

WebServices

JOURNAL

.NET J2EE XML

WSJ2.COM

SAVE UP TO \$400

see page 41 for details



SUBSCRIBE TODAY

and get UP TO **3 FREE CDs!**
(WHILE SUPPLIES LAST)

From the Editor Mixed Nuts

by Sean Rhody pg. 3

Product Review Sift 1.5 by Service Integrity

Reviewed by Brian Barbash pg. 24

Web Services in the Real World Knowing When to Use Web Services

by Michael Blank pg. 26

Service-Oriented Architecture The Agile Enterprise

by Cathy Lippert
& Sharon Fay pg. 36

Enabling the High Performance Corporation

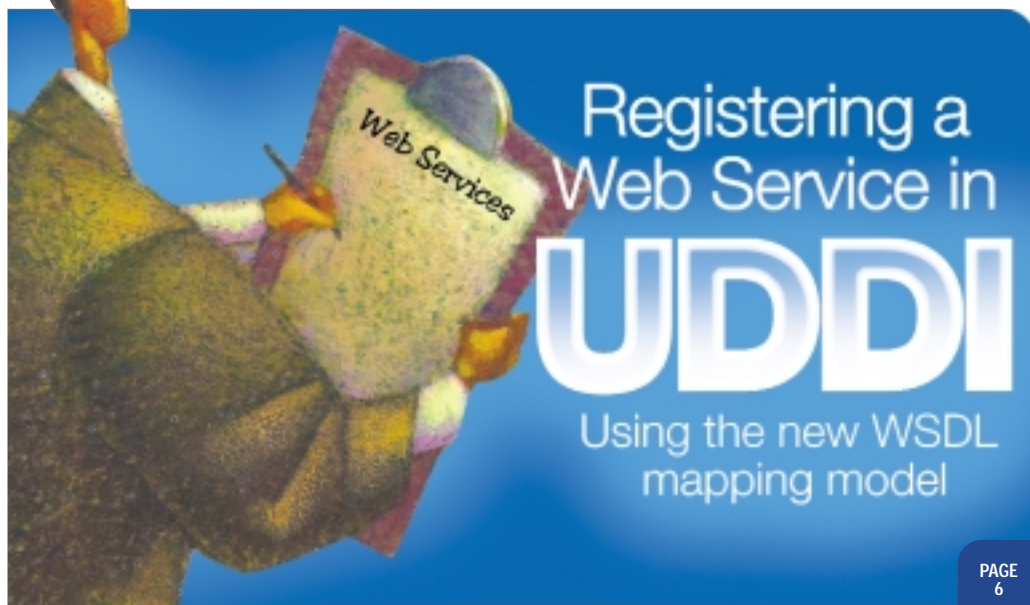
by Jonathan Sapir pg. 42

RETAILERS PLEASE DISPLAY
UNTIL NOVEMBER 30, 2003

\$6.99US \$7.99CAN



**SYS-CON
MEDIA**



PAGE
6

FOCUS ON UDDI/STANDARDS

Bringing Order to Enterprise Service Proliferation

The foundation of a flexible infrastructure



Toufic Boubez

12

Toward Web Services Management Standards

An architectural approach to IT system design

James Phillips

& Heather Kreger 18

Transactions in Business Processes—

A New Model Connecting services to desktop applications



Rich Rollman

& William Cox 34

A Look at WS-I

A conversation with Tom Glover & Andy Astor



Sean Rhody

46

Building Your Own SOAP Client & Reporting Tool

Moving from evaluation to Adoption

Bikash Behera

& Jim Winfield 52

Think Async

The core of a new architecture



Edwin Khodabakhchian

58

WSJ Feature: The Enterprise Service

Bus A developer-friendly integration engine



Nigel Thomas

& Warren Buckley 30

BPM: Facing the Challenges of Web Services BPM

A first leg on the BPM journey

Alejandro Danylyszyn

& Cesare Rotundo 40

Parasoft

www.parasoft.com/wsj10

INTERNATIONAL ADVISORY BOARD

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

TECHNICAL ADVISORY BOARD

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

EDITORIAL

EDITOR-IN-CHIEF

Sean Rhody sean@sys-con.com

INDUSTRY EDITOR

Norbert Mikula norbert@sys-con.com

PRODUCT REVIEW EDITOR

Joe Mitchell joe@sys-con.com

.NET EDITOR

Dave Rader davidr@fusiontech.com

TECHNICAL EDITORS

Andrew Astor aastor@webmethods.com

David Chappell chappell@sonicsoftware.com

Anne Thomas Manes anne@manes.net

Mike Sick msick@sys-con.com

EXECUTIVE EDITOR

Gail Schultz gail@sys-con.com

EDITOR

Nancy Valentine nancy@sys-con.com

ASSOCIATE EDITORS

Jamie Matusow jamie@sys-con.com

Jean Cassidy jean@sys-con.com

ASSISTANT EDITOR

Jennifer Van Winckel jennifer@sys-con.com

PRODUCTION

PRODUCTION CONSULTANT

Jim Morgan jim@sys-con.com

LEAD DESIGNER

Richard Silverberg richards@sys-con.com

ART DIRECTOR

Alex 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 Barbash, Bikash Behera, Michael Blank, Toufic Boubez,
Warren Buckley, Alejandro Danylyszyn, Sharon Fay, Edwin
Khodabakhian, Heather Kreger, Cathy Lippert, Anne Thomas Manes,
James Phillips, Sean Rhody, Cesare Rolundo, Jonathan Sapir,
Nigel Thomas, Jim Winfield

EDITORIAL OFFICES

SYS-CON MEDIA

135 CHESTNUT RIDGE ROAD, MONTVALE, NJ 07645

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

WEB SERVICES JOURNAL (ISSN# 1535-6906)

Is published monthly (12 times a year)

By SYS-CON Publications, Inc.

Periodicals postage pending

Montvale, NJ 07645 and additional mailing offices

POSTMASTER: Send address changes to:

WEB SERVICES JOURNAL, SYS-CON Publications, Inc.

135 Chestnut Ridge Road, Montvale, NJ 07645

©COPYRIGHT

Copyright © 2003 by SYS-CON Publications, Inc. All rights reserved.

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

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

Mixed Nuts

Written by
Sean Rhody



Author Bio:

Sean Rhody is the editor-in-chief of Web Services Journal and managing editor of WebLogic Developer's Journal. He is a respected industry expert and a consultant with a leading consulting services company.
SEAN@SYS-CON.COM

One of the most frustrating things I've ever encountered in my life is trying to loosen a nut using a socket from the wrong measurement system. You know, I've got a metric nut, but an English socket set. So I find a socket that's close, but it's loose, and inevitably I end up stripping the nut, bruising my knuckles, and generally using language I don't care to repeat. To paraphrase an old saw – the only thing worse than no standard is two standards.

Web services often feels equally bruising, to my brain, if not to my knuckles (although I tend to bruise them hitting something in frustration too). Whoever said, "The wonderful thing about standards is there are so many of them" was certainly familiar with Web services.

In this issue we focus on standards and interoperability. Although I rant and rave, things are better now than they were initially. Most SOAP implementations will now work with one another, and the WS-I has released the first version of the Basic Profile, which will assist everyone. I remember one of our earliest proposals concerned workarounds to make different Web services implementations actually work together. We've come a long way.

But that doesn't mean we're out of the woods by any means. As usual, there is a problem. Or rather, there are multiple problems. Most of them involve standards. The first problem is that there are too many standards bodies. Between the W3C and OASIS, it's a cinch you can make your marketing point by proposing some standard and seeing which place it sticks. Then the poor WS-I gets to figure out how to make it work with all the other various overlapping standards. It doesn't help that some of these bodies will actually accept multiple standards for the same thing – it's bad enough the two bodies have a competition going, but to have one body running multiple standards for the same idea is really dumb.

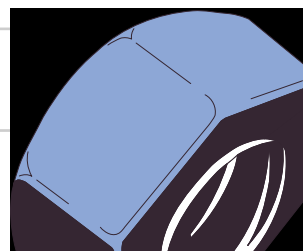
Fortunately for the most part the basics are now working. But the battle around the next set of standards – security, business process, and management – is still sizzling. It looks like the business process specification is solidifying under BPEL4WS, but the various orchestration and choreography specifications and their subspecifications are still up in the air. Security is stuck with the split personality of SAML and WS-Security, and management is a mess.

What we need at this point is a grass-roots movement to limit the number of standards and focus adoption on a single standard – effectively short-circuiting the whole argument. This would allow us, the consumers, to effectively accelerate the development of Web services by starving out unworthy standards. Much like a consortium uses its buying power to arrange better pricing.

So, I hereby propose the establishment of the WSSSC – the Web Services Standards Selection Committee. The WSSSC will not propose new standards – we'll leave that to OASIS and W3C. Nor will we try to make them work together; we'll leave that to the WS-I. Our sole purpose will be to sort standards into meaningful categories, and to pick the one that we think best suits the needs of the industry. Once we make a selection, we'll be bound to use only that standard.

Of course, we'll need offices, computers, and such. Send your contributions, starting at \$10,000 addressed to me, care of the magazine office. Once we reach \$1 million, I will get everything rolling...

But seriously, we do need to pressure the industry to stop the insanity. Too many bodies, too many standards. Options are good, to a point, but the automobile only really got started when Henry Ford said "You can have any color you like, as long as it's black." Time for a little black paint in the Web services soul. Oh, and any contributions I do receive will be used for a new socket set with both English and metric. ☺



Mindreef News

www.mindreef.com/nl0309

Mindreef

www.mindreef.com

Written by Anne Thomas Manes

Registering a Web Service in UDDI

Using the new WSDL mapping model



UDDI (Universal Description, Discovery and Integration) is a registry for Web services. It provides a mechanism to advertise and discover Web services. Although you don't need to use UDDI to implement a Web services solution, you'll find that a UDDI registry greatly simplifies the management and administration of your services, particularly once they have reached a certain critical mass.

AUTHOR BIOS:



Anne Thomas Manes is a research director at Burton Group, a research, consulting, and advisory firm, where she leads research for the Application Platform Strategies service. Anne is a renowned technologist in the Web services space. She participates in standards development at W3C and OASIS and is a member of the Web Services Journal editorial board, frequent speaker, and author of numerous articles and the book, *Web Services: A Manager's Guide* (Addison Wesley).
ANNE@MANES.NET

Once you've developed more than a few services, and once you start giving access to those services to more than a few controlled individuals, management starts to get more challenging. Potential service consumers need a way to search for available services, to determine which services might be applicable to their projects, and to find metadata about those services. In particular, a service consumer

may use UDDI registries set up by its customers and business partners. The UDDI Business Registry (UBR) is a free, public registry operated by IBM, Microsoft, SAP, and NTT, although few businesses feel comfortable advertising their business services in such an insecure public forum.

One key difference between a UDDI registry and other registries and directo-

"A UDDI registry greatly simplifies the management and administration of your services"

must find the WSDL (Web Services Description Language) files that define a service. UDDI was designed to support these requirements.

A business may set up multiple UDDI registries in-house to support intranet and extranet operations, and a business

ries is that UDDI provides a mechanism to categorize businesses and services using taxonomies. For example, service providers can use a taxonomy to indicate that a service implements a specific domain standard, or that it provides services to a specific geographic area. These



oriented concepts, a *type* is the definition of a thing, and an *implementation* is an instance of a thing. A service-type registration, called a technical model (tModel), represents a technical specification or some other metadata. For example, a tModel could represent a WSDL document that defines a Web service.

A service implementation registration represents a service offered by a specific service provider. It specifies information about the business entity that offers the service (businessEntity), describes the service (businessService), and captures the binding information (bindingTemplate) required to use the service. The binding Template captures the service endpoint address, and associates the service with the tModels that represent its technical specifications. Figure 1 shows the UDDI data model that captures these registrations.

UDDI also uses tModels to represent taxonomies. Service providers categorize UDDI registrations using a construct called a keyedReference. A keyedReference allows you to assign property values to a UDDI entity using a name/value pair. The “name” is the tModelKey of the taxonomy tModel. The “value” is the descriptive information supplied for this categorization. You may define a limited

set of valid values (a Value Set) for a taxonomy. For example, you could define a taxonomy to represent the deployment status of a service, and define a Value Set with values of “development”, “test”, and “production”. To indicate that a service is in production, you would add a keyedReference to the businessService entity that represents the service. The keyedReference would specify the tModelKey for the deployment status tModel taxonomy, and a value of “production”. It would look something like this:

```
<keyedReference
  tModelKey="uuid:2e444afb-33e5-
    3a7b-81b7-1cb8a373f457"
  keyName="deployment status"
  keyValue="production"/>
```

You’ll notice that the keyedReference includes three attributes: tModelKey, keyName, and keyValue. The tModelKey and keyValue attributes are required. The keyName attribute is optional and insignificant. The keyName attribute allows you to assign a human-readable name to the keyedReference attribute, but you cannot search the registry using this name.

taxonomies make it easier for consumers to find services that match their specific requirements.

Now, of course, in order for consumers to find a service, the service provider must first properly register the service. There are many different ways to register a service, so it’s important to define registration conventions so that consumers know what to search for.

In July 2003, the standards group that manages the UDDI specification (the OASIS UDDI-Spec Technical Committee) published a Technical Note called “Using WSDL in a UDDI Registry, Version 2.0.” This Technical Note defines conventions for registering a Web service based on information in the service’s WSDL definition. (This Technical Note supercedes the previous UDDI Best Practices document, “Using WSDL in a UDDI Registry, Version 1.08.”)

The UDDI Data Model

A UDDI registry manages information about service *types* and service *implementations*. Going back to basic object-

The WSDL Data Model

A WSDL file describes a Web service. A



FIGURE 1 The UDDI data model

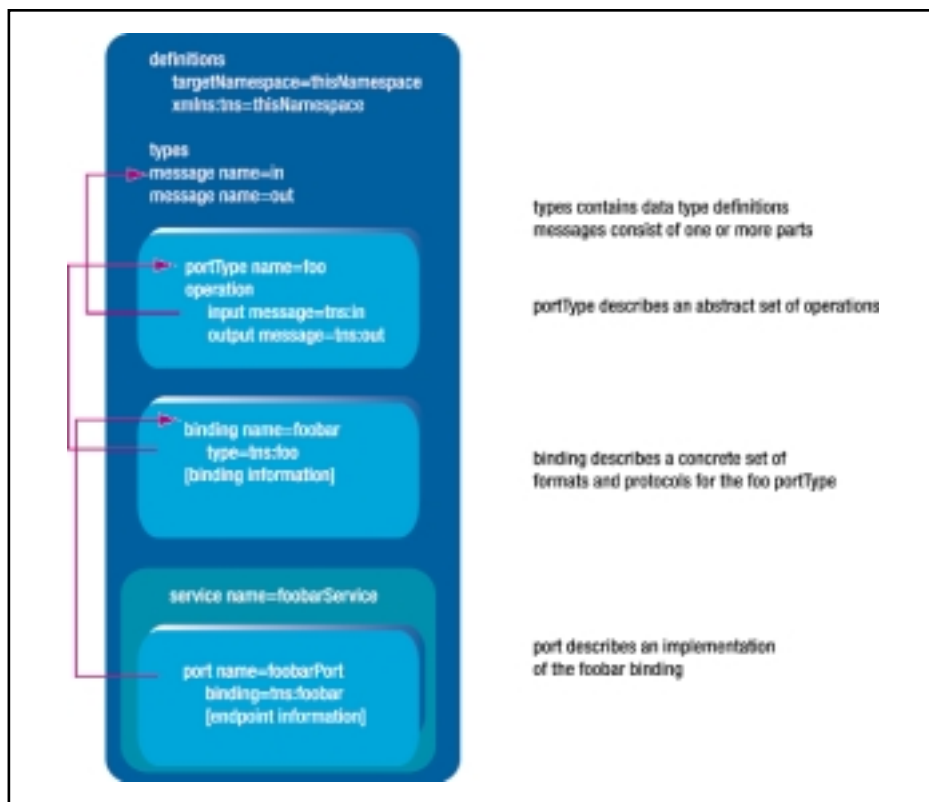


FIGURE 2 | The WSDL data model

WSDL file describes *what* functionality a Web service offers, *how* it communicates, and *where* to find it. Figure 2 shows the WSDL data model, and shows the relationship of the various definition elements.

The *what* part of a WSDL document (a *portType*) defines the abstract interface of a Web service. A *portType* is a reusable definition component. Multiple providers can offer Web services that implement the same *portType*. A *portType* specifies what operations the abstract service supports, and it specifies the input and output messages associated with each operation. Each message is defined in a separate *message* definition, which in turn references element or type definitions in the *types* section.

