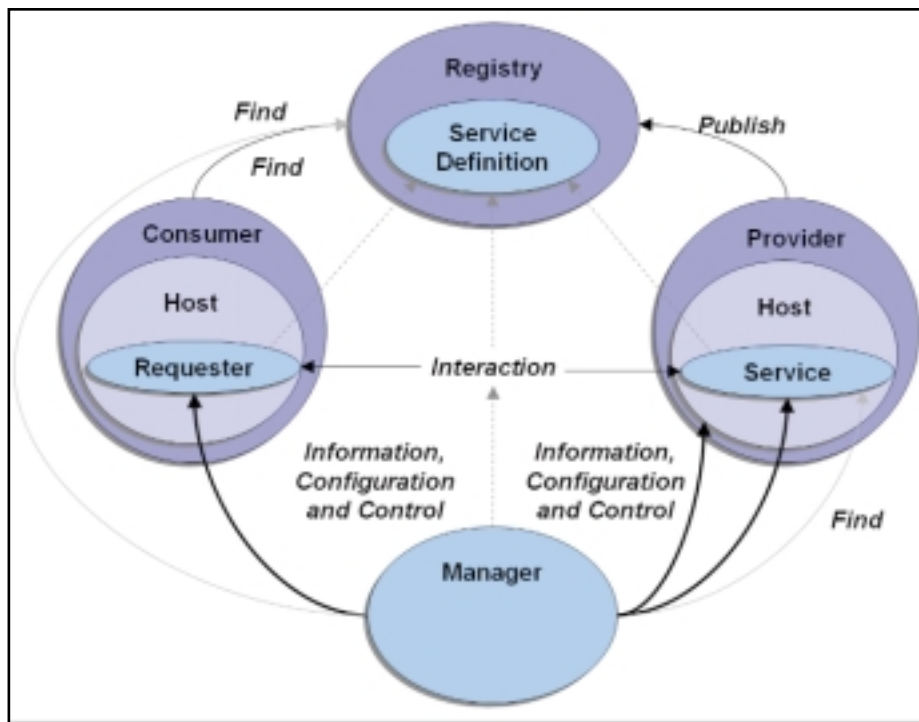


FIGURE 2 Management role in a Web services architecture

infrastructure used to implement the Web service (hardware, networks, etc.).

Web Service-Level Management

Web services and their description form the basis for a common definition of assets between business and IT. Once the process-based Web services and individual Web services are identified and described they can both have service-level objectives (SLO) and service-level agreements (SLA) associated with them. This then enables business and IT to have a common definition of assets to support the alignment of business needs and IT resources, as well as define a level against which the service will be measured and managed.

For systems management to really meet the needs of the organization, both the resources being managed (Web services) and the manager (WSMP) need to take on certain responsibilities. The resources must provide enough information and operational interfaces such that the resources can be centrally monitored and controlled. The Manager must be capable of analyzing the information provided by individual resources and correlating information from multiple resources, and provide the ability to act on the information to better manage the Quality of Service (QoS) being achieved by services and offered to consumers. The

manager, where appropriate, should also be able to manage the environment proactively such that every attempt is made to meet the declared QoS, whether those be internal SLO or a more formal SLA.

The major roles involved in interactions are the Provider and the Consumer; therefore, management should be addressed from the perspective of both roles. A Manager role, when seen from the Provider's perspective, has management capabilities and visibility beyond the Web service and into the service instance or implementation. The Provider, therefore, has management capabilities (visibility and control) over elements of the architecture that a Consumer would not, e.g. the hosting environment. The separation of service from its implementation and environment means that the lower-level elements that support the service are important to the Provider so they can manage at both levels. For example, service Providers may want to replicate service instances, launch new service instances to meet SLA at peak load times, or perhaps failover to alternative service instances hosted elsewhere. In this case, being able to manage the service as it is exposed to the Consumer and the elements of the architecture supporting that service is critical to meeting business objectives.

From the Consumer's perspective, only the service as defined by the service defini-

tion they have consumed can be managed. The Consumer acting as a Manager will have visibility and control of the requesters, but in most cases will not be offered management beyond visibility (metering and monitoring) for the service. For example, the Consumer (or a third party) may want to, and be allowed to, look at the performance and availability metrics or measurements offered by a service it consumes to ensure adherence to SLA.

There are three interrelated arenas of responsibility for Web services;

1. **Service monitoring and reporting:** Monitoring and reporting on the usage, health, and QoS being delivered by services
2. **Service execution management:** Routing of requests, differentiated service offerings, security, and fault management
3. **Service environment management:** Dependency management, deployment, non-disruptive versioning, and upgrades

Service Monitoring and Reporting

At the very least, a Manager has to be capable of garnering metrics and events about a service's performance, availability, usage, and configuration (policy driven). Metrics represent information logged by the service, gathered periodically, or requested at a point in time, and should be as raw as possible such that ambiguity in their meaning is avoided, since any type of metric or measurement is meaningless without knowing the formula used to derive value. The Manager also correlates information from multiple services such that measurements and monitoring cover entire transactions or processes that may involve multiple service interactions occurring over extended periods of time. Event monitoring includes listening for events posted by services that signal significant individual events, e.g. failure or state changes. Based on this management information, the Managers are capable of showing the health of the services and the system overall at any point in time.

Collected metrics are used to calculate measurements regarding performance availability and usage. With the Manager defining and recording measurements, services can be monitored to ensure they are meeting agreed upon service levels. Breach conditions and early warning of potential breach conditions can then be monitored and managed. The SLA agree-

ments that define measurements should include the formula used for measurements that define the parameters of the SLA (average response time, throughput, and availability). SLAs are defined for specific customers and therefore the metric collection needs to be consumer aware and tie directly into the policies for authentication and authorization for the service. SLAs also define periods of time for which service-level objectives are applicable. Again, this definition affects configuration of the services in terms of warning events and metrics.

Service Execution Management

Management of Web services is not just about providing metrics concerned with performance and availability, it is also about configuration of policies that drive execution and deliver the best QoS from the services being managed. This includes managing error conditions and fault situations, managing the load of requests based on the performance of available services and the identity of requesters, and ensuring security policies for authentication and authorization are adhered to.

Web services that are provisioned may well be supported by many instances of the service so that anticipated capacity can be met within the agreed service levels defined for the service. This means that requests for service need to be intercepted or inspected so they can be routed to the most appropriate service instance available. In Grid computing, this can also mean life cycle management, where more service instances can be launched (capacity on demand) to support the Consumers and meet SLA. Routing may also be affected by policies concerning the identity of the requester. Important Consumers may be shown greater consideration when routing their requests, or Consumers with a more stringent SLA may take priority over others with looser agreements, such that all SLA are met.

Web services can also evolve over time, especially where Consumer requirements drive changes into existing services. Routing must therefore be cognitive of versions and compatibility between services, such that rolling upgrades as well as side-by-side versions of services can be managed appropriately.

Managing the interactions between Consumers and Providers includes security

and is part of managing the overall QoS. A managed environment provides security services that enable applications to enforce Access Control, Identity Management, and Entitlement Management. Requirements that define Quality of Protection (QoP) may well be defined in SLA, and policies that enforce that QoP are part of the configuration of a service and need to be enforced and managed.

Errors and failures are inevitable in any environment but should be managed (resolved) wherever possible such that Consumers are unaware of problems and can continue undisturbed. Errors and fault conditions need to be intercepted by the Manager before being returned to the Consumer to see if there are any ways in

“
Web service management
platforms and Grid
computing are critical
components on the path
to creating dynamic,
service-centric networks
of self-managing
computing resources”

which the faults can be resolved. This is, again, based on configuration metadata and driven by policies that govern retry attempts with the requested service, routing to alternate services, back off times, and overall timeouts for requests.

Service Environment Management

As more Web services are deployed in organizations, they are unlikely to be isolated and will depend on other services to complete the functionality they offer. This means that deployed services will need to be managed in terms of their dependencies and relationships to other services. Management therefore needs to ascertain information about services and their relationships at deployment time so that they can be better managed. Relationships include requirements and dependencies – for example process-based Web services should define

the services (more appropriately service types) they require for the activities defined within the process. Relationships can also be declared with respect to the lineage of the service, which encapsulates compatibility or noncompatibility between service versions. Relationships should also define conflicting or exclusive relationships where only a single instance of a service can be available at one time in a managed environment. These relationships help the Manager maintain the QoS of the services it manages in terms of availability, problem cause analysis, and evolving service deployment.

Information specific to a service definition (functional) can also be related to other information concerning the service, for example the generic SLA that applies to this service, or other service definitions (operational) that can be utilized at runtime for monitoring and configuring services.

The Manager uses these declared relationships and associated meta information at deployment time and runtime. Once deployed, the Manager is responsible for publishing the service to the appropriate discovery mechanism, making management and operational controls available to management applications and consoles, and accommodating non-disruptive evolution of the service. Evolution includes managing side-by-side versions of services and routing appropriately between them based on Consumer requests and rolling upgrades, where service implementations can be changed and extended without disruption to the Consumers.

All three areas of management are interrelated and need to work collaboratively within a WSMP to deliver real business value to an organization. For example, performance and availability information needs to be made available to the execution management facilities such that it can effectively manage QoS; service level objectives need to be considered for reporting and monitoring and execution management, security and identity information needs to be utilized for differentiated service offerings and service relationships need to be used for correlations of management information and metrics and support of nondisruptive change management.

Moving to Web Service-Level Management

In order for Web services to be manageable in an interoperable manner, three building blocks must be standardized: the basic

manageability information that the components of the Web services architecture must support, how that information is accessed, and how the manageable components are discovered.

The first of the building blocks, the minimum, basic information required for managing Web services and their environment, is being developed and standardized in the W3C's Web Services Architecture Working Group's Management Task Force. The Web Services Architecture includes a requirement that implementations of the architecture must be manageable. A task force was initiated to satisfy this requirement and is working to publish a manageability model for each of the components of the Web services architecture, i.e., services, hosting environment, and discovery agency. The manageability model must include identification, configuration, metrics, and events for the components.

The second and third of the building blocks, access to and discovery of the management information, are being developed and standardized in the OASIS Management

Protocol Technical Committee (MPTC). The MPTC is defining how to access manageability information for *any* managed resource using Web services. The same specification should also work for Web services as a specific type of managed resource.

These building blocks have been discussed in terms of how to manage Web services in particular, but you can see that the same principles can be applied to managing IT resources in general. The use of Web services to expose IT resources on Grid systems is being developed and standardized by GGF at Globus. They are also defining how to manage these IT resources using Web services and Grid services. It is logical that they should be able to leverage the same foundation building blocks being developed by the W3C and the OASIS MPTC.