The *how* part of a WSDL document (a *binding*) maps a *portType* to a concrete set of formats and protocols. A *binding* indicates how the input and output messages defined in the referenced *portType* should be packaged into a message, and it specifies what communication protocols can be used to convey the messages. As with

Choreology

www.choreology.com

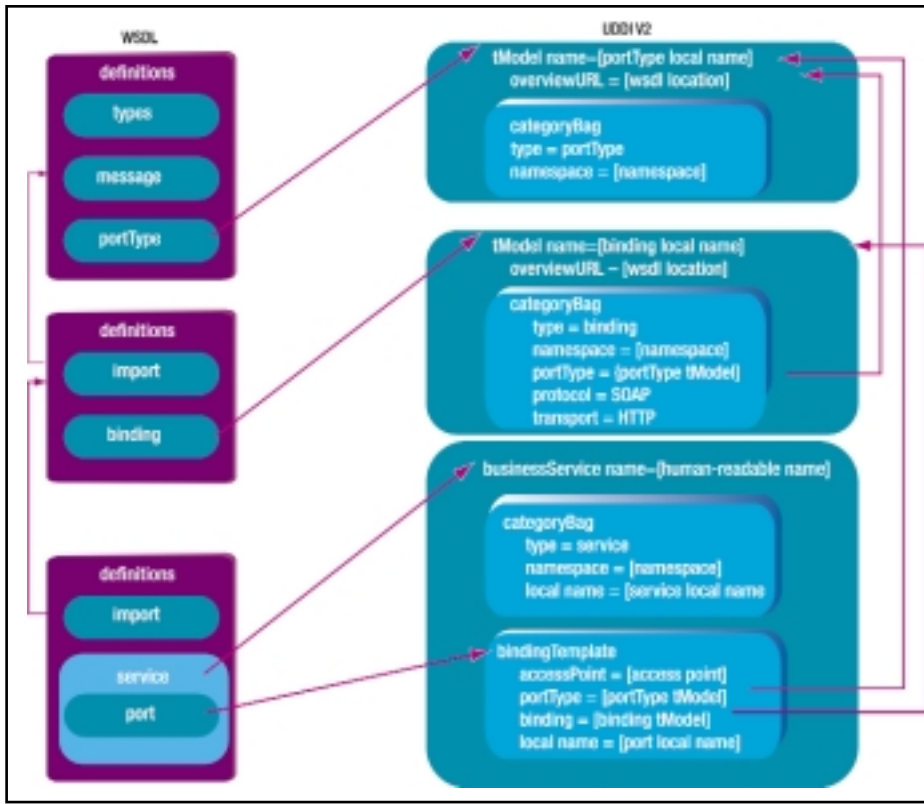


FIGURE 3 | Mapping WSDL to UDDI

portTypes, a WSDL binding is a reusable definition component. Multiple service providers can support the same WSDL binding. A service can also implement multiple bindings of the same portType, thereby supporting multiple protocols.

The *where* part of a WSDL document (a *service*) describes a specific Web service implementation. A Web service implementation contains one or more *ports*. A port implements the referenced binding and specifies the endpoint of that Web service binding. A business might offer multiple endpoints to a particular service, each implementing a different binding.

Mapping WSDL to UDDI

The “Using WSDL in a UDDI Registry” Technical Note defines a mapping model between WSDL and UDDI that supports the reusability characteristics of the WSDL data model.

Figure 3 shows a Web service description that consists of three different WSDL files for the portType, binding, and ser-

vice definitions. The service definition imports the binding definition, and the binding definition imports the portType definition. By breaking the descriptions into multiple files, the portType and binding definitions support reusability.

capture the namespace, to reference the tModel that describes the portType for this binding, and to indicate protocols supported by this binding.

- **Register each WSDL service as a business-Service.** Give the businessService a hu-

“The new mapping gives users some powerful options when searching for services”

This is the approach people should take when defining corporate standards or industry-specific domain standards. A standards group should define the abstract service portTypes and bindings, but it should not define specific service implementations. A service provider that implements a standard service description should not redefine the standard definitions. It should import the standard definitions.

The WSDL-to-UDDI mapping model is designed to help users find services that implement standard definitions. It also helps users find services that support a specific set of protocols, such as SOAP over HTTP. The Technical Note defines a new set of canonical tModels to help capture the necessary information. These tModels include categorization systems to capture the WSDL element type (portType, binding, or service) and the element's fully qualified name (its local name and namespace name), along with the protocols supported by a WSDL binding.

The mapping model works as follows:

- **Register each WSDL portType as a tModel.** Give the tModel the same name as the portType local name. Capture the URL of the WSDL file in the tModel overviewURL. Use keyedReferences to indicate that the tModel represents a WSDL portType and to capture the namespace.
- **Register each WSDL binding as a tModel.** Give the tModel the same name as the binding local name. Capture the URL of the WSDL file in the tModel overviewURL. Use keyedReferences to indicate that the tModel represents a WSDL binding, to

man-readable name. Use keyedReferences to indicate that this service is described by WSDL and to capture the WSDL service local and namespace names.

- **Register each WSDL service port as a bindingTemplate.** Capture the port endpoint address in the bindingTemplate accessPoint. Using the tModelInstanceDetails structure, associate the service with all the tModels that describe the service. At a minimum, the bindingTemplate should point to the tModels that represent the WSDL portType and WSDL binding definitions. Capture the port's local name in the InstanceParms field of the WSDL binding link.

Conclusion

The new mapping gives users some powerful options when searching for services. It permits users to perform the following types of queries:

- Find specifications by namespace name
- Find specifications by WSDL portType or binding name
- Find all bindings for a portType
- Find all SOAP bindings for a portType
- Find implementations by namespace name
- Find all implementations of a WSDL portType
- Find all SOAP implementations of a WSDL portType
- Find all implementations of a WSDL binding

As your investment in Web services grows, you'll no doubt find these capabilities valuable. ©

WebServices JOURNAL

PRESIDENT AND CEO

Fuat Kircaali fuat@sys-con.com

VP, BUSINESS DEVELOPMENT

Grisha Davida grisha@sys-con.com

GROUP PUBLISHER

Jeremy Geelan jeremy@sys-con.com

TECHNICAL DIRECTOR

Alan Williamson alan@sys-con.com

ADVERTISING

SENIOR VP, SALES & MARKETING

Carmen Gonzalez carmen@sys-con.com

VP, SALES & MARKETING

Miles Silverman miles@sys-con.com

ADVERTISING DIRECTOR

Robyn Forma robyn@sys-con.com

DIRECTOR, SALES & MARKETING

Megan Mussa megan@sys-con.com

ADVERTISING SALES MANAGER

Alisa Catalano alisa@sys-con.com

ASSOCIATE SALES MANAGERS

Carrie Gebert carrieg@sys-con.com

Kristin Kuhnle kristin@sys-con.com

SYS-CON EVENTS

PRESIDENT, SYS-CON EVENTS

Grisha Davida grisha@sys-con.com

CONFERENCE MANAGER

Michael Lynch mike@sys-con.com

REGIONAL SALES MANAGER

James Donovan james@sys-con.com

CUSTOMER RELATIONS/JDJ STORE

CIRCULATION SERVICE COORDINATORS

Niki Panagopoulos niki@sys-con.com

Shelia Dickerson shelia@sys-con.com

Edna Earle Russell edna@sys-con.com

JDJ STORE MANAGER

Rachel McGouran rachel@sys-con.com

SYS-CON.COM

VP, INFORMATION SYSTEMS

Robert Diamond robert@sys-con.com

WEB DESIGNERS

Stephen Kimurray stephen@sys-con.com

Christopher Croce chris@sys-con.com

ONLINE EDITOR

Lin Goetz lin@sys-con.com

ACCOUNTING

FINANCIAL ANALYST

Joan LaRose joan@sys-con.com

ACCOUNTS RECEIVABLE

Kerri Von Achen kerri@sys-con.com

ACCOUNTS PAYABLE

Betty White betty@sys-con.com

SUBSCRIPTIONS

SUBSCRIBE@SYS-CON.COM

1-888-303-5282

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

COVER PRICE: \$6.99/ISSUE

DOMESTIC: \$69.99/YR (12 ISSUES)

CANADA/MEXICO: \$89.99/YR

ALL OTHER COUNTRIES: \$99.99/YR

(U.S. BANKS OR MONEY ORDERS)

WorldWide Newsstand Distribution

Curtis Circulation Company, New Milford, NJ

FOR LIST RENTAL INFORMATION:

Kevin Collopy 845 731-2684

kevin.collopy@edithroman.com

Frank Cipolla 845 731-3832, frank.cipolla@epostdirect.com



Ektron

www.ektron.com/ws

StrikeIron

www.strikeiron.com

StrikeIron

www.strikeiron.com

Bringing Order to Enterprise Service Proliferation

The foundation of a flexible infrastructure



UDDI has been around for almost three years now. It has gone from an initial proposal by three companies (Ariba, IBM, and Microsoft), to a consortium effort (www.uddi.org) with a community of hundreds, and finally into the hands of the OASIS standards body. Along the way, the original specification has gone through two additional revisions. Perhaps more important, with each new revision the business value of UDDI has shifted from being a registry of public services to a central fixture in private EAI and partner-integration efforts.

The most recent version of UDDI, V3, continues this trend by making UDDI more usable as a common service registry inside an extended enterprise and between partners. Thus this technology, originally designed for publishing and discovering Web services, can today be leveraged by enterprises to provide a complete solution for managing access to Web services including subscription control and endpoint indirection.

Is UDDI Only for the Public Internet?

Along with SOAP and WSDL, UDDI is considered one of the three “pillars” of Web services technologies. Unfortunately, it is probably the least understood, at least initially. At the root of the various misconceptions about UDDI is the schizophrenic nature of the specification. The UDDI idea initially (and to this day) encompasses two **very** distinct concepts. The first one is the concept of a universal, online business registry, the

Universal Business Registry (UBR), maintained and operated by a set of impartial operators along the lines of the DNS infrastructure. It is hoped that this UBR will act as some form of universal, multilayered “phone book” for registering businesses and the services they provide across the world, (for good or bad, this “universality” is baked into the UDDI name). This is the concept most people know about, and the one that is the most discussed, at least by the various UDDI naysayers. It is also perhaps the one least relevant to business, and just one incarnation of the more important concept behind UDDI: that of a generic services registry based on a set of open, standards-based, data structures and a set of SOAP-based APIs to programmatically register, organize, and find services and their descriptions.

Where UDDI is seeing its greatest adoption and showing greatest value is inside the enterprise. There, with some minor enhancements, it can bring security and flexibility to

the process of publishing, discovering, and subscribing to internal services. The alternative is a disorganized soup of services that are published by local teams without any central organization, security, or management. With UDDI V3, the necessary functionality now exists for UDDI to take its place as core enterprise integration infrastructure.

UDDI Basics

When UDDI was first released, the concept of registries was by no means new. Corporate IT departments had been using naming and directory services for some time and several vendors had commercial registries on the market. The difference was that UDDI proposed an open collaborative effort to produce a standards-based registry that was not controlled by any vendor. Requirements for a registry are well understood: a data model for the metadata and a set of CRUD operations on these structures. The data model has to enforce ownership and containment requirements; a logical, consistent referencing system for the containment relationships; and a classification taxonomy to simplify searching. The CRUD API has to enforce security or authentication for operations that change data, and provide a query language for searching and retrieving data. The requirements that went into designing UDDI are no different.

UDDI has been described in detail in many articles and books but it would be helpful for us as a baseline to go over its structures and API. The UDDI data model has four main elements: `businessEntity`, `businessService`, `bindingTemplate`, and `tModel`. Use of the word “business” in two of the elements is unfortunate, but it should not be a deterrent since the pattern of registering and querying entities and the services they pr-

Author Bio

Toufic Boubez is the chief technology officer of Layer 7 Technologies, specializing in Web services security. While at IBM, he was the chief architect of IBM's Web services initiative and the architect of the first iterations of the IBM Web services Toolkit. He was also IBM's technical representative to UDDI and a coauthor of the UDDI V1 API specification. He is the coauthor of *Building Web Services with Java and Java P2P Unleashed* (Sams).

The 2003 Borland Conference

connect.borland.com/borcon03

“An example enterprise scenario facilitated by UDDI V3 but not its predecessors occurs when Web services are being developed internally and published to a test registry”

ovide is useful, regardless of the name of the elements. Actually, with the proper XML transformations, the UDDI data model can be reused for a variety of purposes by changing the names of the elements.

The containment model is very simple: business entities can contain a set of business services; business services have binding templates that provide implementation details for the various implementation flavors of the business service (e.g., Web browser-based or e-mail-based implementation of the same basic service); and finally tModels provide the reference or namespace mechanism used in the description of how to access these services.

The tModel concept is extremely important in understanding UDDI. You can think of it as a set of technical signatures that are assigned to services to facilitate searching and categorization. A set of tModels has already been established to represent some of the more important technical concepts such as HTTP or SOAP. As an example of how they are used, consider a service that is accessible through SOAP over SMTP: it would be labeled, among other tModels, with the SOAP and SMTP tModels. When searching for a particular service to access through SOAP over SMTP, the search can be restricted to entries labeled with those two tModels.

Another important use of the tModel is as a namespace. An industry group, for example, can create their own tModels to designate business processes and services that are provided by their members; in this case, searches can be restricted to services

within that namespace by using the appropriate tModel as a qualifier.

The UDDI API has two subsets: an authenticated Publishers' API to save, update, delete, and manage entries with security token management; and an open Inquiry API to find entries and get detailed content from them. In addition to the data model and the API, the UDDI specification allows the use of taxonomies for organizing the entries space and searching through it. Three taxonomies, for businesses, products and services, and geographic location, were originally specified.

UDDI V2: Becoming Enterprise Friendly

The UDDI v2 specification provided some additional flexibility in the UDDI usage model, mainly in providing the capability to add third-party taxonomies to the three that were already part of the specification. This provides enormous flexibility for organizations, whether private companies or consortia, for example, to define their own taxonomies and overlay them over other taxonomies. Another important concept introduced in v2 is that of business relationships through so-called publisher assertions. This allowed businesses (or organizations) to indicate that they are related through a partnership or other affiliation. Along with the additions to the data model, V2 introduced some changes to the API to handle the new data structures and relationships.

UDDI V3: Built for the Enterprise

The original UDDI specification was

mainly concerned with enabling a set of central universal registries and some of the constraints reflect these concerns. For example, the original specification required operators to issue unique identifiers for every registry entry. These identifiers took the form of UUIDs, and could only be generated by the operators, in order to avoid clashes and inconsistencies in the identifier namespace. This is a perfectly acceptable restriction in the case of one universal registry, but becomes unworkable when several independent registries need to share information as is common inside the extended enterprise.

An example enterprise scenario facilitated by UDDI V3 but not its predecessors occurs when Web services are being developed internally and published to a test registry. Once these services become operational, they have to be migrated to one or more operational registries, whether internal or external. Once propagated, the integrity of these registry entries needs to be ensured against tampering, since the single UBR, with its trusted operators, is not being used. This scenario touches on several important requirements that were taken into account for V3, namely the ability to share entries between registries, while maintaining the referential integrity of the assigned keys; and the ability to sign and certify registry entries.

The V3 specification provides many new features, but the significant ones are geared towards making UDDI registries into more of an enterprise-class IT infrastructure by addressing some of the issues above. These improvements can be grouped under three main categories:

General extensions:

- **Publisher-assigned keys:** In versions 1 and 2, the role of assigning unique keys for every entry in UDDI fell to the registry operators. While essential for the integrity of the registry, this constraint made copying entire entities from one registry to another, while maintaining the unique key, impossible. V3 does



away with that constraint, in order to support multi-registry environments. Using the Publishers' API, publishers can propose their own keys while publishing entries to new registries. This is called entity promotion from one local UDDI registry to another, and depending on its policies, a registry might not accept the suggested keys. Of course this creates its own complexities in terms of referential integrity of the keys. That's why three new and important concepts were introduced: support for root and affiliate registries; support for human-friendly URI-based keys; and support for digitally signed entries.

- **Federation of registries:** In order to support inter-registry data sharing while avoiding key collision, V3 now supports the concept of federated UDDI nodes, with a root registry and a set of affiliate registries. Publisher assigned keys are now scoped only within a hierarchy of UDDI nodes. Data from within the hierarchy can be freely moved around while conserving the uniqueness quality of the keys.
- **Human-friendly, URI-based keys:** in order to facilitate inter-registry data copying, the restriction for generating unique identifiers (UUIDs) for entries was relaxed, and a new format of identifiers was introduced, similar to the DNS format.
- **Support for digital signatures:** Although the Publishers' API (allowing the creation, deletion, and editing of entries) has been authenticated since V1, entries can be misrepresented by third parties, or errors introduced by copying them to other registries now that entries can be moved about within a set of federated registries. V3 allows entries to be signed digitally, for an enhanced level of security and integrity. Digitally signing an entry allows users who receive the entry to be certain of its origin, and that it hasn't been tampered with.