Conclusion

Management of distributed computing environments has always been difficult, but the dynamic nature of Web services and the more loosely coupled nature of interactions

make it even more difficult to control and administer. However, Web services, coupled with service-level management in a WSMP, offers opportunities for organizations to better leverage their existing IT investment, better manage IT assets and resources in alignment with business objectives, and introduce automatic concepts in managing their IT infrastructure that effectively reduce the overhead and cost of managing and maintaining their operational systems. Web services management should not be seen as a requirement once services are deployed and proliferating throughout the enterprise, it is a requirement for day-one deployment of your first Web service.

Web service management platforms and Grid computing are critical components on the path to creating dynamic, service-centric networks of self-managing computing resources and as Gartner rightly points out, "enterprises that do not embrace the producer and management platforms will fail to deliver any Web services beyond trivial initiatives through 2004." ☺



THE INSIDER INTELLIGENCE YOU NEED...

TO KEEP AHEAD OF THE CURVE

Go to www.SYS-CON.com

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

JAVA > Newsletter

WebServices > Newsletter

XML JOURNAL > Newsletter

wireless > Newsletter

WebLogic > Newsletter

WebSphere > Newsletter

GoldFusion > Newsletter

.NET Journal > Newsletter

FREE

E-Newsletters

SIGN UP TODAY!

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

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

on-Demand computing

Written by Bernhard Borges

Incrementally adapting to a new frontier

When an enterprise needs more electrical power, it doesn't usually build a generating station. When it needs to transport employees to meetings in far-flung places, it generally does not build its own aircraft. Instead, an enterprise would consume services from an existing network of available services. The growing mantra of On-Demand Computing (ODC) suggests that similar ease and efficiencies may one day be available for enterprise computing.

AUTHOR BIO:



Bernhard Borges is a senior technologist for IBM's Business Consulting Services EAI practice. His areas of expertise are distributed computing, J2EE, Web services, and enterprise integration. Bernhard holds a Ph.D. and MBA

from Simon Fraser University, Burnaby, BC, Canada and resides in Phoenix, Arizona.

BERNHARD.BORGES@US.IBM.COM

That day is not yet here. The end game of perfect ODC – in which resources flow seamlessly from where they are available to where they are needed – is at least a decade away. Between now and then, the transition path is fraught with peril. But the greatest danger is to completely ignore the reality that ODC brings to the table. Viable interim ODC service offerings are already available. Early adopters are embracing ODC-based solutions.

Undoubtedly, the magnitude and scope of ODC makes it one of the most disruptive forces in the technology-enabled enterprise space. In a world where the traditional “make or buy” decision is entirely supplanted by an option to consume on demand, those who can adapt to this new reality will radically out-perform the competition. To win that race, however, enterprises need an appreciation of the fundamental properties and evolving nature of this emerging phenomenon.

In this article I discuss the core technology and solution components required

to generate ODC. I also examine the relationship between these building blocks and current collaborative enterprise requirements. Within this context, I suggest that service-oriented architectures can function as a crucial bridge between today's reality and a future full of ODC.

ODC: Components, Solutions, Implications

ODC is a concept based on distributed computing, utilizing the Internet as a universal platform. The on-demand delivery of IT resources combines two major capabilities commonly found in traditional utilities: ubiquitous availability and standardized components. In the case of the electrical grid, for example, power can be dynamically provisioned (i.e., it is ubiquitously available), and easily consumed by any device that conforms to voltage and form factor constraints (i.e., it is standardized).

Two important related concepts are grid computing, or “infrastructure on demand,” and utility computing, or “business processes on demand.” Both grid and utility com-

puting (and, by extension, ODC) are aided by a third technology component generally referred to as autonomic computing. Autonomic computing encapsulates a set of technologies, methodologies, and best practices enabling the self-management of computing resources. It not only provides service features that make existing IT operations more efficient, but also plays an integral role in ensuring quality of service and reliability in a networked environment.

The network-based provisioning of IT services can lead to tremendous economies of scale on the supply side. Although difficult to estimate at this time, the efficiency gains could well be enormous. Moreover, pay-as-you-go IT resources have the potential to fundamentally revolutionize how business is done. Electricity did not simply replace candles. It spawned numerous related innovations and inventions, utterly transforming our socio-cultural environment. Similarly, ODC has the potential to radically change our prevailing expectations of accomplishment and "best-practices" for both consumers and corporations. Not surprisingly, the benefits of ODC are expected to arise in the context of both top-line growth and cost containment.

Getting on the Bandwagon: Embrace the Network

The realization and delivery of on-demand provisioning of IT resources is inherently tied to the Internet. Even in the absence of full-blown ODC, enterprises are increasingly operating in a network-centric environment. Network-based technologies are driving more and more complex and highly integrated real-time collaborations among businesses. And many of the requirements an enterprise faces to become (Inter-) network enabled are also prerequisites for the adoption of on-demand models. In fact, ODC solutions extend the notion of network-based businesses via smart decoupling of the IT resources required to conduct business in a collaborative-networked environment. Hence, there is a certain overlap in technology and methodology adoption underlying both collaborative business and ODC solutions. It is this deliberate management of the network-centric overlap that allows enterprises to take advantage of incrementally available

resources without having to massively redesign their operations.

Three core areas, or meta-enablers, span the business and IT boundaries and are necessary catalysts for the successful adaptation to a network-centric (business) environment.

Universal Networking

The network is the core substructure for the majority of recent technology advances and business and IT initiatives. It is paramount in ODC. Of course, the "network" is the Internet. Through its low cost, pervasiveness and ubiquity, Internet technology has drastically changed the way we conduct and think about business. In fact, Internet-based connectivity has become so important that the network essentially is emerging as the de facto operating system. The TCP/IP stack is the platform for higher-level services, such as wireless-enabled data, transaction processing, Voice over IP, and so on.

Application and Integration Frameworks and Platforms

Transaction integration and processing are an essential tenet of network-centric enterprise collaboration. The very possibility of (distributed) transaction integration and processing has propelled the Internet to its status as a core enabler of enterprise IT. Unfortunately, the retrofitting of existing application and integration architectures has proven of limited success. These efforts are bound by inherent constraints with respect to scalability, extensibility, and reliability. Hence, a substantially new approach to the design and architecture of application and integration efforts is required. Although frameworks and tools suitable to the task at hand have recently emerged, the ensuing changes arising from revamped designs and architectures touch the very core of enterprise processes as well as IT operations and implementations.

Structured Information Provisioning

In addition to distributed transaction integration and processing, the availability of real-time information (e.g., operational data and business intelligence) is generally considered strategically important. Despite (or possibly, because of) an abundance of packaged and custom applications within

the enterprise, the availability of comprehensive, timely information is still an elusive goal. The rollout of often integrated applications is often outpaced by changes in actual business requirements, such as mergers and acquisitions.

These three "meta-enablers" together pose a formidable challenge that enterprises need to overcome in order to build collaborative and integrated value chains. They are also paramount to the adoption of ODC. Unfortunately, these enablers have associated with them nontrivial hurdles, making it challenging to attain the appropriate state. Hence, smart and comprehensive planning for today's business and IT requirements can lead directly, and in an incremental manner, to an environment suitable for incremental adoption of emerging ODC solutions.

Although a detailed discussion of the core technologies and methodologies enabling each of the "meta-enablers" (e.g., application, integration, and portal servers; messaging platforms; IP security; etc.) is a worthy and valuable endeavor, this article focuses on a more fundamental theme: the design and implementation of a complete delineation of process from underlying technology.

Top-Down Design, Bottom-Up Implementation

An integral part of an enterprise's ability to incrementally adapt to ODC solutions depends on its ability to manage its business logic in a way that is utterly independent from underlying infrastructure. That is, business logic must be network-aware but infrastructure agnostic. Similarly, data also must be decoupled from the applications and infrastructure. For example, self-contained and network-conscious business logic around a (generic) financial management function, or packaged application, allows an enterprise to easily purchase an on-demand service-level agreement, if available, for either the infrastructure (grid) or the complete business application (utility).

Despite the inherent appeal of the separation of business processes from infrastructure, the enabling of business and IT operations indigenous to a network environment is a nontrivial and highly transformational undertaking that requires a great deal of planning and time.

A Top-Down Framework for Decoupling Business Logic from IT Infrastructure

In order to accommodate technological and process change, IT design and architecture need to isolate process from infrastructure. Service-oriented architectures (SOA) are increasingly chosen as the guiding framework for this task. Specifically, SOAs are a family of architectures that govern the design, development, implementation, and management of distributed systems. In particular, an SOA specifies the publication, discovery, and invocation of software components residing anywhere on any network. Hence, an SOA framework enables the execution of transactional processes without any explicit dependence on the underlying infrastructure. Consequently, internal and external enterprise transaction integration, processing, and automation are attainable, enabling (near) real-time enterprise collaboration. Moreover, an atomic, SOA-based implementation of processes allows local and global changes in logic or infrastructure without requiring deep and extensive changes to the whole system.

Web Services Architectures (WSA) are SOAs specified to take advantage of the Internet in a loosely coupled, asynchronous manner. Add to this property an open-standards, XML underpinning, and you have in WSA a favorable and increasingly dominant choice to implement SOA. Hence, a WSA implements an SOA specification by means of mechanisms such as HTTP, FTP, and XML. Specific protocols, such as SOAP, and associated specifications such as WSDL and UDDI, are examples of WSAs incorporating many of the benefits of the Internet including, to a large degree, open standards – itself a necessary condition to ensure widespread interoperability and (relatively) low cost. Moreover, J2EE and .NET are the two major (and largely competing) platforms enabling the implementation of Web services.

Bottom-Up Implementations of Web Services-Based SOAs

Before we discuss specific guidelines governing the implementation of SOAs, and especially Web services, it is worthwhile to briefly address the issue of business logic. Within an organization, two fundamental types of “logic” are present and need decoupling: business process logic and integration logic. (Note: Lest it be overlooked, we

are addressing logical separation at the business level, not the systems design and architecture level.)

Business process logic constitutes the abstraction of outcome-related, functional steps. This generally involves enterprise-specific knowledge, industry best practices, and regulatory requirements. To date, business process logic is widely dispersed throughout IT (sub-) systems and often encapsulated in either custom or packaged applications. Moreover, with the onset of ERP and other integrated applications, significant aspects of business logic are implanted in the “custom configuration” and application functionality embedded in packaged applications, constituting an implicit, often poorly documented, and generally difficult-to-access pool of process and workflow.

Integration logic, on the other hand, is secondary logic in the sense that it performs valuable “choreography” to bridge decoupled functional components. In issuing paychecks, for example, integration mechanisms will often need to interoperate between financial systems, payroll, and possibly security and access mechanisms. Combining such islands of functionality into a seamless and transparent business function is precisely the role that integration logic plays.

A pivotal task in the preparation of an enterprise to take full advantage of network-centric technologies, which in turn requires cross-functional business process management, is the identification, encapsulation, and migration of dispersed logic into discrete software components. The resulting design and implementation must not only remain agnostic with respect to the underlying infrastructure platforms (software and hardware) but also be suitable to a transaction-oriented environment. Detangling highly complex process and integration workflows within and across the enterprise into standards-based service components is an integral part of the transformational efforts required to enable a network-centric enterprise. It should be noted that much of the work involved is highly technical and needs to be conducted within a guiding framework such as SOA.

On the establishment of an SOA framework and process, a set of guiding principles

can be adopted to implement business functionality in an incremental manner. This is best done on an opportunistic basis, driving problem-specific solution design, tools, technology selection, and implementation. For example, the migration of selected custom applications to a J2EE-based platform or the integration of multiple channels (e.g., Web, VoIP, etc.) within a CRM effort are suitable opportunities through which to begin the creation of service-oriented architecture and infrastructure islands. This migration can gradually propel the overall business and IT structure toward a network-based environment necessary to be able to take full advantage of ODC.

Specific “meta” guidelines that might be useful to incrementally transform enterprise business and IT toward a SOA-based structure are:

- **An enterprise must commit to the broad adoption of service-oriented architectures:** This commitment must include appropriate funding. Specifically, it seems useful to establish a “Fund The Future” budget that covers the incremental cost difference between an SOA-based implementation approach and an ostensibly cheaper “dead end” proprietary legacy solution. Early experiences suggest that the accrued benefits SOA-based systems carry over to new solution requirements essentially reduce the overhead incurred with each new project. Such budget allocations need to be booked against the time-delayed benefits from both collaborative and on-demand solution adoption.
- **Think globally and act locally:** Design, architect, and implement isolated (business) solutions against the overall SOA vision. Carefully monitor and model the additional resources required to attain this goal.
- **Rigorously embark on and reinforce an SOA-based component (re-) design and implementation strategy:** The SOA framework is not just another design alternative; it is built on and for the network.
- **Rigorously commit to and enforce SOA-based design and technology selections, such as open-standards and extensibility requirements:** Open standards ensure the interoperability required for service-oriented enterprise functionality.
- **Closely monitor the maturity of problem-**

specific Web services tools and (support) technologies: While SOA ought to be the guiding design principle for all IT activities going forward, Web services may not always be the best implementation approach due to the immaturity of some components. Consider alternative approaches that allow the development and implementation of decoupled, atomic, and service-oriented business logic to be easily accessible and extensible (e.g., J2EE EJB containers).

- **Thoroughly identify, categorize, and inventory business logic and functionality** for overarching, iterative redesign and normalization within and across problem domains.
- **Be prepared and committed to design processes that allow the revisiting of solution implementations** and to change closed-out projects to fit the vision requirements in an iterative manner.

Conclusion

On-Demand Computing, a service-oriented approach to the ubiquitous and per-

vasive provisioning of IT resources, will fundamentally change business and IT operations. The constellation of enabling technologies based on ubiquity, open standards, and Internet infrastructure make for a very compelling vision of the future. Moreover, ODC is composed of several building blocks such as grid, utility, and autonomic computing that allow the provisioning of interim on-demand solution capabilities which, in turn, can drive competitive positioning of enterprises.

Enterprises will have to engage in significant preparatory work to be able to adopt ODC solutions. They will need to satisfy the requirements of three meta-enablers of network-centric operations: universal networking capabilities, application and integration frameworks and platforms, and structured information provisioning. An underlying theme to all of these meta-enablers is the daunting task of decoupling process and integration logic from the underlying infrastructure.

I believe that service-oriented architectures are a suitable top-down framework to drive the development of network-enabled, cross-functional, atomic business process components. I also believe that Web services tools and technologies can already provide significant guidance and implementation opportunities to tackle specific business problems today while simultaneously driving the necessary transformational process in an incremental, bottom-up manner.

References

- "Technology Enabled Enterprise." IBM Business Consulting Services, IBM.
- Wang, Ko-Yang; Greenberg, Bob, et. al. "Enterprise of the Future White Paper, IBM Global Services." IBM.
- Borges, Bernhard (2002). "Navigating Web Services." *Web Services Journal*, October.
- Sutor, Bob. "Perspective: The five biggest myths about Web services." C|Net, November 26, 2002. (<http://news.com.com/2010-1071-971149.html?tag=lh>) ©

WebServices

.NET J2EE XML JOURNAL

LEARN WEB SERVICES. GET A NEW JOB !

SUBSCRIBE TODAY TO THE WORLD'S LEADING WEB SERVICES RESOURCE

Get Up to Speed with the Fourth Wave in Software Development

- Real-World Web Services: XML's Killer App!
- How to Use SOAP in the Enterprise
- Demystifying ebXML for success
- Authentication, Authorization, and Auditing
- BPM - Business Process Management
- Latest Information on Evolving Standards
- Vital technology insights from the nation's leading Technologists
- Industry Case Studies and Success Stories
- Making the Most of .NET

- Web Services Security
- How to Develop and Market Your Web Services
- EAI and Application Integration Tips
- The Marketplace: Tools, Engines, and Servers
- Integrating XML in a Web Services Environment
- Wireless: Enable Your WAP Projects and Build Wireless Applications with Web Services!
- Real-World UDDI
- Swing-Compliant Web Services
- and much, much more!



Only \$69.99 for 1 year (12 issues)*

* Newsstand price \$83.88 for 1 year

Subscribe online at www.wsj2.com or call 888 303-5252

*Offer subject to change without notice



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

Critical Decisions for Service-Oriented Architecture

Four keys to a common, interoperable foundation

Services-oriented architecture (SOA) is quickly emerging as a strong technical foundation for enterprise applications. My company, involved in the development of application architecture solutions, has seen a growing emphasis on SOA to meet the integration, process-level reuse, and flexibility requirements of the dynamic enterprise environment. While simple Web services have educated the market on services, SOA is the implementation that most enterprises are adopting to achieve their goals.

The W3C has a Web Services Architecture group that is working in this area. While the group is called Web Services Architecture, that term is used interchangeably with services-oriented architecture. The emergence of the W3C as a standards body working to bring some cohesiveness to this space can only be a good thing for the enterprise. By bringing clarity to the issues and requirements involved and then working to define standards, this will result in common interoperable foundations. This interoperability is key if the enterprise is to realize the long-term advantage of turning a heterogeneous IT environment into a corporate asset.

Four Key Decisions

With that backdrop, let's look at the four key architecture decisions that will lay the groundwork for the successful adoption of SOA. These key decisions are:

- Service Transport
- Service Protocol
- Service Interface
- Service Interaction Pattern

Those of you who are familiar with the OSI (Open Systems Interconnection) Reference Model will probably recognize the benefit of establishing common vernacular and standards for these decisions. Since history is a valuable tool for education, let's take a quick look at the OSI. Work on the OSI started in the 1970s with the goal of standardizing the way computers communicate so that they could communicate ubiquitously. Everyone seems to have heard at some point (usually from their college days) about the seven layers of the OSI model: physical, data link, network, transport, session, presentation, and application. These layers cover everything from the lowest level, how bits go over the actual network hardware, to how the application interfaces with the communication mechanism. Standardizing these layers is what allows hardware vendors to build Ethernet cards that interoperate and, somewhat more prominently, what allowed the Internet to become such a widespread success through the use of

TCP/IP and HTTP, which fits into the OSI model.

If services-oriented architecture is to share that same success, common answers must be made, or at the minimum it should be realized when answers conflict so that some interoperability can be established where necessary. Recognize also that these decisions build on each other, similar to the OSI layers. The transport is the lowest layer and the interaction pattern the highest layer.

Service Transport

The service transport is the mechanism by which the services will communicate with each other. The two choices that are most evident are direct communication using HTTP, or messaging-based communication using any of the available message-oriented middleware (MOM) vendors. In talking with companies that are currently adopting SOA, it is evident that the messaging transport provides the desired loose coupling of services with the robustness required of 24x7 enterprise applications. The HTTP transport is being used mainly for communication within the extended enterprise.


Service Protocol

The service protocol is the language the services will use when communicating with each other. SOAP is a standard that is emerging at the lowest level. But that is not sufficient for defining the services protocol – it is akin to two people who know only eighth-grade English attempting to discuss physics. Enterprises have the option of adopting higher-level standards from sources such as ebXML or RosettaNet, or creating their own service protocol. The best path to take depends on the integration touch points, where stan-



Author Bio

Walter Hurst is the founder and CTO of Wakesoft. He has been working with advanced technologies for over 10 years. Before founding Wakesoft, Walter was an independent consultant and used early versions of Wakesoft technology to deliver Internet solutions to clients. Walter received a B.S. in computer engineering from the University of Michigan.
w.hurst@wakesoft.com



dards may be more beneficial, and the internal needs of the enterprise, which may best be met with a custom protocol.

Service Interface

The service interface defines what the service looks like to those who want to use it. The desired service granularity defines the typical service interface, which is then customized for particular services. A rule to follow is that an API-based interface forces early binding while a document-based interface allows late binding. The document-based interface provides the desired flexibility and loose coupling of services. One pitfall to watch out for is fine-grained interfaces that lead toward too much interservice communication. This introduces unnecessary overhead that will impact performance and scalability and also leads to tight coupling of services.

The service interface should also be definable outside of the implementation. Supporting service definitions is important for separating the implementation of a service (called a service provider) from the user of the service (called a service consumer). This service definition is often called a *service contract*. The service definition is a very important contributor to the workings of the service interaction.

Service Interaction Pattern

The service interaction pattern defines the landscape for how the services work together. The interaction pattern determines the controller of the service interaction and how much the services know about each other. Central-dispatcher and service-to-service workflow are two of the leading options for controlling service interaction. The service definition that was mentioned earlier plays an important role in determining the interaction by allowing a service seeker to find a service provider. Since the service provider may change, the service definition is the stable point for linking a service consumer and provider.

Another powerful concept employed in

service interactions is *service leasing*. Service leasing refers to the concept of providing the service consumer with a handle to a service provider, with the caveat that the service provider handle is only good for some period of time (or possibly until some external event or condition takes place), after which the service provider handle expires. Once the service provider handle expires, the service consumer would once again have to find a service provider. A useful analogy is DHCP (Dynamic Host Configuration Protocol), which is used for controlling IP address allocation (along with other networking configuration). In the old days of networking, every computer had its own IP address, which had to be allocated and tracked manually by the IT staff. As computers were added and removed, the IP address tracking could become quite an administrative task. Enter DHCP, which allows a central server to allocate IP addresses to client computers using a leasing mechanism. As computers join a network, they receive an IP address that is good for some amount of time. Once it expires, the DHCP server gets the address back and can reallocate it. There are no wasted IP addresses given to machines that have been removed from the network and if there are network changes that need to be made, the server will propagate them when the clients release their IP address.