- **Support for policies:** V3 registries now support policies that outline authorization models, audit policies, and confidentiality policies, among others.

Information model extensions

- **Improved WSDL support:** Until V3, a WSDL document had no special status in describing a service. Through the introduction of the `useType` attribute to the `accessPoint` element, WSDL documents can now be queried and retrieved directly.
- **Added categorization capability for the `bindingTemplate` element:** Implementation details can now be searched using the same taxonomies used for businesses and services.

Extended discovery

- Support for complex queries to be consolidated into single queries through the use of nested queries
- New and extensible `find_*` qualifiers with extended wildcard support
- Management of large results sets

So What's Still Missing?

Taking all these improvements into consideration, the UDDI V3 specification provides an infrastructure that, in keeping with the spirit of UDDI, provides a flexible and extensible framework for Web services without actually specifying the detailed scenarios. However, two important issues have not been addressed by the specification. Consider the scenario where an organization is using a private UDDI registry to provide internal services for its IT department, and to provide a registry of interfaces for trusted business partners for the purposes of integration. This is probably the most common use of UDDI today and for the foreseeable future. For security and functionality, such usage would have to be through a proxied UDDI registry and would impose two additional requirements on any UDDI implementation.

- **Access control on the Inquiry API:** In keeping with the original intent of a "universal" registry, the current Inquiry

API is completely open, and only the Publishers' API is authenticated. In the scenario above, the organization will require an authenticated Inquiry API to control access to its registry, whether the requests are coming from inside or outside the firewall. This is a relatively simple requirement that can be added to the existing specification by adding an access control layer to the implementation.

- **Customization of the query returns to the requester:** Even an access controlled UDDI registry is not completely functional in the scenario above. Typically, different departments will have access to different services. And even when they have access to the same services, they might have different endpoints or they might expose different interfaces. This is even truer in the case of external partners. In these cases, providing some access control to the UDDI registry is only the first, and simplest, step in proxying it in terms of security. What is required is a mechanism to present different results to the same query based on the requester. For example, as a result of the same binding detail query, platinum-level partners will get a WSDL that exposes a richer interface on a particular service than gold level partners, and might even expose a different endpoint.

While these capabilities may seem minor they represent the bedrock of creating a secure and flexible Web services infrastructure. Without these capabilities there is no way of controlling and personalizing access to the various services published on the UDDI v3 registry, an important aspect of current and future Web services usage. The good news, however, is that when combined with a Web services security and personalization technology capable of proxying both WSDL and UDDI, UDDI v3 can be employed by enterprises to manage the entire service publishing and subscription lifecycle at the center of service oriented integration. ©

written by James Phillips & Heather Kreger

Toward Web Services Management Standards

An architectural approach
to IT system design



The work being done in WSDM will lay a firm foundation for effective distributed system management, both leveraging the unifying strengths of Web services in the solution itself and addressing the specific requirements for managing what are rapidly becoming the universal glue in enterprise system design.

The OASIS Web Services Distributed Management Technical Committee (WSDM TC) was chartered in March 2003 to recommend standards to address a problem that has been developing for many years. But with the widespread emergence of Web services and service-oriented architecture implementations in mainstream enterprise IT environments, the distributed systems management problem can no longer be ignored. The WSDM TC is focusing on two

distinct tasks as it attempts to solve some pressing distributed system management problems.

The first activity area, called Management Using Web Services (MUWS) addresses the use of Web services technologies as the foundation of a modern distributed systems management framework – including using Web services to facilitate interactions between managed resources and management applications. The same characteristics

that make Web services successful for application integration make them an excellent choice for use in solving the management integration problem – facilitating communications between managers and resources across numerous vendors, platforms, technologies, and topologies.

In addition to the use of Web services in the creation of a management framework, WSDM is addressing the specific requirements for managing Web services like any other IT resource. This activity is called Management of Web Services (MOWS). The manageability models that are being developed for Web services will be exposed using the techniques defined as part of the MUWS task.

Web Services Accelerate a Problem

For over two decades, there has been an evolution underway in the architecture of enterprise IT systems – a move away from big, unwieldy monolithic systems toward distributed architectures. Technologies such as DCE, CORBA, and DCOM were attempts to standardize mechanisms for software system interoperability. Client/server architectures – two-tier, three-tier, and then n-tier – were stepping stones along the path to what is rapidly emerging as the dominant

AUTHOR BIOS:



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 OASIS Web Services

Distributed Management Technical Committee, was colead of JSR109, Web services for J2EE environments and a member of the W3C Web Services Architecture Working Group. She is the author of *Java and JMX: Building Manageable Systems*.
KREGER@US.IBM.COM



James Phillips is chief strategist and senior vice president, products and marketing, at Actional Corporation. Actional offers a Web services management platform that is uniquely architected to help organizations

manage the impact of the constant change inherent in enterprise Web service networks. Actional's patented active-management technologies provide users with a complete view of the entire enterprise Web services environment.
JAMESP@ACTIONAL.COM

form of enterprise IT system design – the service-oriented architecture (SOA).

The SOA is a distributed system made up of software components that interact by passing messages to discoverable service access points. The SOA stresses interoperability, location transparency, and loose coupling.

While most consider the SOA an evolutionary step in the long road toward enterprise-scale distributed computing, the rate of adoption of SOAs based on Web services technologies is nothing short of revolutionary. This revolution is being fueled by the unprecedented vendor support of Web services standards. Web services deliver the required messaging and discovery building blocks for enterprise SOA deployments.

As Web service-based SOAs spring up across the enterprise landscape, they are accelerating and exacerbating a systems management challenge that has been growing in urgency in parallel with the development of enterprise-scale distributed computing.

In a monolithic application (or tightly coupled, client/server application), application boundaries are relatively clear and fixed. In an SOA environment, applications cross system (and even organizational) boundaries, they overlap, and they can change over time (see Figure 1).

Managing these applications is a serious challenge – but an absolute requirement. Failure or change of a single application component can bring down numerous interdependent enterprise applications. The addition of new applications or compo-

nents can overload existing components, causing unexpected degradation or failure of seemingly unrelated systems. Application performance depends on the combined performance of cooperating components and their interactions.

Effective systems and application management in a Web service-based SOA requires a management framework that is consistent across an increasingly heterogeneous set of participating component systems, while supporting complex aggregate (cross-component) management use cases, like service-level agreement enforcement and dynamic resource provisioning.

The rapid emergence of Web services and their acceleration of SOA adoption have made this need urgent – resulting in the charter of WSDM in March 2003.

Basic Structure and Components

Figure 2 highlights the basic structure and components of today's management frameworks. This generic model will be used to highlight the specific areas of management being addressed by WSDM and how they differ from traditional models.

At the bottom of the framework is a managed system that contains a collection of managed resources. A managed system typically contains multiple managed resources – for example, CPU, memory, disks, software components, and Web services. In an SOA, a key management framework requirement is the ability to understand the relationships between managed resources both within and across system boundaries. Management systems today do a relatively poor

job of building, maintaining, and leveraging information about resource relationships – particularly across managed system boundaries.

Next up the stack is the *management agent*. This component models managed resources and represents these resources in interactions with a *manager*. The manager always addresses and interacts with the management agent. The agent then interacts with managed resources.

A *management protocol* is the communication channel between the manager and the management agent. A management protocol is usually a two-way communication channel that allows the flow of operation requests and responses (e.g., get, set, and stop). It may use the same channel for passing notifications of events of interest.

In the distributed systems management context, it is increasingly important for management agents (and supporting management protocols) to define and support active capabilities versus traditional passive capabilities. For example, rather than merely raising an alert when a given Web service is unable to meet the performance requirements of a given service-level agreement, the management framework should be able to take corrective action. This action could take the form of rerouting requests to a backup service that is less heavily loaded, or provisioning a new application server with an instance of the software providing the service if no backup is currently running and available.

The manager is an application that communicates with management agents via management protocols, gathering information about managed system and managed resource status and performance, and supporting specific management tasks (e.g., root cause failure analysis, SLA monitoring and reporting, and capacity planning). It is this set of supported tasks that should drive (and is driving in the case of WSDM) the requirements for the rest of the management framework.

Existing Standards and Solutions Do Not Address the Problem

System and application management considerations have always been secondary to the creation and introduction of new systems and applications.

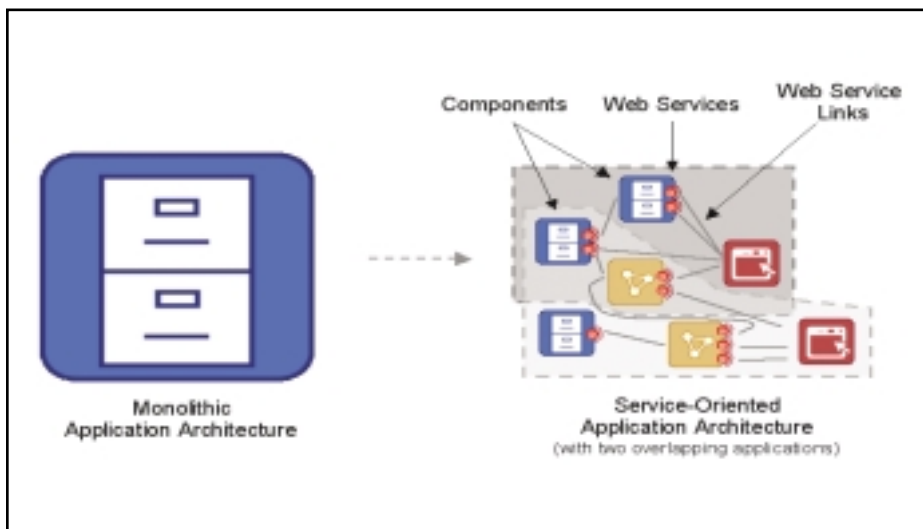


FIGURE 1 Monolithic versus service-oriented application architecture

Management standards and solutions typically trail the introduction of new approaches to IT system design. Over time, the result has been a proliferation of fractured, simplistic management standards and proprietary solutions that are usually targeted to solve point management problems. These point solutions have been specific to managed resources (e.g., storage, network infrastructure, software applications), management problems (e.g., failure detection and analysis, performance management, security policy enforcement), or specific vendor or technology solutions (J2EE application management, Windows server management).

In a world of complex distributed systems, these no longer suffice. To effectively manage a distributed application there must be a uniform mechanism for consistently managing its constituent components; understanding and managing component relationships; and managing the resulting aggregate system as a whole. In addition, emerging customer requirements for autonomic computing capabilities require management solutions to be far more active in nature – solving problems, not just identifying them. Management systems must be able to do all of these things for applications and systems across the enterprise, and across enterprise boundaries.

WSDM addresses many of these traditional management solution shortcomings (e.g., lack of integration across vendors and platforms, end-to-end information collection, agent dependency and passivity) through two distinct focus areas as highlighted in Figure 3.

Management Using Web Services (MUWS)

WSDM MUWS activity focuses on defining how Web services architecture and technologies can be used to manage any IT resource. In particular, MUWS will define how to describe the manageability capabilities of managed resources using WSDL documents.

An overarching requirement for MUWS is that it must use existing Internet infrastructure technologies and be compliant to the Web Services Architecture developed by the W3C WSA Working Group. Many current MUWS contributors were original contributors to W3C WSA activities – where management requirements and issues related to Web services were initially laid out.

Web services technologies offer substantial advantages in the context of distributed

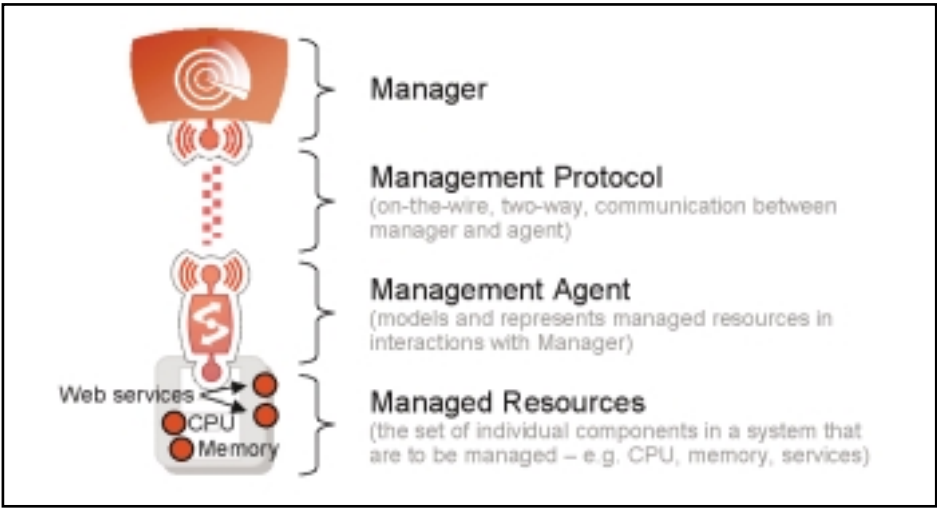


FIGURE 2 | Traditional management framework components



FIGURE 3 | WSDM activity in the traditional management framework context

systems management. They are language and platform neutral. There is tremendous industry momentum and support across a wide variety of system offerings. They offer on-the-wire interoperability and resource discovery with extensibility, separation of interface and implementation, and a rich metadata mechanism. In short, they allow MUWS to concentrate on developing specifications for problems specific to distributed systems management that require this set of infrastructure capabilities, without reinventing them.

There are two activities MUWS is *not* engaging in.

MUWS is not creating Web services technologies outside the scope of management. Required platform technologies will be obtained from the Web services community in the form of existing standards, specifications, and best practices. Examples are

security, policy, and reliable messaging.

MUWS is not creating a new model for representing managed resources. Rather, the requirement is that MUWS must be able to work with multiple, existing, domain-specific models. CIM from DTMF is the most prominent, but not the exclusive, model for this work. SNMP and OMI information models will also be considered. The objective is to bind a unifying layer on top of these various models to facilitate consistent management in a heterogeneous distributed environment. This on-the-wire unification of disparate resource models is the real power of Web services in the management context.

There is a variety of functionality requirements that the MUWS recommendation must address, including (note that these were preliminary as of July 2003):

- **Conformance and consistency with other**

International Conference & Expo

Edge 2004 EAST

Development Technologies Exchange

February 24-26, 2004

Hynes Convention Center, Boston, MA

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

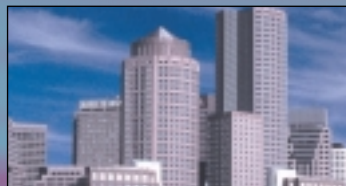
Addressing:

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

Register By
November 21, 2003
SAVE Up To
\$400



Hynes Convention Center
Boston, MA



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

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

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

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

Full Day Tutorials Targeting

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

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

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

management standards and existing managed resource models

As noted above, MUWS will embrace and extend the usefulness of, versus replace, existing management models and frameworks. MUWS will allow Web service-based access to managed resources already instrumented for existing management technologies. This increases the number of managers that can manage these resources without requiring the managers to support each management technology independently. In particular, MUWS is coordinating with DMTF, GGF, and the W3C. The Distributed Management Task Force (DMTF) offers management models (CIM) that will need to be described as manageability interfaces in WSDL. The Global Grid Forum (GGF) is working on solving dynamic resource sharing and provisioning problems. The GGF Open Grid Services Infrastructure (OGSI) defines how to represent and access IT resources as stateful grid services. OGSI also has the Common Manageability Model (CMM) working group, which focuses on representing manageability using grid services. There is much synergy and experience to be shared among MUWS, OGSI and CMM. The WSDM TC is working with other OASIS technical committees, including the security TC and provisioning TC.

- **Wide scope of manageable resource support:** MUWS will support the development of manageability interfaces for hardware and software resources, both physical and logical. A wide variety of management capabilities will be supported, including life cycle state and control, monitoring, and configuration management. Manageable resources can support these capabilities selectively and incrementally.
- **Distributed management:** MUWS must enable management of highly distributed environments, including supporting occasionally connected resources, resource aggregation, hierarchical managers, and multiple managers.
- **Variety of message exchange patterns:** Supporting both synchronous and asynchronous communication methods, one-way and two-way messages, and initiation of communication from either manager or managed resource (e.g. a manager queries for information using request-

reply model, managed resource provides one-way event notification).