Answering these four critical questions will form the services-oriented architecture environment. These architecture decisions must be made in a consolidated effort in order to realize the full benefits of the SOA. Because of their interdependency and impact on each other, they cannot be made independently. While this will form the enterprise standard for typical services, it should not be constraining. Some services may deviate because of their unique requirements but those situations should be closely examined and noted. Referring to the adoption of the OSI Model, it can be seen that the more layers that have common answers (i.e., using TCP/IP vs Novell's IPX), the more interoperability will be possible, with the nice side effect of lower maintenance and administration costs.

Conclusion

While these four decisions lay the groundwork for an SOA, as its adoption in the enterprise expands and becomes more advanced there will be application-level requirements that need to be addressed. The solutions to the application requirements will be built on the SOA layers we just discussed. Some requirements that are likely to be encountered are session management, context sharing, security, and transactions. Session management is important if a service gets many invocations from the same consumer; how can the service become part of that session of communications without the consumer having to repeatedly send the same information. Context sharing streamlines the support for many services to work together to fulfill a business function. Because the behavior of the services may vary based on certain environment conditions (such as user, request, or session variables), these services need access to that information. In lieu of the service consumer proactively sending all possible environment variables, it would be nice if the service provider could access it if needed. Security and transactions are more straightforward because they are traditional application concerns: how do you secure services so that you have control over who can access them and how do you maintain data integrity when many services may be involved in processing business functionality? SOA is evolving to address these requirements.

While it may seem that there is a lot to grapple with when adopting an SOA, don't be intimidated. Education is the first step – understand what the factors are for laying a successful SOA foundation. If the decision is made to adopt an SOA approach for constructing your business assets, you will encounter many build versus buy decisions.

Remember to trust your experience and intuitions. SOA is a big step in the evolution of distributed computing but it is not a brand new invention or a silver bullet. ©

Operational Management: A Web Service Barrier-to-Entry

The need for service-centric management solutions



Web services represent a quantum leap forward towards achieving not only interenterprise application integration, but also cross-enterprise application interoperability. These services will enhance operational efficiency and agility, and unlock the intrinsic value of the company's goods and services, thereby creating new market opportunities.

AUTHOR BIO:



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

However, as organizations develop initiatives to electronically expose their core competencies and express them as Web services, they are confronting some specific challenges. The first is that most organizations may not have considered the need for a paradigm shift in their IT departments toward a more e-business service-delivery approach. The second is that current management tools look only at technology- and component-based metrics apart from the concept of quality e-business service delivery. Both of these challenges make it difficult to successfully implement Web services in an organization.

Defining "Service"

To frame the discussion, let's first define "service": a) the occupation or function of serving, b) a facility of supplying some public demand, and c) contribution to the welfare of others. Web services fulfill public demand by using technology to facilitate consumer interactions (consumers can be internal or external to the organization). That is, they act as an abstraction between service consumers and the application and

infrastructure components which provision them. Web services synthesize technology into electronic business deliverables – electronic manifestations of business processes. They distill the interactions of service consumers into single points of electronic contact. A Web service can be further defined not only by the value it exposes but by its repeatability and reliability.

As an example, let's suppose there is a Web service that makes it possible for brokerage firms to inquire into a customer's credit history. The value and viability of this Web service could be described not only by the information it exposes, but by the timeliness of the response to the brokerage's request, the accuracy of the information, and whether the service can consistently facilitate the needs of the consumer. Essentially, a Web service that provides all the required information quickly and consistently over time, defines the service itself. Therefore, implementing a Web service requires more than just the technical task of developing and provisioning it – it also requires adhering to standards of consistent performance and

reliability. Another way of saying this is that it is as important to implement an operational environment of high-quality, consistent execution around Web services as it is to create the services themselves.

Essentially, Web services represent the dividing line between infrastructure (the domain of IT management) and consumers (the domain of business line management). From this, we can draw two important conclusions. First, Web service quality is the single biggest factor in determining customer satisfaction, and thus the most meaningful target for managing and controlling service levels. And second, from an operational perspective, the necessary alignment of business and IT management can only occur if both are focused on common service delivery objectives.

Shifting IT to a Business Focus Using Service-Level Management

It's clear that Web services and their ensuing deployments will affect not only development staff, but also IT operations and business line managers. Web services

International Web Services Conference & Expo

Web Services Edge 2003

web services **EDGE**

CONNECTING THE ENTERPRISE WITH WEB SERVICES, JAVA, XML, AND .NET

March 18-20, 2003
Boston, MA

Featured technologies and topics will include:

- ▶ Interoperability
- ▶ Enterprise Networks
- ▶ Securing Web Services
- ▶ Integrating existing networks
- ▶ Leveraging your existing software
- ▶ Real Time Web Services
- ▶ Where and When Should I Use Web Services?
- ▶ Web Services: From Consumption to Publication
- ▶ UDDI

Event Sponsors:



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

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

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

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

Contact information: U.S. Events: 201 802-3069 or e-mail grisha@sys-con.com • European & Asian Events: 011 44 208 232 1600



Boston
March 18-20

London
June 3-5

Berlin
June 24-26

Hong Kong
Coming soon...



Conference program available online!
www.sys-con.com/webservices2003east

will require an institutional shift that involves new service-centric processes, methods, and tools that can be shared effectively across these functional areas to manage and assure service quality.

While most development shops have figured out how to technically leverage Web services, few have considered the impact these initiatives will have on those parts of the organization that are chartered with maintaining them. For Web services to be successful in their ability to provide application service interoperability inside and outside the organization, there must be a concomitant shift in the IT operations domain away from technology-focused operations toward business-focused operations. New software solutions to enable this change are in order.

Traditionally, the coordination of activities between IT development, operations, and lines of business have been difficult and a barrier to the "agility" that everyone speaks about. The proliferation of Web services will only exacerbate this problem as business line organizations request more Web services and demand accountability, while IT organizations continue to be resource constrained.

The good news is that Web services will act as a catalyst for institutional change and inevitably forge a closer bond between IT and business. IT organizations will need to engage in a new form of management, namely, defining and managing services to strict business-centric guidelines. They will need to manage and monitor technology to the objectives of business. In practice, service-level agreements are the vehicle for accomplishing this. They will be used to describe the metrics of quality in accordance with business objectives for the service. They will also act as an enabler to transition IT from technology-centricity to more business-focused operations.

Additionally, as Web services are implemented and become commonplace, IT operations will be tasked with not only managing their own organization's Web services, but also external Web services which their own services rely on. Service-level agreements (SLAs) between organizations will play a vital role. Clearly, this implies that standards and methods for defining SLAs will need to be created.

These standards would then be used between IT and business lines internally, and between organizations consuming each other's Web services. Without such service level agreements between departments and organizations for assuring some degree of service quality, it is difficult to imagine a world of interdependent and business-critical Web services.

With that said, SLAs do exist today but are generally technology-oriented. They are usually more focused on network service availability than they are on describing and measuring service quality against business standards. Future SLAs must be defined using both business and technology performance metrics, since both characteristics are required to fully describe the health and viability of Web services.

One way of shifting IT operations to a more service-centric model is to empower them with tools that translate the language of infrastructure performance monitoring into business impact and service-level management. This necessary generation of software solutions will change the way organizations look at application and infrastructure performance by tying it directly to the business services they provision.

Why Traditional Management Tools Don't Work for Web Services

Current management tools and approaches are fundamentally incompatible with the Web services paradigm because traditional tools manage infrastructure and application metrics divorced from the key performance indicators that define service quality. A typical management tool will look at performance metrics for specific tiers of the complex application architectures – database, Web server, application server, or network. They can drill deeply into what might be wrong with a specific component, but can't correlate that problem with the overall health and performance of the Web service that relies on it. This makes it nearly impossible for an IT organization to understand the relationship or impact of technology to service delivery.

Web services-enabled applications will further challenge traditional application management tools. While many Web appli-

cations and their associated architectures are n-tiered and inherently dynamic in nature, traditional management tools were designed to manage more monolithic and static computing environments. Traditional management tools don't detect problems, they just report raw data as most approaches use non-intelligent static thresholds as a mechanism for detecting and reporting problems. They also don't attempt to monitor applications and services holistically across all related tiers. These approaches just won't work for managing today's multi-tiered Web applications and services, since it makes no sense to manage a dynamic environment with static thresholds. It is also inappropriate to manage applications component-by-component without regard to how these components work together to deliver services.

The result of not looking across tiers and relying on static thresholds is a noisy and inaccurate management system, which provides numerous false alarms and notifications that might not be relevant to any service context. Thousands of superfluous alarms lead to either an apathetic approach to operations management or to an intensely reactive, finger-pointing, and firefighting approach. Remember what happened the last time you asked someone to explain why an application was running slow?

So a major shortcoming of traditional management tools is their inability to get to the root cause of a problem. Because they focus on monitoring the individual health of components, they are not looking at the health of the entire e-business process. The individual components could look fine, but the overall process could still be broken. This inevitably leads to guesswork and finger-pointing as IT personnel shift blame and avoid responsibility.

The lack of insight into the root cause of problems can lead to over-expenditure on redundant resources just to ensure that the system runs. Without an understanding of where breakdowns and failures occur, redundancy is the insurance policy. Unfortunately, excessive redundancy further contributes to the complexity of the infrastructure and is not usually a good long-term solution.

There are many documented cases where organizations attempted to retrofit

legacy management approaches to Web applications and services, and the results were inevitably disappointing, even disastrous. This is solely because traditional management tools and their approaches are based on solving the wrong problems, assuming of course that your ultimate goal is to manage service quality.

The New Generation of Management Tools

To provide meaningful management of Web services, operations staff and business line managers alike require a new breed of evolved management solutions that understand the intricate dependencies and interactions of application components to deliver Web services. This way, management functions are performed in accordance with service-quality objectives instead of component-by-component without regard to service context.

These new service-centric management solutions must monitor the behavior of Web services from the top down instead of the

bottom up. They must also incorporate root-cause-analysis capabilities so that Web services and their constituent parts can be monitored as cohesive and federated entities. They must, in essence, manage infrastructure health and performance from the perspective of Web service quality. Anything less and the result will likely be a reactive finger-pointing exercise with excessive operational costs to IT and opportunity costs to business.

New service-centric management solutions will proactively assist operations personnel to answer tough questions in real time. How is a performance anomaly affecting service delivery relative to committed service levels? Which business processes will be affected? These questions cannot be answered without understanding the relationships between infrastructure components and the Web services they are facilitating. This understanding will promote efficient operations and the avoidance of costly, unnecessary infrastructure expenditures.

These new tools will also make it possible to realize the paradigm shift needed in IT operations by enabling IT staff to adopt methods and procedures that are focused on service quality. Instead of measuring network availability, or throughput, without service context, IT will interpret these downstream infrastructure metrics within the context of impact on business service objectives. IT will finally be able to align itself to business using service level agreements as the lingua franca. It will also be able to use this mechanism for managing the performance and viability of externally provisioned Web services.

The true test of Web service-oriented management tools will be their ability to intelligently correlate application and infrastructure health and performance information to Web service performance, in real time. The rate of success, speed, and adoption of Web services will be directly tied to organizations' adoption of tools to effectively manage them. ©

ATTN: Developers

STEP UP
to the mike
and be...

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

HEARD!

**Calling Sleek
Geeks Everywhere!**

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

i-Technology
News

i-Technology
Views

i-Technology
Comment

i-Technology
Debate

**DEVELOPER
SYS-CON
.COM**

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

SYS-CON
MEDIA

Reaching Back to Unlock Legacy Systems

Finding the secrets that already exist

Written by James Phillips and Brian Anderson

As companies have viewed new technologies historically, the smart enterprise adopted new technologies while improving on its existing technology investments. Enterprises that embrace this methodology can drive more value out of both the new and the old technology. Why spend millions on buying and implementing a new system if you can repurpose your existing applications and data to deliver the same benefits of a new system? Today, companies are finding they can get more life out of mainframe legacy systems by exposing valuable business logic as Web services. Web services are allowing legacy applications to be integrated seamlessly throughout the enterprise.

"Many existing mainframe organizations are reevaluating the value proposition of this platform," said Dale Vecchio, research director at Gartner. "The ability to leverage existing systems in a services-oriented way requires technology to easily encapsulate these systems as services, and then manage them with the same quality-of-services expectations."

Web services offer a solution to the classic application development quandary – how to universally integrate disparate programs spanning programming methodologies, languages, and computing platforms. Web services allow the enterprise to securely expose core business logic both inside and outside the firewall. As an enabler for application-to-application interaction, Web services have no equal. No other technology has delivered on this "holy grail" of universal application integration.

The immediate impact of Web services will be in unlocking the hidden treasure of the Global 2000 – a multi-decade investment in what is now termed legacy-based applications. These applications, residing on the IBM mainframe and mid-range platforms, were written in COBOL and RPG.

AUTHOR BIOS:



Brian Anderson is the senior product marketing manager at ClientSoft (www.clientsoft.com), a leader in delivering fast and secure integrated applications to legacy systems. He is responsible for ClientSoft's product marketing and management functions. Before joining ClientSoft, Anderson worked in product marketing for Jacada. Anderson received a BS in geography and computer science from Florida State University and pursued graduate work in marketing at the University of Notre Dame. BANDERSON@CLIENTSOFT.COM



James Phillips is senior vice president of marketing and product management at Actional. He has worldwide responsibility for Actional's product and market strategy and market execution. In more than 17 years of software industry experience, James has held senior marketing management, software engineering, and business development roles with Intel, Intuit, Synopsys and Central Point Software. He is a frequent speaker and editorial contributor on Web services-related issues and serves on the international advisory board for Web Services Journal. JAMESP@ACTIONAL.COM

They may be cumbersome to use, provide dated user interfaces, and appear unable to easily integrate with new technologies. Legacy applications are all this and more. But more importantly, they are also finely tuned repositories of corporate business logic and data. These applications have evolved with the needs of the business over the last 30 years and cannot be replaced easily. A decision to buy a multimillion dollar packaged application or to invest in a costly rewrite will not guarantee the successful duplication of these applications. The simple fact is, these applications are the lifeblood of business computing and are not going away in the near future.

Web services provide the key to integrating the legacy applications that still power the Global 2000 with any new application or technology. Web services from legacy applications, partnered with a Web services management solution, deliver seamless integration throughout the enterprise. Let's explore the seemingly disconnected worlds of mainframe applications and Web services solutions.

The Mainframe Is Alive and Kicking

Enterprises have invested much time and money in legacy systems. According to industry analysts, over 70% of the world's data is contained in legacy systems. IBM states that there are more transactions processed by CICS systems today than by the entire Internet. CICS handles more than thirty billion transactions per day, and is used by 492 of the Fortune 500. Nine out of ten ATM transactions are done using COBOL. IT departments in large corporations manage over 30,000 CICS code modules written in COBOL, Assembler, or PL/I. The oldest modules are 25–30 years old, but still process mission-critical data every day and almost certainly will continue to run for years to come. You get the idea – legacy systems still have an important role to play in business today.

This enormous amount of code, running on mainframe and AS/400 systems, is collectively known as “legacy applications.” Between 70 and 80% of IT budgets are spent yearly on maintaining and evolving this code.

Many corporate IT departments have turned to Windows NT and Unix-based machines as a complement to their existing

mainframes for developing and deploying Web and composite applications. The mission-critical legacy applications residing on the mainframe must be leveraged to provide both data and business logic to power these new Web and composite applications. It is too costly to simply recode legacy applications in languages designed for these new platforms.

Legacy application integration is the solution. Existing legacy applications must be made available to these new applications. Corporate IT departments must be able to isolate specific business transactions from within one or more applications and expose it to the Web or create composite applications. Web services is the technology powering this new form of application integration.

How to Unlock Legacy Applications

The difficulty in integrating legacy mainframe applications stems from how they were written. The vast majority of legacy applications were designed with the business logic and presentation layer tied together. This prevents several problems for programmers attempting to integrate these applications with new development projects. Let's explore the common approaches used for access to legacy application data.

Two primary methods have been used:

1. Direct database access
2. Emulation-based access

Direct database calls to the raw legacy data sound like the ideal solution. They deliver lightning-quick response times, and allow the developer to repackage the data upon retrieval in any format needed. Unfortunately, all is not as it seems! Here's why: the original developers of legacy systems were faced with a shortage of hard-drive storage space. To get around the tremendous costs of storage on the mainframe, most legacy databases were stored in various compressed and very confusing formats. Additionally, by accessing the data directly, you lose the real benefit of most legacy applications – the business logic. Remember, the business logic is tied into the user interface layer, so by going to the raw data you lose the ability to decipher the meaning of the raw data. You also run the risk of corrupting the data when writing back to the legacy database, since many

applications have data that is updated from batch (or offline) applications.

Emulation-based access has been the solution of choice for most developers who need to integrate with mainframe legacy applications. Emulation-based access, commonly referred to as “screen-scraping,” is the process of taking the data from the screens of a legacy program that was formatted for a “green-screen” terminal and reformatting the data for use in a new user interface. The input from the new user interface must also be reformatted to allow the underlying legacy program to understand the request.

Screen-scraping products range widely in their degree of sophistication and scalability. Low-end products exist that simply use the geometry of the screen (row and column) to provide a new presentation or programmatic interface in a two-tier deployment. They do not allow any reengineering of the legacy business logic to allow for more complex integration scenarios. Several vendors have released products introducing the concept of a mid-tier application server to the screen-scraping market. These vendors offer high-end legacy integration solutions with automated development environments that leverage the capabilities of a middle-tier legacy application server to provide elaborate reengineering or “intelligence” around the navigation of the legacy screens. These tools are based on terminal emulation, but provide many additional features designed to improve both the development and deployment of applications based on legacy integration.

Emulation-based legacy integration tools have proven valuable in application integration projects. Enterprises have used tools such as the ClientSoft ClientBuilder product line to unlock legacy applications and expose them as Java or Microsoft components, and now as Web services. These components or services can then be integrated into new J2EE applications and other packaged applications, such as a Siebel CRM application or a new portal application.

While emulation-based legacy integration tools have been the only sure-fire way to access legacy transactions they do have some considerations. The performance and scalability of the new interface is a primary concern. The underlying use of terminal

emulation is a limiting factor of application performance and scalability. The datastream can only handle 1920 characters of data at any one time, based upon the 80x24 size of a legacy host screen. This fundamentally limits how much information can be passed to and from the legacy application via terminal emulation. A second consideration is the "dual maintenance burden". The geometry of the host screen requires programmers to engage in manual row-and-column based development projects that fail if the position of the fields on the underlying host screens is altered. This forces application developers to modify the emulation-based interface any time a change is made to the underlying legacy application.

While emulation-based legacy integration solutions work, there is a better way to accomplish this integration without any of the concerns discussed above – a direct integration approach. Direct integration to the host applications provides a higher-performing integration solution and increased stability. Enterprises want to maximize the performance of their legacy integration solutions and remove some of the weaknesses associated with emulation-based approaches.

Vendors that offer a direct integration solution include ClientSoft, HostBridge Technology, InnerAccess Technologies, and SoftTouch Systems. Each of these ven-

dors offers a two-tier solution, with a runtime that resides on the mainframe and a generated client on the front end. This client can be an HTML GUI, a JavaBean or Enterprise JavaBean, or a Web service. ClientSoft and InnerAccess Technologies offer direct integration solutions coupled with a Web service generation option. Let's examine how these vendors accomplish direct access legacy integration.

- **The development environment is Windows-based:** This environment allows direct interaction with the mainframe CICS and IMS applications through one of the following APIs – COMMAREA, FEPI, or 3270 Bridge – not via terminal emulation. The use of each of these APIs will vary depending on the vendor's solution and the backend host type.
- **Runtime component resides on the mainframe inside of a CICS region:** This allows the direct access solution to interact with the transactions using direct APs and not emulation. There is no middle-tier requirement – just the mainframe-based runtime and the component or Web service.
- **The products provide a generated interface, whether presentation or programmatic:** HostBridge provides XML from legacy applications. Soft-Touch Systems provides a browser-based GUI application from legacy

applications. ClientSoft and InnerAccess Technologies focus on programmatic integration, offering components and Web services that can be used stand-alone or in an application server requirement with WebSphere or another J2EE application server.

Direct mainframe integration solutions remove all of the traditional considerations associated with typical emulation-based approaches. A direct legacy integration approach delivers substantial performance gains coupled with a lower risk of application failure at runtime. The combination of direct legacy integration and Web services provides seamless access to the hidden treasures of the corporate data center – legacy applications.

Introducing Web Services

Web services are not mysterious. According to Gartner, Web services are "loosely-coupled software components delivered over Internet-standard technologies."

The challenge is in how to deliver Web services from legacy applications. Several ISV products have emerged to allow the encapsulation of legacy business logic as a Web service. Integration vendors ClientSoft, Jacada, Seagull, and WRQ all offer Web services products that utilize a terminal emulation approach to legacy access. These tools are subject to the considerations detailed above. Both ClientSoft and InnerAccess offer legacy integration products that do not utilize terminal emulation, and provide automatic delivery of Web services from mainframe legacy applications. Let's examine one of these solutions further.