- **Discovery:** MUWS manageable resources will be described through WSDL and XML Schema, making these resources discoverable like any other Web service.
- **Secure:** MUWS will enable secure communication between manager and managed resources, and is being explicitly defined to work across enterprise boundaries to support cross-organization management scenarios. One of the advantages of relying on Web services as a foundation is that MUWS doesn't need to explicitly solve this problem. Rather, it can reference security technologies and standards already available for Web services.

In addition to these functional requirements, MUWS must also enable interoperability, extensibility, scalability, usability, efficiency, and internationalization.

Management of Web Services (MOWS)

The WSDM MOWS activity is developing a model of a Web service as a manageable resource. The model requirements are being primarily driven by a set of management tasks that should be supported. These management tasks are numerous and include such varied activities as service metering, auditing, billing, performance profiling, SLA management, problem detection, root cause failure diagnosis, service deployment, and life cycle management. These management functionality requirements are the ultimate drivers of model requirements – they will highlight what Web service characteristics and management methods must be exposed to a manager. Explicit in MOWS is the requirement that it be described and accessible in a way consistent with MUWS.

Like MUWS activities, MOWS aims to build on existing model frameworks (such as DMTF CIM) in its definition of the management model of a Web service, rather than reinventing a general managed resource object model scheme.

In order to enable the expected management application use cases, the MOWS model will include (*note:* these were preliminary as of July 2003):

- **Identification:** Each modeled Web service must have a unique identification,

including a notion of version.

- **Life cycle/State:** Expose the current state of a service and permit lifecycle management including the ability to start and stop a service.
- **Metrics:** Must expose key operational metrics of a Web service, at the operation level, including such metrics as response time and throughput.
- **Configuration:** Will support the ability to make specific configuration changes to a deployed Web service.
- **Relationships:** Must permit the expression of relationships between and among Web services and other IT architectural elements and systems.
- **Types of resources:** Clearly, Web services are manageable resources in the MOWS model, but additional considerations are being made for modeling the scope in which a given service is being leveraged – individual, composite, part of a long-running business process.
- **Extensibility:** The model must be extensible and must permit discovery of supported management functionality in a given model instantiation.
- **Change description and notification:** Will support the description of versions of Web services and notification of a change or impending change to the service interface or implementation.

Conclusion

The adoption of Web services technologies in the foundation of SOA implementations is well underway in the enterprise. This architectural approach to IT system design brings substantial and well-documented business benefits. Consistent with historical behavior patterns, management is a secondary consideration in many of these implementations.

But in this architectural iteration, the costs associated with relegating management to the back seat are substantially higher than they have ever been. The nature of the SOA – complex system interdependencies – tends to magnify and spread system problems. What may have been a contained issue in a monolithic application environment, affecting a limited set of systems or business processes, can be a far-reaching and far more costly problem in an SOA environment.

The initial fruits of these labors are due in January 2004 as the Web Services Distributed Management v1.0 Specification. ©

A new tool for MX professional developers and designers...



ADVERTISE

Contact: Robyn Forma
robyn@sys-con.com
(201) 802-3022
for details on rates
and programs

SUBSCRIBE

[www.sys-con.com/
mx/subscription.cfm](http://www.sys-con.com/mx/subscription.cfm)
1 (888) 303-5282



MX
developer's journal





Reviewed by Brian Barbash

Sift 1.5 by Service Integrity

A strong analytical tool geared toward Web services

With Web services becoming more prevalent in organizations, keeping tabs on performance, analyzing problems, and managing overall quality of service is as important as ever. One tool that provides a monitoring and analysis solution specifically for Web services is SIFT 1.5 by Service Integrity.

SIFT is designed to seamlessly integrate with existing Web services to provide extensive runtime statistics. At the time of this writing, the product was only available for applications built with Microsoft .NET; however, versions that will monitor Java Web services from various application-server vendors are now available.

Application Architecture

There are three main components to the SIFT application: Application Modules, Stream Analyzer, and Stream Sensors. The Application Modules provide administrators with a GUI interface to collect, manipulate, and analyze a wealth of performance-related data for all available services. The Stream Analyzer inspects XML data on the wire as it passes from client to server. Finally, the Stream Sensors are components installed into IIS (and other application servers as future versions are completed) that intercept data necessary for the SIFT application.

Using SIFT

SIFT installation is straightforward and easy. SIFT requires Microsoft Windows 2000 SP3 or Windows XP Pro SP1 or higher, 128 Mb of RAM, 10 or more Mb of hard disk space for logging, the .NET Framework 1.0 SP2, IIS with ASP.NET support, and the Visual J# Redistributable package 1.0. For the server side, the installation procedure creates a new

Web service on the host machine that the SIFT GUI communicates with. This service hooks into IIS to collect runtime statistics on all services running on the machine. The client install provides the SIFT Console for administrators to perform all analytical and monitoring tasks. It should be noted that the console need not reside on the machine that is being monitored.

Setting Up

When the SIFT Console is launched, the user is presented with an option to add remote servers to monitor. Once a server address is provided, a list of available Web Services is presented. Any number of services may be selected for monitoring. For this review, I've created a simple Web service to be monitored that displays summary info and line item data in a basic financial ledger.

Figure 1 shows the Services Group of the SIFT Console that displays all configured host machines. Under each machine, the list of services to be monitored is shown and the rules for generating the logs may be edited.

Logging rules define the file size, rollover frequency, and archiving constraints; and the fields to be captured. In addition to the set of fields provided by SIFT, custom field specific to the services monitored may be added. For example, if an error condition is encountered in business logic, an application-specific error message might be sent to the calling client. A custom field may be defined in SIFT to locate elements within generated error messages to track the frequency of the business error. Once the services to be monitored and all logging rules have been set up, SIFT is ready to go in its default state.

Collecting Data

The Dashboard group in the SIFT Console provides graphing capabilities for viewing archived and real-time performance infor-

Author Bio:



Brian R. Barbash is the product review editor for Web Services Journal. He is a consultant for the Consulting Group of Computer Sciences Corporation, where he specializes in application architecture and development, business and technical analysis, and Web design.
BBARBASH@SYS-CON.COM



COMPANY INFORMATION

Service Integrity
199 Wells Avenue, Suite 107
Newton, MA 02459
617-965-0281
e-mail: sales@serviceintegrity.com

LICENSING INFORMATION

\$2000/server
\$500/developer seat

REQUIREMENTS

Windows 2000 SP3, Windows X Pro SP1, or higher
128 Mb RAM; 10 or more MB hard disk space
.NET Framework 1.0 SP2, IIS with ASP.NET support
Visual j# Redistributable package 1.0



mation. By default, SIFT monitors a host's average response time, the amount of traffic, and the total operations for all hosts. Graphs and charts may be organized into individual dashboards to group related information cleanly.

Custom graphs provide greater insight into the executing services. SIFT provides a wizard interface to construct graphs from any of the logging fields configured in the Services group. Data may be presented in either a time-based or metric versus metric graph. Time-based graphs present data as a line chart while metric versus metric graphs may be line charts, pie charts or bar graphs (see Figure 2).

Within each chart, up to eight series may be created, each assigned to a unique variable in the log file. In the metrics versus metrics charts, data represented by each variable may be aggregated using a standard set of functions including Sum, Average, Count, Maximum, Minimum, and Standard De-

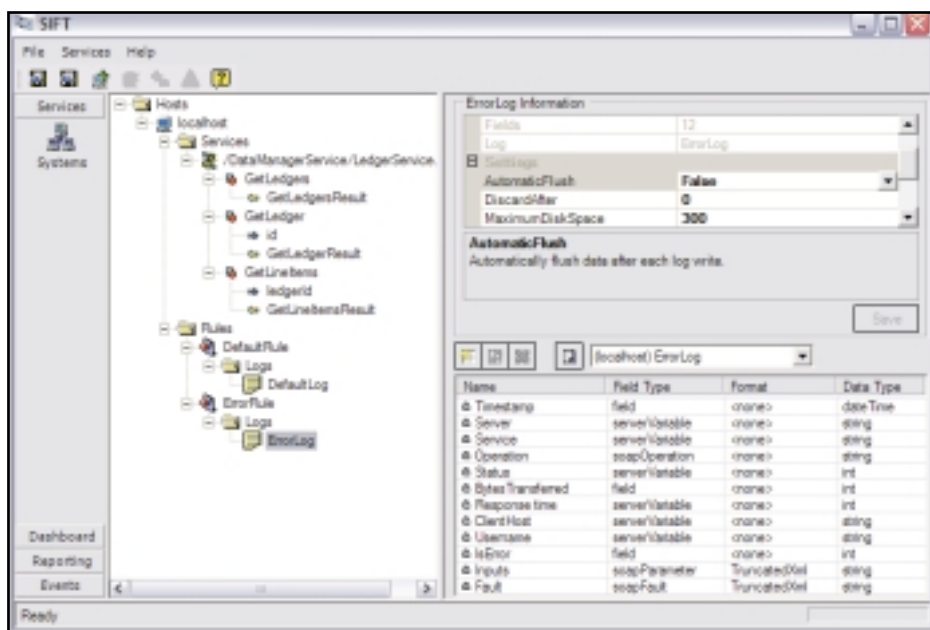


FIGURE 1 | SIFT Console – Services Group

viation. Data may be further focused by adding filters to the values in the log fields graphed. Values may be constrained by the top 10 or top 25, or a Regular Expression; or they may be broken up into numeric ranges to produce a histogram. Once the charts are set up, they may be toggled to collect live data at a configurable interval or to look at a set of frozen historical data.

Reporting and Analysis

SIFT provides three types of reports that may be generated for statistical analysis:

- **Performance:** Indicates the number of requests the system serviced and the response times for those requests
- **Traffic:** Presents the number of bytes transferred to and from one or more hosts
- **Usage:** Reports on the total number of hits a particular Web service received

Each report may be configured to look at a specific time interval.

Events and Notification

In addition to providing analytical and performance monitoring capabilities, SIFT includes functionality to create and send SNMP traps. Created in the Events group, Alerts may be configured to send SNMP traps based on constraints established for metrics within a service log. They may be set up for four metrics:

- **Average response time:** The average response times for all requests over a minute
- **Error rate:** The percentage of the total number of requests over a minute that produced an error
- **Operation count:** The number of operations that occurred over a minute
- **Total bytes transferred:** The total amount of data transferred over a minute

Setting up an alert involves establishing the minimum and maximum value thresholds for a log metric and the number of consecutive violations against those thresholds. To assist the administrator in determining the appropriate constraints for an alert, SIFT provides a graphical representation of historical data. Highlighted in the display are the boundaries of the alert's data thresholds. As the minimum and maximum values are adjusted, the display updates the highlighted range and displays the total number of alerts that would be generated from the current settings. Once the minimum and maximum

values are set, the count of consecutive violations must be specified. When complete, SIFT will create and send SNMP traps whenever the monitored Web service executes outside of the boundaries configured in the Alert item.

Summary

Web services continue to permeate throughout organizations to expose business logic to client systems. More and more services are being exposed across multiple physical systems, thus enhancing the need for effective monitoring and analysis. Service Integrity's SIFT provides a strong analytical tool geared specifically to Web services to assist administrators in this task. ©

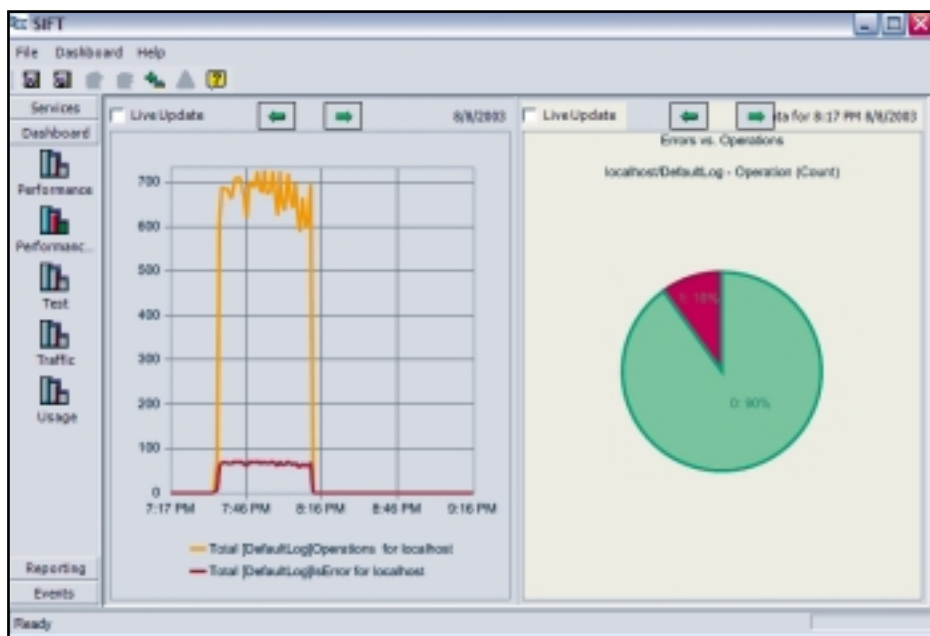


FIGURE 2 | Left-side graph: Time based showing total errors and total operations. Right-side graph: Metric versus metric showing errors versus operations

Knowing When to Use Web Services

A case study of an implementation that worked



Web services are moving from the latest buzzword to a mature and accepted technology. Mainstream companies such as Eastman Chemical, Wells Fargo, and NEC have begun deploying significant Web Services-Based Integration (WSBI) projects. Avnet Computer Marketing (Avnet CM) is one of many companies also betting heavily on Web services. This month, "Web Services in the Real World" describes Avnet CM's strategic foray into WSBI (see sidebar). We'll explore their business objectives, why they chose Web services for some parts of their architecture (and not others), and the results they achieved.

Background

Avnet Computer Marketing (Avnet CM) markets enterprise technology products from the world's premier computer manufacturers and software suppliers. Customers include value-added resellers (VARs) and enterprise customers, and Avnet CM provides them with marketing support, pricing strategies, and supplier-relationship management. Avnet CM is an operating group of Avnet, Inc. (NYSE:AVT), a Phoenix, Ariz.-based Fortune 500 company. Avnet is a technology marketing and

services provider, and one of the world's largest distributors of electronic components and computer products from industry leading manufacturers.

Avnet CM turns to Web Services

Customer: Avnet Computer Marketing is an operating group of Avnet, Inc. (NYSE:AVT), a Phoenix, Ariz.-based Fortune 500 company. Avnet is a technology marketing and services provider and one of the world's largest distributors of electronic components and computer products from industry leading manufacturers.

Challenge: Consolidate and standardize application interfaces; implement a service-oriented integration architecture to speed time to market.

Solution: Avnet's e-business portal allows customers and suppliers to configure and place orders, retrieve real-time pricing & availability information, and view their order status. Web Services-Based Integration (WSBI) feeds the portal with data from various back-end systems.

Why Web Services: Avnet chose WSBI primarily for three reasons:

- Simplicity
- Reusability
- Abstraction

Key Business Benefits: By using WSBI, Avnet benefits from:

- Reduced development and maintenance cost
- IT agility
- Faster time to market

The Challenge

Avnet CM manages Avnet's Hall-Mark e-business portal that allows customers and suppliers to configure and place orders, retrieve real-time pricing and

availability information for products, and view their order status.

To provide customers and suppliers with a broad range of online services, Avnet CM needed to connect the portal to various back-end systems using a variety of protocols and file formats. "Maintaining this growing number of proprietary interfaces became untenable and prevented us from being able to respond quickly to new market opportunities," said Bud Alexander, vice president of Enterprise Integrated Solutions. He wanted to build a responsive IT infrastructure that allowed him to

- **Reduce the cost** of development and maintenance by consolidating and standardizing internal system interfaces, and
- **Speed time to market** by maximizing interface reuse.

The Solution

Alexander's strategy was to wrap applications with business-oriented WSDL interfaces, creating common business services, and then to pull data from multiple applications into one "Business Service Hub." (This is one of the usage patterns identified in the first article of this series, "Patterns in Web Services Projects"; *WSJ*, Vol. 3, issue 5). The goal was to consolidate interfaces and reduce the number of connections among systems. The architecture was required to maximize the potential for reuse, and the services had to be accessible from anywhere using any technology.

With this framework in place, Alexander hoped to reduce system integration maintenance by combining and reusing connections among systems. Such a model would speed integration by reusing Web service components that were already built.

The Architecture

Three integrat-



Author Bio

Michael Blank is a founding member of webMethods, Inc., and was its first software engineer. During his tenure, he has started and commercialized several product offerings. As director of developer marketing, he manages webMethods' developer communities as well as the software evaluation program (<http://evals.webmethods.com>).
MICHAEL@WEBMETHODS.COM

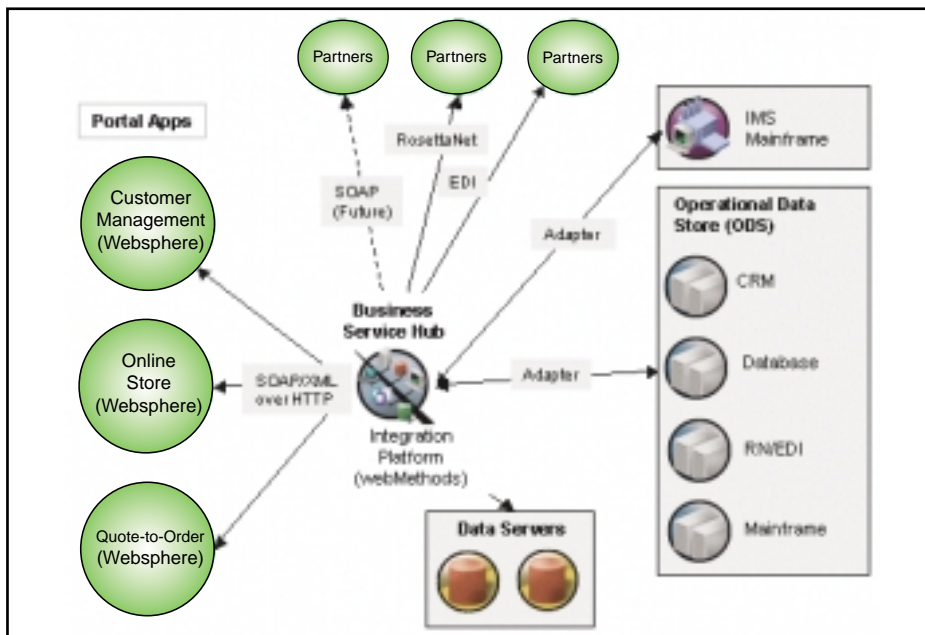


FIGURE 1 Business Service Hub architecture

ed portal applications, each developed and maintained by separate teams, provide the following capabilities to Avnet's sales force, customers, and suppliers:

- **Quote to Order:** Intranet application used by Avnet sales to provide customers with product quotations and to expeditiously convert them to orders
- **Channel Connection:** Customer portal that lets Avnet's authenticated VARs and customers retrieve real-time order status, leads, sales, credit history, and key performance information
- **Customer management:** Internal portal that allows customer service representatives to centrally view and update customers' account information

Avnet CM implemented the Business Service Hub with a Web services-based integration platform (see Figure 1). The portal applications communicate with this integra-

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

BUY THOUSANDS
OF PRODUCTS AT
GUARANTEED
LOWEST PRICES!

GUARANTEED BEST PRICES
FOR ALL YOUR
WEB SERVICES
SOFTWARE NEEDS



N-ARY

\$299.00

n-ary Ticket System v2.0

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



ZION SOFTWARE

\$995.00

JAlerts' Deployment (JIMessageServer)

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



GREENPOINT INC.

\$1,350.00

WEB CHARTS 3D VER. 4.7

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



JALERTS

\$1,495.00

JAlerts Deployment (ICQ)

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



CHUTNEY TECHNOLOGIES

\$755.25

Chutney SOAP+ Toolkit

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



INFRAGISTICS

\$495.00

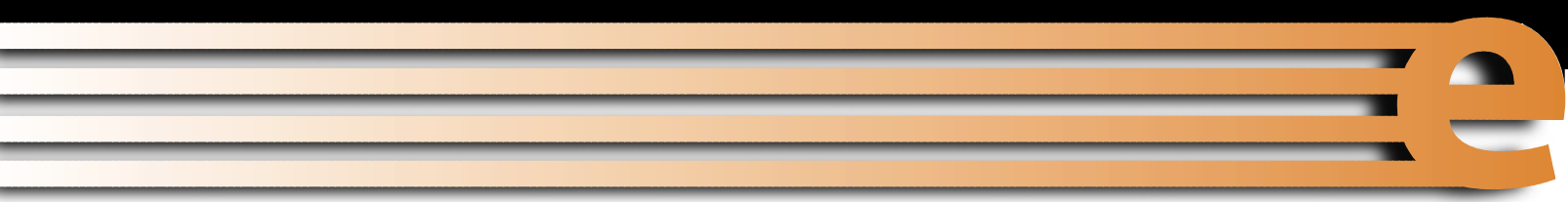
Net Advantage Suite 2003 Volume 2

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



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

OFFERS SUBJECT TO CHANGE WITHOUT NOTICE



tion platform via SOAP to access data from the following back-end systems:

- **The IMS mainframe** processes orders and performs other sales order management functions;
- **The operational data store (ODS)** consolidates customer data to create a single view of the customer. The ODS is exposed as a simple set of Web services that aggregates information from a CRM system, database, mainframe, and external data from suppliers and manufacturers (using RosettaNet and EDI).

The architecture also includes trading partner gateways for processing orders. Avnet CM currently supports RosettaNet and EDI transactions, and is planning to add a SOAP gateway in the future.

Why Web Services?

Avnet CM used Web services between the portal and the integration platform for the following reasons:

- **Simplicity:** Even nonprogrammers are able to assemble Web service-based integration solutions using the integration platform tools. According to Alexander, "Users do not need expertise in SOAP or WSDL, so I don't need to hire senior messaging experts to develop integration services. I could put anyone on my integration team, no matter what their experience level, onto Web services projects." In addition, the portal's application server can readily consume SOAP messages. This made the integration easier, faster, and cheaper.
- **Abstraction:** The portal team is organizationally separate from the IT team, which owns the back-end systems. The IT team wanted to make it easy for the portal team (and others in the future) to access information without having to know about the complexities and data formats of the back-office. "The portal content team can focus on the presentation layer and need not be proficient in databases, mainframes, and CRM systems. They simply call a Web service using whatever technology they already know," says Alexander. The contract between the teams is the WSDL interface.

- **Reusability:** Reuse was critical. Other teams needed to access the same capabilities. Once a connection has been made and exposed as a Web service, it's easy to connect other applications.
- **Performance and reliability:** Even though Avnet CM processes a high volume of orders from their web site, they found performance to be satisfactory. Alexander adds that they have never lost a single order.

These reasons for using Web services are consistent with why other companies said they chose Web services for integration (see "Why Web Services Work"; *WSJ*, Vol. 3, issue 7). Interestingly, Alexander did not use Web services for the entire project. Specifically, his team did not use Web services between the integration layer and the mainframe and CRM systems. Instead, they used adapters and native APIs. Here's why:

- **Back-end ownership:** Alexander's group is responsible for the back-end systems they are integrating with. This means they had the necessary domain expertise, had direct access to these systems, and could control the technology to access these systems. Web services, on the other hand, are better at facilitating the integration between different organizations because they provide an abstraction layer, define a contract (the WSDL interface) between the groups, and let each group use whatever technology they want to access the interface.
- **Lack of SOAP support:** The mainframe's SOAP support was inadequate. The CRM systems actually consisted of four separate applications, and not every one of these had native SOAP support. On the other hand, the four systems all supported APIs and adapters.
- **Performance and reliability requirements could not be satisfied with Web services:** Each service consisted of a complex set of transactions on the back end that required transactional integrity, which was managed by the integration platform.
- **No reuse required:** The mainframe and CRM APIs were never meant to be accessed directly. Thus, reusing the APIs directly was not a requirement. They were instead rolled

up into composite applications that were exposed to the outside world as a simpler set of Web services that were meant for reuse.

The Results

With Web services connecting several back-end systems with their portal applications, Avnet CM realized the following benefits:

- Reduced cost of maintaining and extending the integration architecture by consolidating the interfaces and standardizing on Web services. Avnet CM has been able to successfully retire their proprietary interfaces (custom XML, FTP, and IP sockets) in favor of Web services.
- Faster time to market with improved IT agility. New projects are now measured in days instead of weeks. Previously, integrating the quote-to-order application with the IMS mainframe required approximately six weeks. But, because the Web service was already built, connecting the order entry e-business system took only one day.

What's Next

In the future, Avnet CM will extend the Web services framework to its customers and suppliers. "Web services continue to be central to our IT strategy. The investment in our Web serviced-based integration platform allows us to take advantage of our IT investments and to deliver greater value for the company," concludes Alexander.

Conclusion

Companies like Avnet CM prove that Web services are maturing as an accepted technology. At the same time, it's clear that Web services are not the silver bullet to solving complex integration problems, either. The trick is figuring out when to use Web services, and when not to. Based on the ROI realized by Avnet and other customers profiled in this column, Web services, and the service-oriented architectures that support them, are increasingly becoming a key component of any successful company's integration strategy. ©

WebServices

.NET J2EE XML JOURNAL

LEARN WEB SERVICES. GET A NEW JOB !

SUBSCRIBE TODAY TO THE WORLD'S LEADING WEB SERVICES RESOURCE

The Best
.NET
Coverage
Guaranteed!

Get Up to Speed with the Fourth Wave in Software Development

- Real-World Web Services: XML's Killer App!
- How to Use **SOAP** in the Enterprise
- Demystifying **ebXML** for success
- **Authentication, Authorization, and Auditing**
- **BPM** - Business Process Management
- Latest Information on **Evolving Standards**
- Vital technology **insights** from the nation's leading Technologists
- Industry **Case Studies** and **Success Stories**
- Making the Most of **.NET**
- **Web Services Security**
- How to Develop and Market Your Web Services
- **EAI** and Application Integration Tips
- **The Marketplace**: Tools, Engines, and Servers
- Integrating **XML** in a Web Services Environment
- **Wireless**: Enable Your **WAP** Projects and Build **Wireless Applications** with Web Services!
- Real-World **UDDI**
- **Swing**-Compliant Web Services
- *and much, much more!*

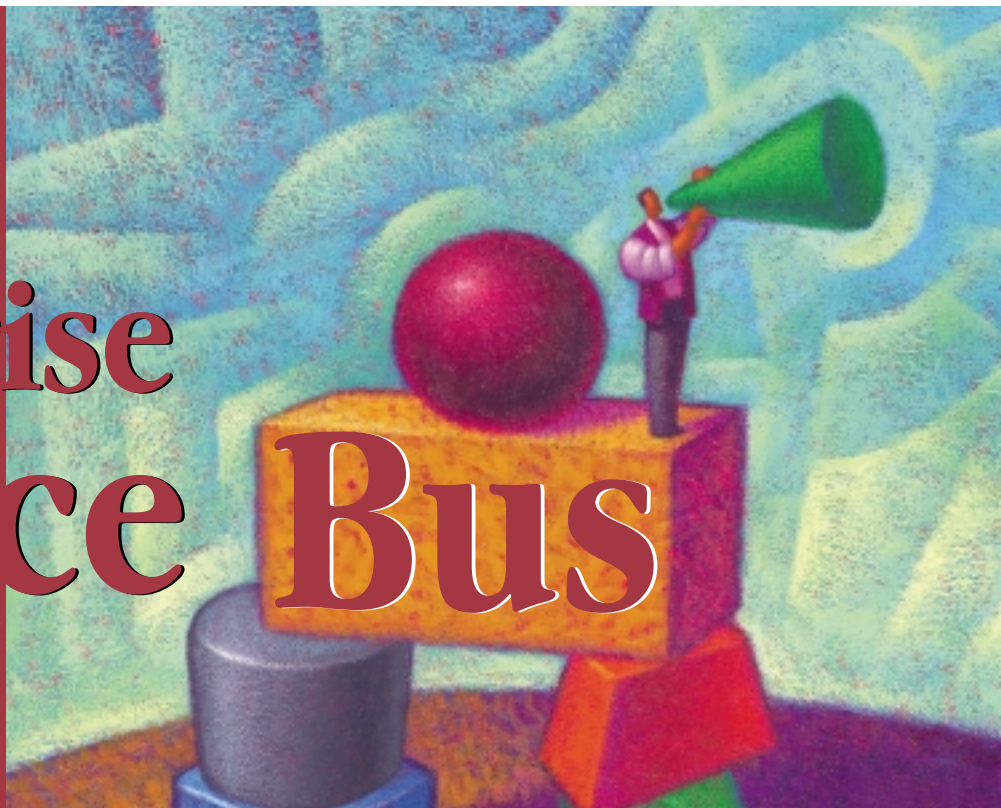
Only \$69.99 for
1 year (12 issues)*
*Newsstand price \$83.88 for 1 year
Subscribe online at
www.wsj2.com or
call 888 303-5252
*Offer subject to change without notice



written by Nigel Thomas & Warren Buckley

The Enterprise Service Bus

A developer-friendly integration engine



In the past there seemed to be two more or less exclusive routes to integration: “roll your own” or buy an EAI product. Typically, developers would choose the first option for maximum flexibility, while project managers preferred the second, for consistency and security.

Now, XML and Web services standards can offer lower cost options for enterprise integration, and have helped to promote the emergence of a new class of integration tool, the Enterprise Service Bus (ESB). Over the last year the ESB has emerged as a “middle way” between these two approaches, providing a developer-friendly integration platform, open to all the latest Java standards and components, without sacrificing the benefits of a packaged approach to integration.

This article goes behind the scenes to

look at why an ESB can be a useful part of your integration toolbox, and looks at some of the key features and qualities that an ESB built using Java technology must have to fulfill its promise as a capable and cost-effective XML integration engine.

Growing Demand for Integration

Over the last decade and a half, the integration market has grown from a few low-level file-transfer and screen-scraping products to a massive software industry. Message-oriented middleware; extract, transform

and load (ETL) tools; and EAI integration brokers have emerged, driven by the inexorable spread of the relational database and packaged business applications.

At first, business applications addressed individual functions in the organization – accounting, manufacturing, and payroll, for example. But it didn’t take long for IT departments to appreciate the benefits of sharing information between these separate systems. Programs were written to transfer key information, at first on a monthly or weekly basis. Moore’s law – which has increased the speed and decreased the cost of computing power – has accelerated that process to hourly and even real time. The “zero latency enterprise” has arrived.

More and more, today’s businesses need agility – they want to be able to modify systems quickly and cost-effectively by simply changing how a few components interoperate, rather than by building or buying whole new systems. Most of that demand is tactical, with rapid project timescales and lower project costs; return on investment in months rather than years is the order of the day. And now, even small businesses can afford well-integrated computer applications – something that was a costly luxury even for large organizations 20 years ago.

The economics of integration product

AUTHOR BIOS:



Nigel Thomas offers independent product marketing consultancy in the application infrastructure software marketplace. He recently spent two years as director of product management for SpiritSoft’s Java messag-

ing, caching, and integration products. Prior to that, he spent five years with EAI pioneer Constellar as product architect, and then as director of product management for the flagship Constellar Hub product.

NIGEL.THOMAS@LYNTONRESEARCH.COM.



Warren Buckley is chief technology officer and cofounder of PolarLake, and has been working in the area of XML and Web services since 1998. Warren was responsible for the development of a number of patent-pending approaches

to optimizing XML processing that underpin PolarLake’s enterprise-strength application integration products. He was previously CTO of XIAM Ltd, a leading mobile middleware company, and systems architect for Bank of Ireland Group Treasury.

WARREN.BUCKLEY@POLARLAKE.COM

development have also changed. Application vendors are increasingly conforming to standards like Web services and the Java Connector Architecture (JCA); a small industry has grown up to offer adapters to so-called “Enterprise Information Systems” (EIS; the generic term for applications and databases). Removing the burden of EIS adapter development has moved the focus towards the EAI vendors’ traditional weak spots: transformation, developer-friendly programmatic processing (in Java, for example), and integration to external technologies. This has opened the integration market to more nimble newcomers, and has led to the emergence of the so-called Enterprise Service Bus category of products.

What Is an ESB?

There is no standard definition of exactly what features an ESB should support, but there is wide agreement over the general outline. An ESB will offer most of the following:

- **Event-driven, document-oriented processing model:** Based on XML standards it delivers an asynchronous service-oriented architecture
- **Content-based routing and filtering**
- **Complex transformation capability**
- **Support for several standard interfaces:** From a list including COM, CICS, .NET, JMS, JCA, JDBC, and, of course, Web services
- **Distributed operation and management:** Rather than a centralized integration hub

Service-Oriented Architecture

Perhaps the oldest “new thing” in the equation is the service-oriented architecture. First conceived alongside technologies like DCE and CORBA, service-oriented architecture separates the core of an application into a series of fairly coarse-grained components – chunks of business logic that provide “services” to client programs. These services can be shared between presentation (GUI) and integration applications (for example, as steps in a business process).