ClientSoft Tanit Objects (CTO) provides a graphical Windows-based development environment. This development environment allows developers, whether their skill-sets are COBOL or another modern language, to utilize three different APIs to connect to mainframe applications:

- DPL
- 3270 Bridge
- FEPI

The choice of API used to connect to the legacy system is based on the type of

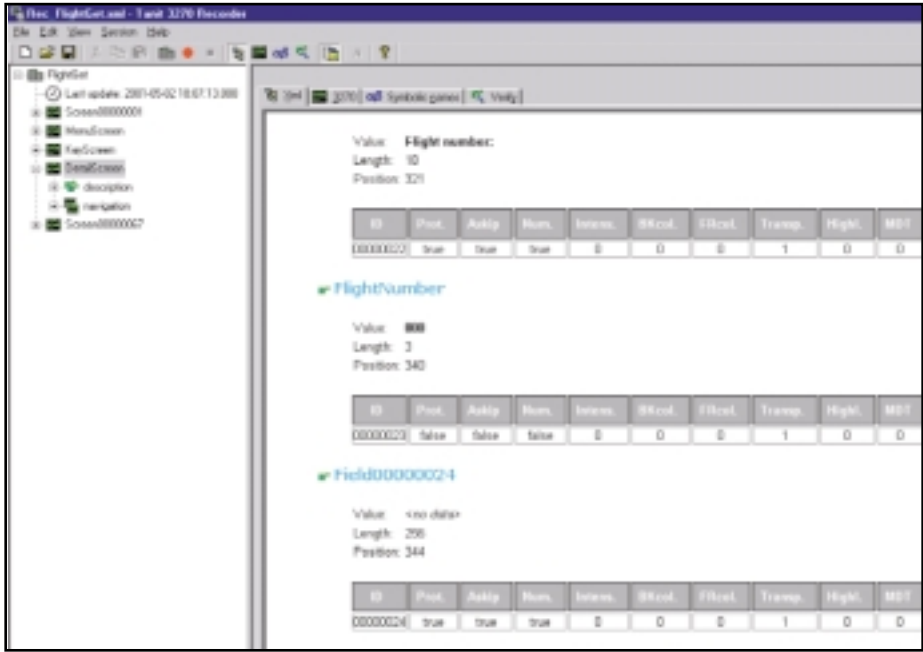


FIGURE 1 | Defining and input and output parameters

legacy application. Visual applications, those applications that traditionally utilize a 3270-based presentation layer, are accessed via the 3270 Bridge or FEPI APIs. Nonvisual applications, those applications that are modularized and don't have the presentation layer and business logic integrated, are accessed via the DPL API using the COMMAREA. The important thing here is that you do not need to have mainframe expertise with these APIs.

The next step is to utilize the tool to isolate the business transaction you would like to expose as a Web service. When working with visual, 3270-based applications the iso-

lation of the business transaction is done with the 3270 Recorder/Mapper. This simulates "green-screen" interaction with the legacy application and allows the developer to easily define the entire unit of work involved in the relevant business transaction and define the input and output parameters for the Web service (see Figure 1). When working with nonvisual modularized applications, the developer uses the Tanit Parser to automatically parse the COBOL copybook and then define the entire unit of work with the appropriate input and output parameters describing the complete business process.

The final step is to use the Tanit Generator (see Figure 2) to have the Web services automatically generated from the development environment.

Tanit Objects will generate the complete WSDL (Web Services Description Language) file based on the series of Input and Outputs defined through either the Parser or Recorder/Mapper. Additionally, it will generate a "starter program" which allows the developer to test the newly created Web service. The generated Web service resides on the mainframe, taking advantage of the inherent security and performance of this platform.

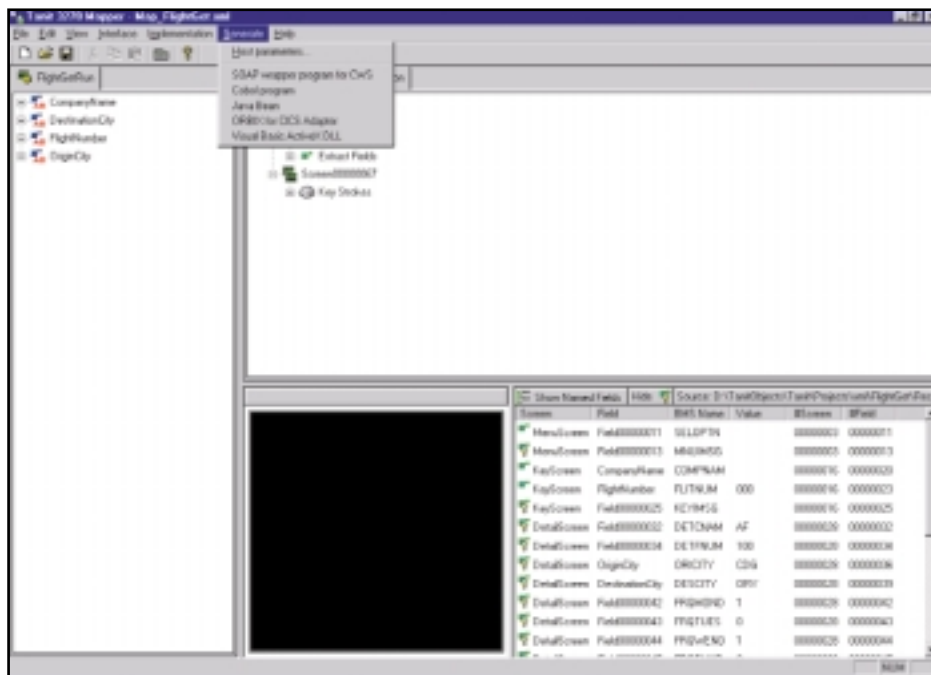


FIGURE 2 | Web services are automatically generated from the development environment

Managing Your Way to Web Services Success

Web services technology promises to finally usher in a world of service-oriented integration, thereby revolutionizing the business of application integration both inside and across enterprise boundaries. And while the technology behind Web services standards is interesting, technologies with similar purposes have come and gone in the past in the form of CORBA, DCE, and DCOM. It is the nearly universal vendor support for Web services standards that will finally make this promise a reality.

Native support for Web services-based connectivity is emerging in packaged enterprise applications, application servers, portal solutions, and application development tools. The leading solutions in each of these software categories today offer the native ability to connect with other software systems via Web services standards. Legacy integration vendors make the mainframe systems that continue to serve as the backbone of many enterprise-computing environments available to these newer software systems.

As more software systems are made available for integration via Web services standards, more connections based on these standards will be built. But left unmanaged, the widespread leverage of Web services can lead to massive complexity and cost. Figure 3 illustrates the exploding set of interconnections that can exist between systems connected via Web services standards.

Without a sound and uniform management strategy, this growing set of

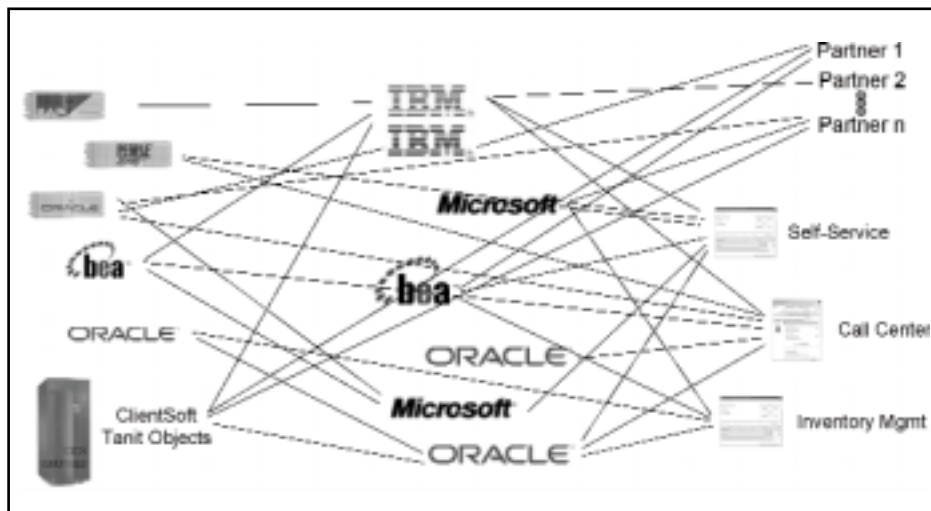


FIGURE 3 | Interconnections that can exist between systems

direct (point-to-point) connections between Web service providers and consumers can create insurmountable problems for an enterprise including:

- **Complex “point-to-point” network of interconnected systems leads to exploding (n²) costs and risks of system-wide failure:**
 - Security risks: Security configuration is duplicated at each provider system
 - complex, coarse-grained and error prone
 - Multiple points of Web services control: Access control, monitoring, interoperability support – solving the same problem many times
 - Lack of administrative consistency between various provider systems
- **Impaired visibility of Web services activity and availability leads to downtime, frustration, and security risks:**
 - Difficulty detecting failures and troubleshooting
 - No monitoring of service availability and quality
 - Hard to aggregate and audit historical Web service usage information for analysis and potential billing
 - Missed opportunity to monitor business activity
- **Interoperability challenges narrow the reach of Web services:**
 - Varying schemas/semantics between application systems: “My purchase order system generates purchase orders in this format, can you accept that?”
 - Varying standards interpretations and implementations
- **Noninsulated, brittle direct connections between consumers and providers impedes**

“Web services allow the enterprise to securely expose core business logic both inside and outside the firewall”

flexibility and limits business agility:

- Upgrades or changes to Web service providers impacts downstream consumers
- Expensive one-off strategies to support migration

These costs tend to increase in a non-linear fashion as each new system begins to depend, directly or indirectly, on a growing set of existing systems. Eventually the number of connections can cause the system at large to collapse under its own weight. Skyrocketing costs mean that beyond a point, each new system leveraging a Web service becomes unprofitable to support. Figure 4 illustrates this concept.

If Web services are to usher in an era of truly universal enterprise application connectivity, then management of these environments must be a consideration from day one.

Recognizing these issues, legacy integration vendors are partnering with Web services management companies to ensure that customers reap the substantial benefits of Web services while avoiding the pitfalls of unmanaged Web service deployments.

Legacy integrators should look for a Web services management provider that offers a platform-neutral, unified solution for Web services monitoring, life cycle management, interoperability, traffic management, and security policy enforcement. In addition, the solution should allow IT organizations to:

- Substantially mitigate security risks through fine-grained Web service access control
- Gain end-to-end visibility of Web services activity and identify root cause of Web services failures, thereby maintaining uptime
- Manage Web services through their entire life cycle without disrupting the operation of systems that grow to rely on those services
- Enhance the value of Web services by cost-effectively growing their potential user community
- Dramatically lower management costs
- Enforce, monitor, and control service availability, Quality of Service (QoS), and Service Level Agreements (SLAs)
- Scale services to meet the needs of demanding and varying sets of consumers
- Monitor business activity as represented by Web services interactions
- Audit the use of Web services in support of development planning, billing, charge backs, and capacity planning

Managed Legacy Access – A Natural First Step

There is substantial momentum behind the push toward Web services as the architectural underpinning of any robust enterprise application integration strategy. A natural first step in the adoption of this strategy is to unlock the systems and assets already in place in the enterprise. Legacy integration vendors make it possible to extend the life of existing main-frame-based applications by unlocking these assets as Web services.