Processes communicate only through documented interface “contracts,” which reinforce the design principles of modularity and encapsulation. Hiding the details of how each module works – and tying down its specification – makes it easier for external developers to reuse the component.

A particular class of service-oriented architecture uses complete documents – business messages if you like – to trigger each step in an overall business transaction. That’s because there may be no single database that the process steps can all share. This style of interaction can be called “document-centric processing,” to distinguish it from the database-centric model commonly used to build internal systems such as ERP and CRM. The document – the message passed from system to system – contains all necessary data items, and takes the same role as input parameters in a well-structured function call. Indeed, it is more flexible; because the document encapsulates all necessary data, there is no need for each step to know which specific data items are needed by the next step.

Using message-oriented middleware, it’s easy to plug services together. And because it’s easier, that makes it cheaper too. Interfaces tend to be simpler and cleaner. An enterprise can assemble systems from best-of-breed pieces, which may be hosted in different departments or even companies. There is a clear separation of concerns; components – each of which can be owned, developed, and deployed autonomously – can be separately scaled, replicated, or replaced. Different services can have different development life cycles, languages, and platforms. An enterprise can mix and match its 30-year-old main-frame systems – using IBM WebSphere MQ to kick off CICS transactions – with J2EE/Unix, Windows/.NET, and any other legacy processing (see Figure 1).

Easier integration also promotes more frequent reuse rather than redevelopment of components. When millions of dollars are invested in software assets, the last thing a business wants is to trash them just because

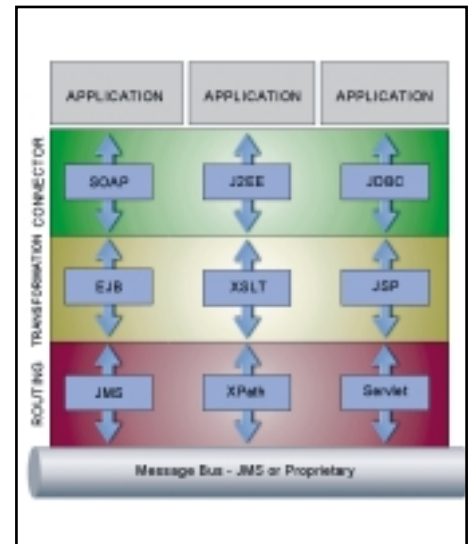


FIGURE 1 Relying on open standards the Enterprise Service Bus allows a wide range of deployed software components to be integrated

it doesn’t match their current preferred platform.

Content-Based Filtering and Routing

A key requirement of any integration platform is to identify which data to process and where it needs to be sent. Selection rules – typically specified using XPATH – identify the key sections of an XML document. These sections can be extracted and routed to processing components, and the results delivered to the next processing stage, based on its content. For example:

```
/order/item[price>10.80]
```

will select all order items with a price greater than 10.80, while

```
/order/[count(item)=1]
```

selects all orders that are for just a single item.

“XML and Web services standards can offer lower cost options for enterprise integration, and have helped to promote the emergence of a new class of integration tool”

There's real value in exposing these selection rules as configuration metadata rather than simply hiding them in code. Business requirements are changing faster than applications can be created and modified. Rules offer a way of encapsulating the business semantics and promoting them to the surface, where they are much easier and cheaper to manage.

Transformation Capability

As it moves from system to system, data needs to be validated, translated, and enriched by calculations or lookups. These operations may be performed by the ESB itself, or they may be delegated to external systems (by synchronous or asynchronous calls). The ESB offers an environment in which these actions can be processed independently (and concurrently) using "actions," which are processing rules written in a range of scripting or declarative languages. XSLT is one common approach. But although XSLT engines are easily available at no cost, the engines can be rather slow and XSLT itself struggles to achieve the kind of complex transformations needed for application integration. ESB vendors need to add their own more powerful transformation engines so that documents can be processed swiftly and efficiently and then reassembled into one or more output documents that can be routed – based on their content – to the next step in an orchestrated sequence.

One important transformation capability is "any to XML" – translating legacy formats (EDI, structured files, and relational data) into XML for processing and onward transmission, and from XML back to legacy formats where required. Because of its semantic power, XML provides an excellent interchange and internal processing format even for non-XML inputs and outputs. The ESB executes these syntactic transformations at the edge – as documents enter and leave the engine – ensuring that a single technology can be used for core processing.

Support for Standard Interfaces

Of course, the ESB will support Web services interfaces, together with JMS for enterprise messaging, JCA for applications, and JDBC for databases. Widely

adopted "industry standards" – proprietary interfaces like IBM WebSphere MQ messaging and CICS transaction monitor, and Microsoft's MSMQ messaging, for example – may also be supported. Increasingly, applications and services will communicate pragmatically, based on widely used protocols such as POP3/SMTP and instant messaging, as well as HTTP, SOAP, and MOM. This makes it easier to manage and share infrastructure between applications and humans.

Once any middleware (such as a MOM) has been deployed, it can be hard to replace or upgrade. That's why so many organizations have two or more existing middleware products deployed, and why it is such a big mistake to tie an ESB to any single vendor's MOM. An ESB needs to adapt to the realities of today's businesses: bridging between multiple MOMs, as well as bridging between asynchronous (messaging) and synchronous (RPC – Remote Procedure Call) domains. An ESB is there to support and integrate whatever is already in use, not to replace it.

Distributed Operation and Management

Early integration products tended to follow a hub-and-spoke architecture – a straightforward approach that simplifies integration topology and centralizes configuration and management. In a batch environment, the slight risk of failure could be insured against by operating a standby hub for cold failover.

As software pervades the modern business, and interchange frequency increases, the centralized approach is no longer tenable and the costs of interruption are too great. Businesses need distributed networks with no single point of failure – to optimize network traffic, to provide alternate routes in case of failure, and to distribute the processing load. Offering services closer to participating applications – where possible – spreads the processing load as well as reducing network bandwidth utilization.

Of course this complicates system management, so it must be possible to manage today's globally distributed ESB from a single operational console, and to adjust its configuration without bringing

down the entire integration network. Downtime must be kept to a minimum – remember that "five nines" (99.999%) availability allows for just five minutes for downtime per year, and even four nines leaves less than an hour.

What's in It for Developers?

Developers have often disliked integration products, preferring to use frameworks of their own devising. The main reasons given for this are:

- **Original cost of licenses and training:** Understandable when products were inflexible, and product licenses came in at six- or even seven-figure numbers.
- **Inflexibility of the purchased product:** Sometimes it seems you have to spend as much time working around the product as working with it.
- **Poor interoperation with other development and management infrastructure:** When all your deliverables must go through development and testing into production configurations, you need a product that's open to development environments and automated configuration management.

However, there are long-term implications in adopting a "roll your own" policy. Will the organization be able to support the framework going forward? Use of a product can deliver significant productivity gains, reducing development time and cost by allowing the developer to focus on business logic rather than framework building or infrastructure coding, so that he or she needs only to write a fraction of the code normally required. Moreover, the code is fully open and transferable and no new skills are required, enabling rapid adoption and return on investment. Now that license costs have fallen, and products are much easier to work with, there's really no excuse for taking this kind of risk.

Existing developers and business users can rapidly deliver new solutions with minimum disruption to existing systems and maximum leveraging of existing assets and skills. XML-based ESB products can complement and readily integrate with familiar IDEs and configu-

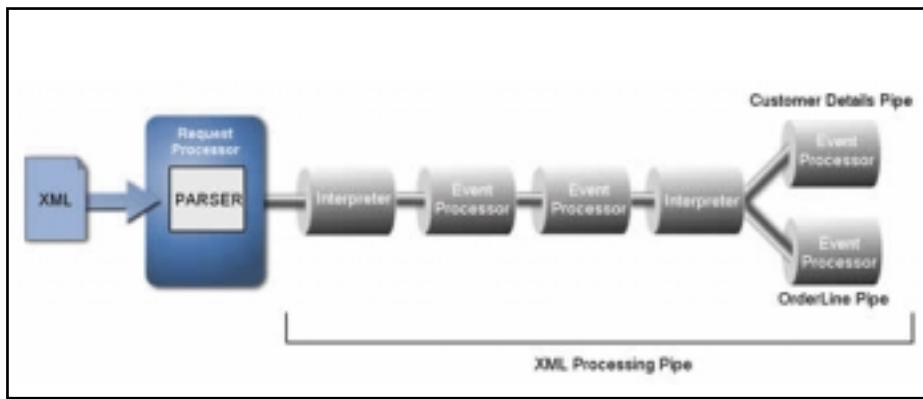


FIGURE 2 | Multithreading

ration management systems; this is much less intrusive on the development process.

Challenges for XML Processing Architectures

One perceived impediment to the exponential uptake of XML is that it can deliver a rather verbose representation of a chunk of data. Mushrooming quantities of data being exchanged present a serious challenge to corporate infrastructure. This is not caused just by the addition of all those “human readable” tags. XML succeeds precisely because its structure is so flexible and extensible. Just as the introduction of the relational model made it easy to add new tables and columns to a database, so XML’s syntactic power allows organizations to refine and specialize XML Schemas to handle all of their specific information needs. If necessary, entire document sub-trees can be added onto a standard schema without breaking third-party functionality. By encapsulating all contextual information needed by a series of business processes, XML encourages the development of loosely coupled, “document-oriented” processing. New systems are easier to plug in as messages can be extended to include all required data – there’s no need for a multiplicity of calls back and forth for additional items.

First-generation XML integration pilots successfully demonstrated the potential of the technology, but they are under increasing strain as the quantity, size, and complexity of XML documents continues to grow. XML cannot be adopted for core systems until these problems are

recognized and addressed. Among the causes of poor scalability are:

- **Whole document parsing:** You need to parse entire documents just to extract a snippet of content for routing and filtering. As documents get bigger, this results in increasing latency.
- **Multiple scanning:** Documents are often re-parsed at every stage in a business flow, with the same document being scanned several times by parsers, XSLT routines, conversion to object networks, etc. This is exceptionally resource intensive, with a major impact on performance and throughput. Working around this problem by passing documents as object networks just tightly couples every step to the chosen representation, aggravating the cost of development and maintenance.
- **Single-threaded execution:** A processing step cannot start until the previous step is completed; latency increases as everything daisy-chains at the pace of the slowest step.
- **Cut and paste development:** Dealing with multiple nonidentical sources results in identical elements of processing being replicated into many processing steps; XSLT processing is hard to modularize. The number of similar transformations adds hugely to the development and maintenance costs, and impedes new business development by lengthening time to market.

Routes to Scalability

What is needed is a processing architecture that helps the architect/developer organize:

- **Document streaming:** Ensures that XML documents are processed as each element arrives, thereby ensuring very low latency; this approach handles large messages as efficiently as small ones.
- **Selective Processing:** Dramatic performance improvements come by processing (and handing around) only relevant fragments rather than the entire XML document.
- **Multithreading:** The engine manages pipelining of sequential steps, parallel execution of independent steps, and load balancing of identical steps while processing these multiple XML fragments (see Figure 2).
- **Single scanning:** Extracting all interesting document fragments in one pass up front, rather than repeatedly re-reading the same document structure.

A broker that can manage these techniques without requiring expert coding and configuration massively reduces the risks of project failure and consequent business damage caused by inadequate performance. Core applications, previously out of bounds by reason of data volume and complexity, can now be confidently addressed.

Conclusion

There has been a tectonic shift away from complex, proprietary, centralized, and costly integration brokers toward more lightweight, distributed, standards-based and inexpensive enterprise service bus technology; this has been driven by the adoption of open standards like XML, Web services, and J2EE.

The ESB offers a powerful and extensible integration platform that supports your development aims without forcing you to adopt proprietary technology, retrain your staff, or radically change your development methodology. By leveraging the substantial community investment in XML and other technologies, ESBs can equal or better the capabilities of earlier integration brokers at a far lower cost, bringing integration capabilities to smaller businesses and more marginal projects. Easy adoption of familiar technologies further reduces the cost of uptake and helps maximize return on investment for any size of business. ©



Rich Rollman

Rich Rollman is a senior engineer on the BEA WebLogic Integration team and the Specification Lead for JSR 207 (Process Definition for Java). He has been involved with the development of XML and Web services since their inception and has 20 years of experience with client/server computing.

RICHARD.ROLLMAN@BEA.COM



William Cox

William Cox is a technical director in the BEA CTO Office, concentrating on transactional and portal architecture. He is a coauthor of BTP, the BTP Primer, and WS-Transaction. He holds a Ph.D. in computer sciences from the University of Wisconsin-Madison.

WILLIAM.COX@BEA.COM

Reproduced with permission from BEA Systems

Transactions in Business Processes – A New Model

Transactions are a common, everyday occurrence in our lives. When you buy something at the grocery store, transfer funds between bank accounts, or simply make a phone call you are executing a transaction. In general terms, a transaction involves one or more changes in state. For example, purchasing groceries involves many state changes. You debit your credit card while the grocery store reduces their inventory. Together, these changes represent a transaction. The together part is important since you wouldn't want to pay unless you actually got to take the groceries home. So, we can define a transaction as a group of state changes (or activities) that must be completed as a unit – all activities succeed or fail together.

ACID Transactions

Database management systems are most commonly associated with transactions. A transaction in a relational DBMS allows changes to any number of tables to be treated as a single, atomic action. This is achieved by locking rows until all changes have been made successfully or until an error occurs, which causes all changes to be rolled back, to their previous state. Database transactions are typically referred to as ACID transactions as an acronym for the guarantees made – Atomic, Consistent, Isolated, Durable. You can find a detailed explanation in most database reference material.

Databases aren't the only systems that implement transactions. Enterprise systems, including parts of J2EE such as EJBs and JMS, use

ACID transactions. The Java Transaction API (JTA) and Java Transaction Service (JTS) implement them. Client/server systems also require transactions but add the special wrinkle that the activities and state changes that compose the transaction span multiple machines, networks, and enterprises. Like transactions in a database, distributed transactions are ACID transactions where resources are locked until all activities complete successfully.

Business Processes and Transactions

The Internet has changed the way businesses exchange information. New technologies like XML and Web services provide the data format and communication infrastructure that enable business partners to easily exchange data. Businesses use these new technologies within a broader integration framework, like BEA's WebLogic Integration 8.1, to build sophisticated applications that execute a complex choreography between business partners. These applications are referred to as business processes.

Business processes use transactions to ensure that all activities complete as a unit. But because activities in a business process utilize resources from many business partners and can execute for hours, days, or longer, the transactions must be managed differently. Consider a purchasing business process. It might solicit quotes, reserve inventory, issue purchase orders, confirm receipt of items,

and transfer funds. For scenarios like this, using a single ACID transaction for the entire business process is impractical. Business processes require a *new* kind of transaction.

Long-running transactions avoid locks on non-local resources, use compensation to handle failures, potentially aggregate smaller ACID transactions, and typically use a coordinator to complete or abort the transaction. In contrast to roll-back in ACID transactions, compensation restores the original state, or an equivalent, and is business-specific. The compensating action for making a hotel reservation is canceling that reservation, possibly with a penalty.

A number of protocols have been specified for long-running transactions using Web services within business processes. WS-Transaction with WS-Coordination, OASIS Business Transaction Processing, and WS-CAF are examples. These protocols use a coordinator to mediate the successful completion or use of compensation in a long-running transaction.

The Future

Efforts to standardize business processes like Process Definition for Java (JSR 207) and OASIS' WSBPEL will drive the creation and acceptance of standards for long-running transactions and compensation. Working with long-running transactions, these technologies will expand a business's ability to share and execute business processes with and among its trading partners. ©

BEA

dev2dev 2003

bea.com/dev2devdays2003

The Agile Enterprise

Migration to a service-oriented architecture



Service-oriented architecture is an attractive vehicle for attaining greater business agility. It has the potential to dramatically improve productivity and increase shareholder value with a comparatively modest (though far from dismissible) incremental investment in information technology.

Applications composed of services allow companies to modify business processes more easily and deliver new, composite software solutions faster and cheaper. But there is a more important element: when successfully managed as a broad architectural initiative, services can chip away the complexity of existing software systems.

The simplification of software systems is critical to business agility because it allows the transfer of resources from costly software maintenance to discretionary projects that will make a bigger difference to a company's future.

When combined with sound principles of governance, and undergirded with a commitment to manage the delivery of "service-as-product," the migration to SOA can streamline an entire software portfolio, enterprise-wide. It fundamentally changes the cost structure of IT.

Unlike other cost-reduction approaches that, for example, shift expenses to another quarter, the adoption of SOA can lead to enduring benefits valued in the hundreds of millions of dollars for large corporations or networks of closely aligned trading partners.

But SOA Is Easier Said Than Done

SOA requires a conscious and careful balance between long-term structural agility and more immediate and localized time-to-market pressures. These concerns have long been at odds within most organizations.