But before embarking on a Web services-based initiative, strong consideration must be given to how these critical business services will be managed. The foundation for success lies in providing a comprehensive system for the enterprise-wide management of Web services. ©

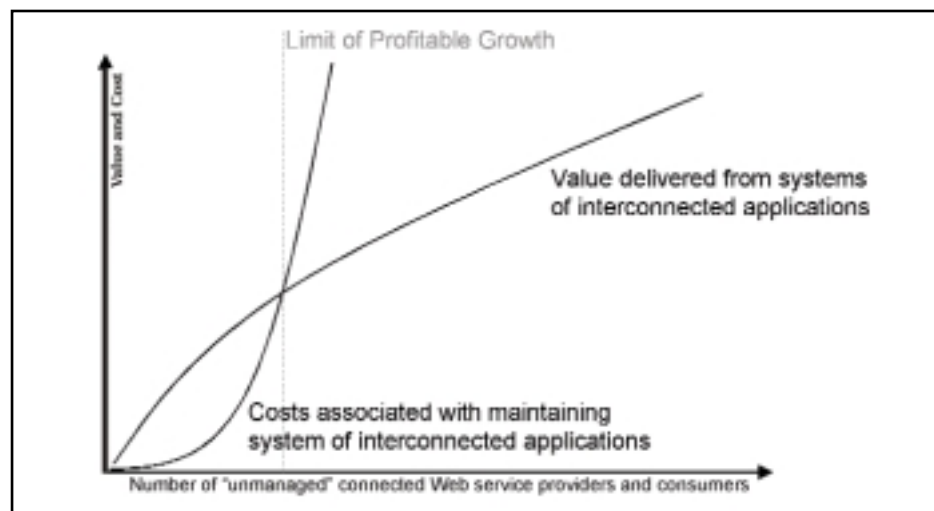


FIGURE 4 | Results of increased costs

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

Boston
2003

London
2003

Berlin
2003

Hong Kong
2003



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

3rd Annual

EAST

Web Services Edge Conference & Exposition



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



March 18-20, 2003
Hynes Convention Center
Boston

**Call TODAY
TO REGISTER**
201-802-3058
[www.sys-con.com/
web-services-edge2003-
east/registernew.cfm](http://www.sys-con.com/web-services-edge2003-east/registernew.cfm)

Special Insert: Web Services Edge East Conference & Expo
Featuring FREE Tutorials, Training Sessions, Case Studies and Exposition

MEDIA SPONSORS



OWNED BY
SYS-CON MEDIA

PRODUCED BY
SYS-CON EVENTS

Special Insert: Web Services Edge East Conference & Expo

Featuring FREE Tutorials, Training Sessions, Case Studies and Exposition



We are pleased

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

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

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



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

All the best,
SYS-CON Events

Features & Attractions

TO ALL CONFERENCE & EXPO REGISTRANTS

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

Keynotes and Highlighted Speakers

Dave Chappell

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



Eric Newcomer

Chief Technology Officer, IONA

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



Simon Phipps

Chief Technology Evangelist, Sun Microsystems

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



John Magee

Vice President, Oracle9i Oracle

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



Event Sponsors

ORACLE

Microsoft



ALTOVA

Rational
the e-development company

COMPUWARE



SUN MICROSYSTEMS Java™ University Program

Web Services Programming Using Java™ Technology and XML

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

Who Should Attend

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

Prerequisites:

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

Overview:

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

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

Benefits

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

Outline

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

Java™ 2 Platform: Architect Certification Fast Path

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

Who Should Attend

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

Prerequisites

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

Overview

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

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

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

Benefits

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

Outline

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

Java™ 2 Platform: Programmer Certification Fast Path

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

Who Should Attend

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

Prerequisites

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

Overview

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

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

Benefits

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

Outline

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



XML Certified Developer *Fast Path*

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

Audience

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

Prerequisites

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

Overview

XML is the foundation of two important emerging

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

Outline

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

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

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

Special Insert: Web Services Edge East Conference & Expo

Featuring FREE Tutorials, Training Sessions, Case Studies and Exposition

Conference at-a-Glance

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

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

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

Conference Overview

Java Technology Track



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

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

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

Microsoft .NET Track



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

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

Web Services Track



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

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

XML Technology Track



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

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

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

Featuring FREE Tutorials, Training Sessions, Case Studies and Exposition

Special Insert: Web Services Edge East Conference & Expo

Microsoft® FREE .NET Web Services Tutorial

Russ' Tool Shed

Wednesday, March 19, 2003

9:00 a.m. – 5:00 p.m.

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

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

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

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

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

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

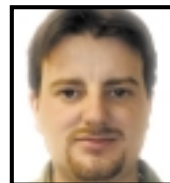
Microsoft .NET Remoting is the .NET technology that allows you to easily and quickly build distributed applications. All of the application components can be on one computer or they can be on multiple computers around the world. .NET Remoting allows client applications to use objects in other processes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

To learn more, visit
[www.sys-con.com/
webervicesedge2003](http://www.sys-con.com/webervicesedge2003)

Advisory Board



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



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



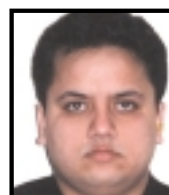
Derek Ferguson
Editor in Chief, .NET
Developers Journal
Chief Technology Evangelist,
Expand Beyond Corp.



Bob Familiar
.NET Architect,
Microsoft
New England



Thomas Robbins
Senior Technology Specialist,
Microsoft
New England



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



J.P. Morgenthal
Chief Services Architect,
Software AG

Hotel Arrangements Are Easier Than Ever!

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

SPECIAL RATES

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

RESERVATIONS

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

REGISTRATION FORM

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

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

THREE WAYS TO REGISTER FOR CONFERENCE

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

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

Please complete sections 1, 2, 3 and 4

YOUR INFORMATION (Please Print)

☐ Mr. ☐ Ms.

1 **First Name** _____ **Last Name** _____
Title _____
Company _____
Street _____
Mail Stop _____
City _____
State _____ **Zip** _____ **Country** _____
Phone _____
Fax _____ **E-Mail** _____

PAYMENT METHOD: (Payment in full due with registration)

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

Check # _____ **Amount of Check \$** _____

Charge my ☐ Visa ☐ MasterCard ☐ American Express ☐ Discover

Name on card _____

Card # _____ **Exp. Date** _____

Signature _____

Billing Address (if different from mailing address) _____

3 PLEASE INDICATE YOUR CONFERENCE CHOICE:

Total Registration fee \$ _____

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

4

A. Your Job Title

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

B. Business/Industry

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

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

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

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

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

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

- ☐ \$10 billion or more
- ☐ \$1 billion - \$9.9 billion
- ☐ \$100 million - \$99 million
- ☐ \$10 million - \$99.9 million
- ☐ \$1 million - \$9.9 million
- ☐ Less than \$1 million
- ☐ Don't know

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

01 ☐ Yes 02 ☐ No

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

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

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



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

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

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

sessions and schedule are subject to change without prior notice.

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



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

Hynes Convention Center, Boston

Multi-Pack Subscriptions

SAVE UP TO \$400

Pick a 3-Pack, a 6-Pack or a 9-Pack for incredible savings and receive as many as **3 FREE BONUS CDs!**



Two Years / 24 issues - \$99.99
One Year / 12 issues - \$69.99
One Year - Canada & Mexico - \$89.99
One Year - All Other Countries - \$99.99



Two Years / 24 issues - \$89
One Year / 12 issues - \$49.99
One Year - Canada & Mexico - \$79.99
One Year - All Other Countries - \$99.99



Two Years / 24 issues - \$99.99
One Year / 12 issues - \$69.99
One Year - Canada & Mexico - \$89.99
One Year - All Other Countries - \$170



Two Years / 24 issues - \$99.99
One Year / 12 issues - \$69.99
One Year - Canada & Mexico - \$89.99
One Year - All Other Countries - \$99.99



Two Years / 24 issues - \$169.99
One Year / 12 issues - \$149
One Year - Canada & Mexico - \$169
One Year - All Other Countries - \$180



Two Years / 24 issues - \$169.99
One Year / 12 issues - \$149
One Year - Canada & Mexico - \$169
One Year - All Other Countries - \$179



Two Years / 24 issues - \$129
One Year / 12 issues - \$89.99
One Year - Canada & Mexico - \$99.99
One Year - All Other Countries - \$129.99



Two Years / 24 issues - \$89
One Year / 12 issues - \$49.99
One Year - Canada & Mexico - \$79.99
One Year - All Other Countries - \$99.99



Two Years / 24 issues - \$169.99
One Year / 12 issues - \$149
One Year - Canada & Mexico - \$169
One Year - All Other Countries - \$179

RECEIVE
YOUR DIGITAL
EDITION
ACCESS CODE
INSTANTLY
WITH YOUR PAID
SUBSCRIPTIONS

3-Pack

Pick any 3 of our magazines for only \$175⁰⁰ and save up to \$275 for a 1 year subscription plus a **FREE CD**

- 2 Year - \$299.00
- Can/Mex - \$245.00
- All Other Countries - \$315.00

6-Pack

Pick any 6 of our magazines for only \$395⁰⁰ and save up to \$350 for a 1 year subscription plus 2 **FREE CDs**

- 2 Year - \$669.00
- Can/Mex - \$555.00
- All Other Countries - \$710.00

9-Pack

Pick all 9 of our magazines for only \$495⁰⁰ and save up to \$400 for a 1 year subscription plus 3 **FREE CDs**

- 2 Year - \$839.00
- Can/Mex - \$695.00
- All Other Countries - \$890.00

Subscribe online today www.sys-con.com/suboffer.cfm

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

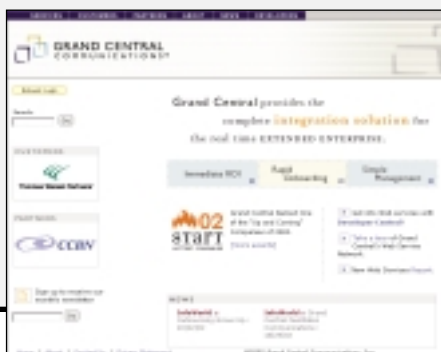
SYS-CON
MEDIA

Grand Central Communications' Web Services Network Obtains .NET Certification

(San Francisco) – Grand Central Communications has announced that its Web Services Network has obtained Microsoft Premium .NET certification. Web Services Network provides customers with a secure solution for extending .NET applications outside the corporate firewall.

The Grand Central Web Services Network leverages open Internet and Web services standards to enable enterprises to quickly, securely, and cost-effectively integrate .NET applications with those of their partners and customers. Since it plugs into an organization's existing infrastructure, companies do not need any additional software or hardware to use the network. Grand Central's Web Services Network also performs data transformation, orchestration, and routing services in the network, so .NET customers can easily integrate their applications with those of their partners even if their partners don't have the .NET Web services interface on their side.

www.grandcentral.com



OASIS Forms Committee on PKI Adoption for Secure Transactions

(Boston) – The OASIS standards consortium has created a technical committee within the OASIS PKI Member Section to advance adoption of the Public-Key Infrastructure (PKI) for Web services and other applications. PKI is a foundation to enable secure e-business transactions.