In many respects, the challenges of service-oriented architecture resemble challenges already met under the guise of mainframe or ASP models of computing, EAI, ERP, or earlier architectures that attempted to drive

standards throughout IT. Similar challenges surface whenever the needs of multiple constituents must be met, or priorities must be balanced between individual business needs and the common good.

Our mainframe colleagues (and more recently ASP operators) are accustomed to balancing the needs of multiple customers. However, development, production, and life-cycle management tended to be centralized. With SOA, these functions are more often distributed. Every team providing services, regardless of its location or competency level, has an implied obligation to manage its services as "living assets" on which other teams rely. Some do it better than others.

The act of publishing and consuming Web services alone won't make an organization more agile. Without additional effort and new coordinating skills on the part of IT management, SOA will deteriorate into just another incomplete technical strategy, further complicating the software legacy for years to come.

The successful exploitation of a service-oriented architecture requires a level of inter-team collaboration that has very little to do with shared whiteboards and instant messaging. It's all about jointly managing dependencies and objectively trading off one set of interests against another.

Although a service-based architecture results in a looser coupling between compo-



Author Bios

Cathy Lippert is vice president of Product Management for Flashline, the leading provider of software asset management and reuse solutions. She has over 20 years of experience in product management and marketing in the software industry. Cathy focuses on product strategy and direction, introduction of new products, and competitive positioning. One of her key responsibilities is to successfully set product direction to meet market needs.
CLIPPERT@FLASHLINE.COM

Sharon Fay is chief software productivity strategist at Flashline and a member of the Flashline Software Development Productivity Council, specializing in the development of metrics to measure productivity, software quality, and time-to-market. Sharon has more than 15 years of IT experience working with Fortune 500 companies in vertical industries, including financial services and manufacturing.
SFAY@FLASHLINE.COM

"The simplification of software systems is critical to business agility"



The Leading Magazine for Enterprise and IT Management

LinuxWorld Magazine

There is no escaping the penetration of Linux into the corporate world. Traditional models are being turned on their head as the open-for-everyone Linux bandwagon rolls forward.

Linux is an operating system that is traditionally held in the highest esteem by the hardcore or geek developers of the world. With its roots firmly seeded in the open-source model, Linux is very much born from the "if it's broke, then fix it yourself" attitude.

Major corporations including IBM, Oracle, Sun, and Dell have all committed significant resources and money to ensure their strategy for the future involves Linux. Linux has arrived at the boardroom.

Yet until now, no title has existed that explicitly addresses this new hunger for information from the corporate arena. *LinuxWorld Magazine* is aimed squarely at providing this group with the knowledge and background necessary to make decisions to utilize the Linux operating system.

Look for all the strategic information required to better inform the community on how powerful an alternative Linux can be. *LinuxWorld Magazine* does not feature low-level code snippets but focuses instead on the higher logistical level, providing advice on hardware, to software, through to the recruiting of trained personnel required to successfully deploy a Linux-based solution. Each month presents a different focus, allowing a detailed analysis of all the components that make up the greater Linux landscape.

Regular features include:

Advice on Linux Infrastructure
Detailed Software Reviews
Migration Advice
Hardware Advice
CEO Guest Editorials
Recruiting/Certification Advice
Latest News That Matters
Case Studies

SAVE 30% OFF!

REGULAR ANNUAL COVER PRICE \$71.76

YOU PAY ONLY

\$49⁹⁹

12 ISSUES/YR

*OFFER SUBJECT TO CHANGE WITHOUT NOTICE

SUBSCRIBE TODAY!

WWW.SYS-CON.COM

OR CALL

1-888-303-5282

FOR ADVERTISING INFORMATION:

CALL 201.802.3020 OR

VISIT WWW.SYS-CON.COM

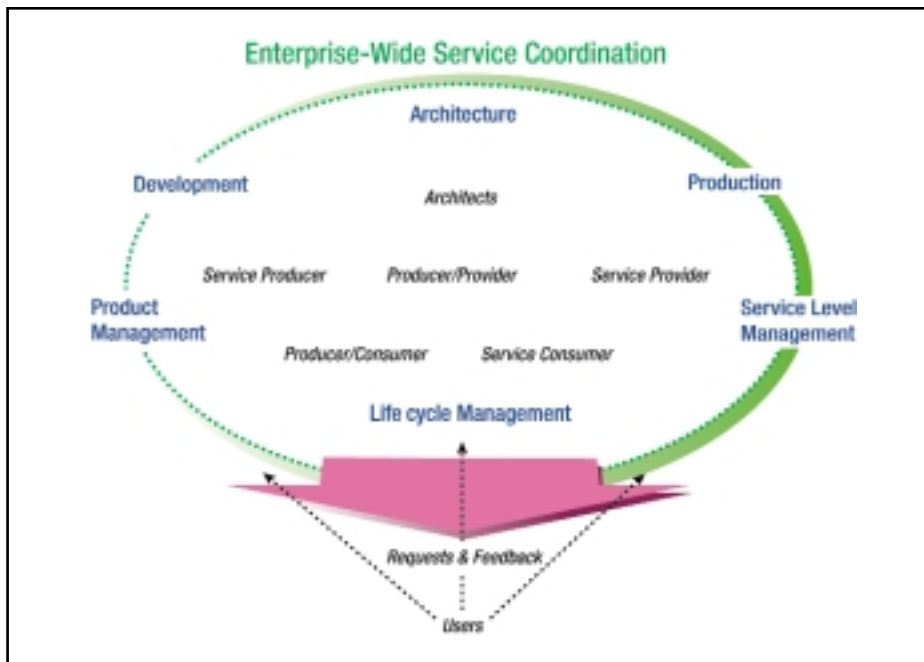


FIGURE 1 A competency center coordinates and supports service producer, provider, and consumer teams, while maintaining the goal to simplify the service-oriented architecture

nents of applications, the architecture also creates new, tighter interdependencies between the organizations that produce, consume, and manage services.

Service consumers must increasingly rely on providers to deliver an acceptable Quality of Service. While Web service management solutions are emerging (just as MSPs did to manage the Internet operations of the '90s), Quality of Service remains difficult to uniformly guarantee. Modern development tools enable virtually any project team or business unit to set up a Web service and register it in UDDI, with no commitment to manage it responsibly for use by other teams, or even a declaration of any intention to maintain it. Management of this exposure is still largely up to central IT.

Compounding that, Web services offer a level of abstraction and interoperability that companies find very attractive for EAI, and the number of systems exposed as services for integration purposes is likely to explode. The danger is that services will be handled as individual touch-points for application integration, outside any unified strategy, thereby increasing complexity.

As a manager in one large telco lamented, "We've had too much enterprise application integration and not enough enterprise integration architecture." Services, if not managed as part of an overall architecture, present similar risks.

Investing in SOA

Implementing SOA is akin to fighting terrorism – it requires a blend of high decentralization in the field; cooperation among loosely coupled agencies; and tight, centralized command.

Executives who are serious about leveraging SOA for business agility must invest accordingly. Not just in Web service technology, but also in organizational solutions that give service providers and consumers the necessary support, allow architecture to exercise its authority, and preserve the autonomy of individual business units to meet profitability goals.

One way to supply this broad coordination is to establish a "service competency center" (SCC). Enterprise architecture is a likely candidate, or perhaps some other high-ranking IT group that

reports to the executive team. Once formed, the SCC facilitates the enterprise-spanning collaboration that SOA demands, and documents the resulting business agility (see Figure 1).

Consolidate to Save

First and foremost, it is the SCC's responsibility to secure a new baseline of enterprise agility by reducing complexity in existing software systems. A well-defined architecture that supports the organization's strategic goals provides the means. The architecture identifies the core set of services required to execute the strategy.

Service reuse is the primary means of consolidation. Each time one team uses a service provided by another (instead of building from scratch), the savings is two-fold: duplicate development costs – valued anywhere between \$10,000 and \$10 million – are avoided, and as much (or more) is saved in future maintenance.

The popularity of the waiver process in some companies suggests that architecture has had trouble making similar Total Cost of Ownership reductions stick in the face of individual business imperatives. With services, the focus begins to shift away from battles over physical architecture and platform vendors to negotiations around service management.

The SCC leverages internal architecture and design reviews to put pressure on project teams to use services, but it must provide assurance of ongoing service availability, support, and responsiveness to change requests or project teams will quickly defect.

Managing Corporate Assets

This brings us to the second responsibility of the SCC: to ensure that the supply and ongoing stewardship of core services deliver maximum value to the enterprise at minimum cost over time.

As an extension of Project Portfolio Management, SCC activities may include:

- Anticipating and monitoring demand for services
- Defining the right mix of services to meet demand

“Services developed internally are corporate assets, and must be managed accordingly”

- Establishing corporate priorities for service delivery
- Provisioning services from internal or external sources

Services developed internally are corporate assets, and must be managed accordingly. These assets are valuable only to the degree that they remain in production, and continue to meet the changing needs of consumers. Interruptions or unanticipated changes to a service can be devastating to its consumers. It is in everyone's best interest that services, together with all of their revisions, are managed responsibly throughout their entire life cycle, from acquisition to retirement.

Business units underwriting the original development of a service are often unprepared for this. Managing a service for other groups incurs new costs, and loyalties are divided. Once a service is delivered, businesses may move on to other projects. Meanwhile, new service consumers with additional requests are left hanging.

Garden-variety IT organizations simply don't have the business knowledge or skill set to manage a service as an asset. The SCC may choose to perform an ownership function, at least temporarily, until service owners can be identified. But this has its drawbacks, too.

The migration to SOA requires policies for service ownership as well as contingency plans for when ownership must shift away from the original service producer. The initial justification for a service should include such plans, or perhaps the service should not be approved for use by other teams. This places the business case for services front and center over the course of their lifetime.

A more commercial appreciation of services as products can smooth the transition

to SOA. But this model also involves a shift in how teams measure success.

Weighing the Priorities

Wherever possible, the SCC should help software teams make a successful transition from software developer to service provider. At the highly distributed, free-enterprise end of the SOA spectrum, service producers continue to manage their own business cases for the services they provide.

Not to be underestimated are practices borrowed from the vendor community, such as software product management and marketing communications techniques for defining and releasing services as products and promoting them to potential consumers. This includes requirements management.

As more services are delivered and the service architecture matures, service consumers inevitably develop conflicting requirements for service enhancements. While the SCC should expect service providers to act as commercial entities, serving the greatest consumer needs on a business-priority basis, the SCC should be prepared to resolve debates. Otherwise, consumers will take enhancements into their own hands and fragment the service architecture.

The SCC is uniquely qualified to negotiate between conflicting interests – especially if it can bring quantitative data to light – and relay the priorities of the executive team.

The SCC can ensure that a comprehensive measurement strategy is in place to:

- Recognize service providers as business entities, and find a basis on which to reward them for contributions to overall business agility

- Track the value of services to consuming parties (including the initial cost avoided, the opportunity value of having the service available, and the ongoing maintenance costs saved)
- Collect data on the underlying operating costs of an SOA
- Assess the net value of each service, in the aggregate, to all consumers
- Objectively model alternatives for service management

The SCC must make visible and quantifiable the basis by which tradeoffs are made between longer-term consolidation goals and business expediency. All too often the justification for a project includes the initial development cost but ignores the future costs of maintaining that software in production. Projecting the reduction of annual maintenance costs achieved by consolidating software systems around a core set of services, and then demonstrating the actual savings once the services are in place, is one way to keep SOA agility goals in focus.

Conclusion

SOA can be viewed as a business enablement strategy in which an organization's core competencies are articulated in software systems and then carefully managed as software assets to maximize the return to the business, both in terms of cost reduction and capitalization on new opportunities.

However, a combination of familiar disciplines (e.g., service-level management and architectural review processes) with relatively new ideas (e.g., portfolio management and asset-centric perspectives on software) must be brought to bear in order for SOA to deliver the promised business agility.

The key concept of treating services as assets ensures that software is recognized, valued, and managed to maximize its business value. Though the management challenges can be daunting, the prospective quantifiable benefits of greater business agility through SOA easily justify the effort. ☺

Facing the Challenges of Web Services BPM

A first leg on the BPM journey



In a previous article (*Web Services Journal*, Vol. 3, issue 7), we looked at business process management (BPM) driven by Web services and the opportunity it presents for new types of business solutions. The potential impact of Web services and BPM is great, but as companies look to harness that power, they must identify and overcome numerous challenges.

We can separate these into technical challenges and service portfolio challenges. The top three technical challenges are:

1. Lack of security controls at the protocol level

A fundamental prerequisite for business process integration is the definition of a "trusted environment." For example, given an activity within a process flow, only a limited number of roles are identified as "allowed to execute." That means that the system should be able to correctly identify the user attempting to perform an activity (authentication) and ascertain that he or she has valid access rights to do so (authorization). This is true whether the user is a person or a computer system (e.g., Web service). We want to guarantee the integrity and confidentiality of any message exchanged, keeping an audit trail of who did what and when. The SOAP, WSDL, and UDDI protocols are inherently insecure and have not addressed these basic requirements. A BPM solution that accepts a SOAP message with instructions for executing a task has no direct knowledge of who initiated the request or the corresponding authorization level. These services must be provided by the enterprise architecture.

Organizations like OASIS and WS-I are defining security extensions to SOAP. These standards, however, need to converge and gain industry-wide adoption before a "trusted environment" can be created outside the firewall.

2. Lack of transaction management capabilities

Current mainstream Web services standards do not provide a mechanism for handling synchronization across multiple enterprise applications. For example, transactions cannot be committed or rolled back as atomic units if they span multiple services. OASIS Business Transaction Protocol and Web Services Coordination+Transaction are examples of standards that are slowly gaining traction. Still, they need to converge and be widely adopted to allow the creation of low-cost, true-enterprise integration solutions.

3. Lack of a universal data definition

Web services rely on XML Schemas for standardizing data formats. Despite some industry-specific efforts, there are no universal standards for canonical representation of data. Companies therefore create their own data formats (for example, DTD/XSD) to exchange data via Web services. This precludes true B2B integration, as the formats from different companies require shared understanding and translation, making it expensive to deploy and maintain.

While custom vendor products exist that repair the lack of security, transaction management, and agreed-upon data semantics, architecture can be thought out to converge toward solidifying standards.

The second set of issues that companies must address often appears once infrastructure technology solutions are solved. These are more long-term issues and can be classified as services portfolio challenges.

The three main services portfolio challenges are:

1. Unstructured Proliferation of Services

Different frameworks, tools, and coding

standards are currently proliferating in business. Applications are wrapped and exposed for exploring the Web services potential rather than for business purposes. Most of these services will not, however, be inserted into BPM and therefore will not be particularly useful for exploiting business value. In general, if Web services proliferate without a management framework, the services they offer will in turn end up being overly complex, low performing, and unmanageable.

2. Lack of Architectural Layering of Services

Web services and BPM favor, but do not guarantee, an appropriate level of abstraction, which is essential in architecting a service-oriented architecture (SOA). Structuring process models and services along separate client, presentation, business, integration, and resource levels of abstraction requires more up-front planning and longer implementation. It is, however, the only assurance that repeatable and lasting solutions are the results of those efforts.

3. Lack of Business Prioritization

Web services solutions tend to be developed in a silo, which is usually in an application or departmental context. The same is sometimes true for BPM applications, which focus on a functional "hub" such as CRM. The preferred approach is to prioritize Web services-enabled BPM solutions as part of the overall enterprise IT portfolio.

Identifying and overcoming challenges is the first leg on the Web services BPM journey. A clear roadmap will be necessary to successfully reach the desired destination. ©



Author Bios

Alejandro Danylyszyn is a senior manager in Deloitte Consulting's. He has worked for over 15 years as a consultant to large high-technology manufacturers, telecommunications carriers, and financial services companies in the areas of strategy, operations/process improvement, and solution design/implementation, with a focus on systems integration, enterprise portals and Web services. Alejandro holds a masters in software engineering from Carnegie Mellon University. ADANYLSZYN@DC.COM

Cesare Rotundo is a senior manager in Deloitte Consulting. Cesare Rotundo is a senior manager in Deloitte Consulting. His expertise is in applying IT for business results and in managing large implementation projects around Real Time business integration and customer integration solutions. His IT focus is around the enterprise software infrastructure, particularly EAI, B2Bi, enterprise portals, BPM, Web services, and J2EE. Cesare holds an MBA from INSEAD. CROTUNDO@DC.COM

SUBSCRIBE TODAY TO MULTIPLE MAGAZINES

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

RECEIVE
YOUR DIGITAL
EDITION
ACCESS CODE
INSTANTLY
WITH YOUR PAID
SUBSCRIPTIONS

3-Pack

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

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

6-Pack

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

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

9-Pack

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

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



TO
ORDER

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

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

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

☐ Linux World Magazine

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

☐ Java Developer's Journal

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

☐ Web Services Journal

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

☐ .NET Developer's Journal

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

☐ XML-Journal

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

☐ WebLogic Developer's Journal

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

☐ ColdFusion Developer's Journal

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

☐ Wireless Business & Technology

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

☐ WebSphere Developer's Journal

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

☐ PowerBuilder Developer's Journal

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

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

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

SYS-CON
MEDIA

Enabling the High Performance Corporation

Is the PSB the next step?