The new committee will serve as a global information resource for PKI and will work to increase awareness of digital certificates as an important component for managing access to network resources, delivering secured electronic messages and conducting electronic transactions.

www.oasis-open.org

DataPower and Contivo Offer Standards-Based Solution

(Cambridge, MA) – DataPower Technology, Inc., provider of intelligent XML-aware network infrastructure, has

announced a joint marketing and development agreement with Contivo, Inc., provider of automated data integration, to integrate the Contivo Enterprise Integration Modeling (EIM) solution with the DataPower XA35 XML Accelerator. The joint solution provides enterprise customers with a complete standards-based XML integration solution.

DataPower's XA35 provides the world's fastest XML data transformation engine. In a jointly conducted test, the XA35 was able to execute Contivo transaction mappings over 15 times faster than transactions attempted in general-purpose software.

www.datapower.com, www.contivo.com

Allstate Partners with Microsoft to Build Producer Network

(Redmond, WA and Northbrook, IL) – Microsoft Corp. and Allstate Financial, a business unit of the Allstate Corp.,

have announced a network solution for independent producers that the companies jointly developed in just seven months. Allstate wanted to extend its five existing policy management systems for access by the company's national network of independent producers. Using Microsoft .NET and the .NET Framework, Allstate built a solution to put Allstate Financial tools at producers' fingertips.

Allstate used Visual Studio .NET, the .NET Framework, Windows 2000 Server, SQL Server 2000 and BizTalk Server 2002 to harness XML Web services along with other technologies, including J2EE connectors. Allstate developed AccessAllstate.com as a strategic technology solution to integrate multiple information systems and services and present them in a way that enables producers to transact business with the firm. Agents can service accounts, catalog forms, retrieve product and policy information, and access sales support 24 hours a day, 7 days a week. By easily navigating one central Web site for all these capabilities, agents will be more efficient in servicing their customers, while focusing on opportunities to reduce operating costs and increase revenues.

<http://allstate.com>, www.microsoft.com

Propylon Launches PropelX 2.0

(Dublin) – Propylon Limited, a provider of data integration technology, has released PropelX v2.0, the world's first data integration platform based on XML pipeline transformation technology.

PropelX provides a graphical development center that combines the ability to design, develop, and deploy data integration solutions. It uses XML to radically simplify the task of integrating content and data systems. Rather than modify systems to conform to each other, Propylon's products mediate between the systems using advanced data transformation technology. This mediation approach to interoperability is more cost effective and simpler than normal integration efforts.

www.propylon.com



Optika Announces Web Services and .NET Capabilities with Acorde 3.0

(Colorado Springs, CO) – Optika, Inc., a provider of enterprise content management (ECM) solutions, has announced the controlled release of its next-generation product suite. Over 20 Optika customers are participating in this program.

With the Acorde 3.0 product family, customers can utilize powerful Web services to access the Acorde repository, leverage the Microsoft .NET platform via the Acorde toolkit, and provide for full text search and retrieval of critical business documents. In addition to the new feature set, the Acorde 3.0 product family includes extended integration capabilities with Oracle E-Business Suite and the Microsoft Business Solutions product family.

www.optika.com

The World's Leading Java Resource!

JAVA DEVELOPER'S JOURNAL

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

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

Subscribe Today &
SAVE
30% Off
the annual cover price

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

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

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

WSJ ADVERTISER INDEX

ADVERTISER	URL	PHONE	PAGE
.NET Developer's Journal	www.sys-con.com/dotnet	888-303-5282	25
Altova	http://xmjl.altova.com/xmlspy5		13
BEA eWorld	www.bea-eworld.com		19
Edge Web Hosting	www.edgewebhosting.com	866-EDGEWEB	17
Java Developer's Journal	www.javadevelopersjournal.com	888-303-5282	57
JDJ Store	www.jdjstore.com	888-303-5282	23
Mind Reef	www.mindreef.com	603-465-2204	4, 5
Parasoft	www.parasoft.com/ws2	888-305-0041	2
Sitraka (now part of Quest Software)	www.sitraka.com/jclass/ws	800-663-4723	6
Sitraka (now part of Quest Software)	www.sitraka.com/jclass/ws	800-663-4723	59
SYS-CON Media	www.sys-con.com	888-303-5282	31
SYS-CON Media	www.sys-con.com/suboffer.cfm	888-303-5282	55
SYS-CON Media	http://developer.sys-con.com	888-303-5282	41
Web Services Edge 2003	www.sys-con.com	201-802-3069	39, 47-54
Web Services Journal	www.wsj2.com	888-303-5282	35
WebLogic Developer's Journal	www.weblogicdevelopersjournal.com	888-303-5282	27

Advertiser is fully responsible for all financial liability and terms of the contract executed by their agents or agencies who are acting on behalf of the advertiser. This index is provided as an additional service to our readers. The publisher does not assume any liability for errors or omissions.

IN THE NEXT ISSUE OF *WSJ*...

Focus: Web Services Security

Extending Web Services to the Real World

The first in a series, this article looks at J2ME and wireless Web services. We look at the limitations of small-footprint devices and how using XML/RPC over HTTP is an efficient way to propagate transmissions and content.



Effective Web Services Security Architecture

Traditional applications are connection-oriented, but Web services are message oriented and lack the guarantee of a direct connection between service provider and consumer. What are the business requirements for your security architecture in the face of these differences?



Security Standards

Web services standards for security are just beginning to emerge and stabilize, yet lack of security standards is listed as the top barrier to Web services adoption. This article will explore strategies for implementing Web services security in ways that can help isolate the developer from standards that are in a constant state of flux.



A Success Story: End-to End Encryption for Web Services

What does it take to convince aerospace and defense companies that it's safe to permit sensitive product design information to be shared and stored outside the corporate firewall?



Web Services Need Security, But What Type?

What is it about Web Services that provoke the security concerns? This article attempts to dispel some of the confusion around Web services security, and examines the current options for the deployment of Web services.



The Web Services Challenge

A look at the business problems addressed by Web services, the benefits they offer, and the new challenges they present, as well as the security strategies available today.



Web Services
JOURNAL





Rajiv Gupta

Dr. Rajiv Gupta is the founder and CEO of Confluent Software. Prior to this, he was the general manager of Hewlett-Packard's e-Speak business unit, which was responsible for developing the first Web services development and deployment platforms. Previously, he was manager of the Advanced Architecture Group at HP Labs. He holds more than 35 patents, several of which are in Web services technologies.

RGUPTA@CONFLUENTSOFTWARE.COM

The Promise and Peril of Web Services: Management Will Make the Difference

The concept of Web services has seen more than its fair share of media coverage over the last year. And so has the concept of service-oriented architectures, which is the use of Web services to define a model of loose coupling between applications. But the industry buzz regarding this latest "Next Big Thing" is surprisingly devoid of one important aspect that will mean the difference between the success and failure of Web services or service-oriented architectures – management.

Simply put, without the proper management of Web services deployments, enterprises will not see the business agility or improvements in developer productivity that they had hoped for. Instead, it is likely that they will find themselves in a chaotic mess with reduced qualities of service and higher costs to bring control back into their application environ-

ments. Such an outcome could seal the fate of Web services before they can deliver on some of their very real and achievable promises.

In traditional application development, management was an afterthought – the process went from develop to deploy, and only later did we think of managing what we deployed. As we started to deploy distributed applications the fallacy of that approach made us realize that management and manageability need to be considered much sooner. Web services, with their highly distributed, loosely coupled, plug-and-play, and oftentimes asynchronous nature demand that the process be develop, manage and deploy

Think of it in terms of airplanes and air traffic control. Sure, a pilot can fly a plane from point A to point B. But if you don't have air traffic control, you better hope no other planes are in the air. Even if there are no other planes in the air, the pilot still needs air traffic control to inform him of changing ground conditions. The pilot focuses on flying the plane, and air traffic control focuses on providing visibility and enforcing discipline consistently across the planes. With Web services, visibility and control are essential and are provided by a management platform that is to Web services what air traffic control is to planes. Like air traffic controllers, IT managers must have the tools in place that allow them to keep a pulse on a constantly changing environment and to take corrective action when necessary. And they need to take this corrective action without changing the Web service itself, just like the air traffic controller does not need to change any parts in the planes that are in the air.

There is another aspect of Web services and Web service management that is important: we need to think in terms of the life cycle of Web services –

from development, to integration, to deployment, to change management. And testing, testing, testing throughout the life cycle. While this was important for traditional application environments, it is critical for Web services for the simple reason that the line between development and deployment is starting to blur. We now hope to have more "in-the-field" upgrades of Web services, more distributed, less coordinated change, and we will.

The management platform should provide seamless coordination among integration, management, and business analytics – helping companies respond quickly and intelligently to constantly changing business conditions.

Integration: Reduce Costs and Gain Control. IT organizations looking to leverage Web services in business-critical integrations need to define and enforce policies for security, quality of service, logging, and change management. A good management platform should provide a central console that reaches across multiple services – which can cut development time and keep managers focused on higher-level business logic instead of low-level plumbing, dramatically reducing maintenance and upgrade costs.

Management: Achieve Operational Visibility. The end-to-end management platform should also include monitoring capability to deliver operational visibility into Web services and Web services integrations – enabling IT managers to keep a close eye on critical service metrics across the enterprise, such as performance levels, downtime, and security violations. By knowing what is happening at every level of the integration – thereby improving application integrity, slashing problem diagnosis time, and enhancing security – enterprises can meet and exceed service-level requirements and ultimately improve customer satisfaction.

Analysis: Monitor Business Activity. Since Web services interactions can, and typically do, expose higher semantic information in the XML payloads and in the WSDL descriptions, the management platform should allow this business information to be processed and monitored. Therefore, a component of a comprehensive management platform is a business activity monitoring solution to provide business users with continuous visibility into important business performance indicators, e.g. number of orders, value of those orders, and number of goods shipped. With the ability to continuously monitor and analyze business processes, IT managers can proactively detect business problems or opportunities and rapidly notify appropriate business operations staff.

Enterprises know that to succeed in today's budget-constrained and highly competitive business environment, they need to be responsive to customer, partner, employee, and supplier demands – sometimes called a real-time enterprise. Achieving a real-time business state is possible with today's Web services technologies, but it does have risks. Only with a comprehensive management platform in place can companies hope to achieve significant ROI and fully realize the promise of Web services. ©

Sitraka

(now part of Quest Software)

www.sitraka.com/jclass/ws

BEA Systems

www.dev2dev.bea.com/userworkshop