To quote Peter Drucker: "The most important, and indeed the truly unique, contribution of management in the 20th century was the 50-fold increase in productivity of the manual worker in manufacturing. The most important contribution management needs to make in the 21st century is similarly to increase the productivity of knowledge work and the knowledge worker."

Unfortunately, to date, our efforts to improve the productivity of this increasingly important segment of the organization have been less than spectacular. As is pointed out in a recent study by the Center for High Performance and its parent company Hudson Highland Group Inc. ("Unlock Corporate Performance: America's Knowledge Workers Provide The Key") a "performance crisis" has hit Corporate America, hindering its ability to shake off the effects of the sluggish economy and return to sustainable growth.

For organizations to achieve peak performance, they will need to dramatically change the way in which their information systems are built. Fortunately, we are rapidly approaching a breakthrough

in the way applications are developed and managed that will facilitate this change. A number of individual factors, including the proliferation of Web services, are working together to create this breakthrough.

The Tipping Point

In his book *The Tipping Point*, Malcolm Gladwell searches for catalysts that precipitate a "tipping point" – that moment in time when the boiling point is reached. This concept holds that small changes will have little or no effect on a system until a critical mass is reached. Then one final small change "tips" the system and a large effect is observed.

One example of this phenomenon is the establishment of e-mail as a primary means of doing business. Although personal computers in business had been around in various forms since the early 1980s, they were not seen as primary means of communication. In order for that to change, several factors had to occur: PC prices had to come down low enough to place them on every desktop; users had to become more comfortable with their use to incorporate them into their daily routines; network technology had to improve to the point where internal connections extended throughout the organization; a public network (the Internet) had to be established to allow external point-to-point communication;

and easy-to-use e-mail software had to be created. The tipping point came when e-mail software was on enough desktops to drive further adoption. Suddenly, if you didn't have e-mail you were out of touch, and unable to conduct business the way the rest of the world was conducting it.

Today, technology has become so ingrained in the daily life of business users that their requests for automated solutions far outstrip IT's ability to deliver them all. One of the consequences of this is that many users have resorted to creating their own solutions using desktop tools (for example, creating macros in spreadsheets to perform repetitive calculations). The problem with this is that these systems are isolated from the rest of the organization, and the tools used to build them are often ill-suited to the task at hand. But we are moving toward a tipping point, and the change will be swift and sudden. Key drivers toward this point are Web services and service-oriented architectures (SOAs). These new concepts will enable solutions developed by and for a single business user to be easily extended to others in the organization. All that is needed to reach the tipping point is the right tool built on these concepts – a personal service builder (PSB).

The impact of PSBs will be profound. Before spreadsheets came along, for example, financial analysts had to get IT to develop a system for them in order to automate their work. Today, the thought of outsourcing a spreadsheet to the IT department seems absurd. The same will be true of many new kinds of systems that will be developed by users using PSBs.

From IT Department to IT-Savvy Organization

One of the keys to the tipping point being reached is the willingness and



Author Bio

Jonathan Sapir is president of InfoPower Systems, Inc., developers of SnapXT, an event-driven, service-oriented, rapid application development and deployment platform. JASAPIR@INFPWR.COM.

growing ability of business users to computerize their own part of the business. This is rapidly leading to what can be called the "IT-savvy" organization. In this new organization, responsibility for IT will go from being solely the domain of the IT department to being a responsibility shared by (nearly) everyone within the organization.

This idea is a radical departure from the current norm, where practically all development projects are controlled by IT. Just the thought of giving users the ability to develop applications, no matter how small, outside the umbrella of IT is enough to give CIOs nightmares. Indeed, because (unlike spreadsheets, for example) the applications developed by users will become part of the organization's portfolio (they won't remain isolated on desktops), there needs to be an overall platform, controlled by IT, that assures compliance with standards and security. In addition, IT has a critical role to play in providing PSBs secure and easy access to the information contained in legacy applications.

Why the IT-Savvy Organization is Necessary

Creating an IT-savvy organization addresses one of the great conundrums of the current IT landscape: the mandate to do more with fewer resources. Since organizations began to prioritize cost containment over innovation, IT budgets have been shrinking. Yet there has been a marked increase in demand for new applications, both large and small. Getting business users involved in automating their own job functions frees IT resources for the more complex, detailed work that affects larger numbers of users. Essentially, this concept extends the IT department to include the whole company. Users have the need and the desire, and given the right tools they have the ability. Let's look at each of these factors in more detail.

The Need

Nicholas Carr's famous (or infamous, depending on your point of view) article in the *Harvard Business Review*, "IT Doesn't Matter," claimed that IT is no

"Business is not a static entity, but rather a series of constantly occurring and evolving events"

longer viewed as a business advantage, but instead has become more of a commodity in the way electricity has. While many photons have been burned arguing both sides of the case, the one core truth behind it is that having technology alone is not enough. It's not a question of whether your company's business functions are computerized and your competitors' are not.

The big difference now is the time frame for development. When technology was seen as a strategic advantage, organizations were willing to invest huge amounts of resources and wait months to roll out a major initiative. They felt the long-term payoff was worth the short-term wait. Today, with IT being viewed as a means to cost savings rather than competitive advantage, those time frames have been compressed. Business users want the advantages technology can offer now, not six months from now.

Complicating this scenario is the fact that in recent years 75% of new application projects have come in over-budget or later than projected, with many applications scrapped before they are deployed. There are many reasons for it, including users changing their requirements in mid-stream and general shifts in the business landscape such as the increase in globalization.

As the pace continues to increase, the old methods of developing applications to address the whole of the business from end to end become increasingly difficult to justify. There is an undeniable need to change the way technology is developed and rolled out in order to keep pace with the way business now operates.

The Desire and the Ability

When computers were first introduced

into the workplace, they were viewed as mysterious and intimidating by many business users. Those users learned the functionality by rote, and never strayed from what they'd been taught. It's hard to believe now, but I can remember when programmers had fun with user naivete by telling them not to drop the disk drives because the data would fall off! Users could be kept in line by telling them they could wipe out the entire organization's records with a single incorrect command (and in some cases they probably could). The users back then were therefore certainly reluctant to experiment or discover new capabilities on their own.

The current generation of business users has no such qualms. The older group has been using computers as part of their jobs for the past 20 years, and now give them no more thought than they do the telephone or copier. Younger workers are even more comfortable with them, having never known a time without computers, and having played with digital toys since the age of three. Many learned basic programming skills in high school or even middle school, and manipulated functions in applications even before that.

In their jobs, both groups use computers to create database queries, set up spreadsheets to perform complex calculations across multiple worksheets, and do multiple other tasks on a daily basis without any experience with Java, Visual Basic, or other programming languages. At home, they create Web sites for their personal interests, despite the fact that they know nothing about HTML.

They do all of these things by choice – and because they now have the ability. Consider the database query. A few years ago, setting up a database query would have required submitting a request to IT

“Business users want the advantages technology can offer now, not six months from now”



and waiting a week to two weeks for an IT staffer to code it. Now they use intuitive tools to create the query themselves and are able to obtain the results immediately. The technology has advanced to the point where they're able to create these applications for themselves, without IT intervention.

Moving the Concept Forward

The next logical step is to create an environment where the services created by users can become part of a larger enterprise "application." That's what PSBs are designed to do. They give business users the ability to create services that solve their immediate needs, yet may be combined with services from other users to address larger issues. Think of those Russian nesting dolls. The smallest is an individual service. The largest is the enterprise's IT system. Services created with PSBs are able to cascade in the same way to create both of those layers, and all

the layers in between.

So what exactly is a PSB? One way to think of it is as a development environment that lets business users create simple applications with techniques such as mind mapping instead of writing code – or depending on IT to write the code. Those business users know their jobs very well. They know what they need to do it better. But unless they've taken programming courses, they've lacked the knowledge of how to translate their ideas into something practical. Now, business users simply need to understand the logic – A follows B follows C – and then map it out with the PSB. The rest happens behind the scenes.

As opposed to today's monolithic and inflexible applications, the new PSB-based model is designed to match the way business operates. Business is not a static entity, but rather a series of con-

stantly occurring and evolving events. Today's events may continue, or they may be replaced by other events. With the PSB-based model, users are able to react immediately to changes in their business environment and develop new services as the need arises. Placing these services into an SOA controlled by IT allows those new services to be shared easily with others. Once the standards are in place, the organization can become more agile, thus gaining a true business advantage over slower-reacting competitors. At this point, IT has again become an advantage – not for its own sake, but for what it enables business users to do.

Unlocking the Potential

PSBs are one key to unlocking the potential of business users to create a more effective, more efficient corporation that the Hudson Highland Group details in their report, because PSBs do for business development what programs such as Microsoft Front Page did for Web site building. They provide the tools that allow business users to create many of the services they need without coding or worrying about technical infrastructure. Business users are thus able to apply their knowledge to solve many of the challenges facing the organization – including the pervasive need to reduce costs – while relieving IT of the burden of supporting numerous smaller user requests. As a result, IT is free to focus on bigger picture issues that affect the entire organization.

We are at the tipping point for the next revolution in IT; SOA-based PSBs will push us over the edge. As they grow in use, they will dramatically change our perceptions of IT. Cycle times will be reduced or even eliminated in some cases, replaced by an ongoing interest in development. Entire systems will change over, but will do so on an ad hoc basis, much as the river remains constant but the water within it changes. In the end, PSBs will help us realize the vision of the high performance corporation, and create a new era in IT. ©

THE INSIDER INTELLIGENCE YOU NEED...

TO KEEP AHEAD OF THE CURVE
FREE E-Newsletters
SIGN UP TODAY!

Go to www.SYS-CON.com

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

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



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

at www.sys-con.com

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



A Look at WS-I



A conversation with Tom Glover, chairman of the Web Services Interoperability Organization; and Andy Astor, chairman of the WS-I Marketing Committee



TOM GLOVER



ANDY ASTOR

Rhody: *Let's start with a high-level overview of WS-I and, if you can, highlight key members, new members, and what your broad mission statement is.*

Glover: As far as key members are concerned, I'd have to say every member is a key member.

The mission statement is really easy. As emerging standards, particularly Web services standards, are adopted by the marketplace, and as it is discovered that there are interoperability issues related to the use of those standards, WS-I will produce profiles and supporting materials that address those interoperability issues and resolve them. That is, in a nutshell, what we do.

We started in February 2002 and have been under way for about a year and a half. We started by focusing on what was commonly perceived to be the bedrock for Web services – SOAP for transport, WSDL for definition of Web services, and UDDI for discovery. Those three documents – we can call them specifications or standards; I leave the distinction up to braver hearts than mine – have been available for some time from a variety of sources. Of course they are all in either W3C or OASIS. They have been adopted by a number of platform, tool, and application vendors, and there were well-understood interoperability issues related to using them, singly or together, to produce Web services.

WS-I's first job was to look at those three standards, identify the interoperability issues, and come up with recommendations on how to resolve them. That is, in fact, what we are preparing to do. We have the Basic Profile version 1.0 in working group approval draft status now [Note: the Basic Profile has since been released], and that document is the WS-I position on how to address the interoperability issues related to the use of those standards in Web services. That's what we've focused on for the past year. It's the first



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 consulting services company. SEAN@SYS-CON.COM

profile we've produced, so we've been learning as we go. In addition to producing the profile, we've produced a number of pieces of supporting material.

First, we've been producing testing tools that help people analyze services and artifacts related to them, such as UDDI descriptions and WSDL documents, to verify that those services and supporting documents conform with the Basic Profile. Those tools will be rolled out, we hope, this year. They are going into their approval cycles now.

In addition, we've produced a set of sample applications that developers can look at to help them understand what a conformant set of Web services looks like. In fact, we have a showcase of close to 12 companies that have implemented part or all of the sample application and put it out on the Internet on servers, together with an interface that lets you exercise those Web services and prove to yourself that when someone has, in fact, implemented interoperable Web services they can be mixed and matched. You can, for instance, one day start using the Microsoft implementation service, and the next day shift to the BEA or the Sun or the IBM implementation of that service, and because their interfaces were developed using a common standard and all adhere to the Basic Profile, it doesn't matter which one you use; they will all work in the application. So we've got the profile, we've got the tools, we've got the sample applications, and there are a few more documents that wrap around them that help describe them a bit more, but that's the root of what WS-I does.

Looking ahead a bit, what's next? There is, I believe, consensus in the trade press and among the Web services community that security is the next thing we need to solve. We now have the Basic Security Working Group under way, and that group is developing a profile that addresses security. It hopes to have a preliminary draft available this summer but beyond that, of course, these things take as long as they take. We hope to have a draft profile before the end of the year; that's the goal of the working group. Once that's available, we'll be better able to estimate when the profile will be final. That's the next big deal for WS-I, dealing with security.

There's also an effort under way now to take the Basic Profile and figure out how we can add attachment support to it because a number of our members have indicated that that is something they want. It's not as close to completion as Basic Profile 1.0, but the working group has a draft that they're kicking around internally. They're not satisfied that they've solved all of the technical issues yet, which is why it has not yet entered the public eye.

Anything past security becomes more speculative. There are a couple of things that we look at before we start work. We look to see that a standard had been accepted by the industry – that's a very nebulous term, but in general it means that a standard has been implemented by the vendors; it's being used; it's demonstrably something that the industry thinks is important.

The second hurdle we have to go over is that there need to be interoperability issues related to the use of the standard. SOAP was widely implemented in the middleware that most of us rely on. It was being used by people who develop Web services. There were obviously interoperability problems with it, so that's an example of a model. We expect to go through the same cycle when we look at what the next set of issues we deal with is. We can conjecture, and we did in fact in the roadmap we used when we launched WS-I. We pointed to things like reliable messaging, workflow, choreography, business rules, as things that some of us thought would probably come up as the next sets of issues for the Web services community to deal with. And we pretty much still think those are things we'll need to address down the road. However, we're not there yet. Until there are accepted standards that address those topical areas, and until it's clear that interoperability issues exist, those simply aren't topics that WS-I is going to take up and start developing profiles to deal with.

Rhody: *Obviously SOAP, UDDI, and WSDL have seen a lot of attention. Will you continue on those, like the attachments, or are more coming up?*

Glover: Where the Basic Profile is concerned, we're going to get 1.0 out the door and then we're going to deal with attachments. I'm fairly confident that that is something we'll focus on this year. Again, speculating for a minute, you can look at what's going on with the basic definition of Web services. Inside W3C right now there is work going on on SOAP 1.2, there's work going on on WSDL 1.2. Within OASIS, there's work going on on UDDI 3.0, and all those are follow-on standards to the standards that are in the Basic Profile. It's reasonable to suggest that as those standards are rolled out by various owning organizations, there will be industry uptake. Once we see that SOAP 1.2 is supported by platform and tools vendors, and incorporated into Web services, it's reasonable to expect, again, that despite the wonderful job that standards developers have done, there will be interoperability issues. If that's the case, it's reasonable to think that that may be a set of work that WS-I takes on somewhere down the road. But again, we're going to wait. SOAP 1.2, WSDL 1.2, aren't out in "rec" form from W3C yet, and OASIS hasn't released UDDI 3 as a recognized standard yet. We'll wait for that first. We'll also wait to see the various vendors incorporate those standards into their products, and we'll wait to see WS-I members come back and say, "I've tried to build Web services incorporating SOAP 1.2, or WSDL 1.2, or UDDI 3.0, or all of the above, and here are the inter-

operability problems I need help with." Once we get a growing set of interoperability problems identified by our members, that's what will prompt us to say it's time to write a profile.

Rhody: *That makes sense. Do you see pretty much all of your work as front loaded by other standards bodies, or is anything coming directly from WS-I?*

Glover: No, WS-I's job primarily is to do follow-up work. There is some work we do that is not initiated solely by the standards groups. We've discussed what prompts us to make a profile. Stepping away from that for a moment, we're spending an increasing amount of time with the WS-I membership to identify the Web services usage patterns they're seeing in their applications. And of course we're trying to cultivate the membership so we get a representative cross-section of the industry. We'll sit down with our membership and say, "How are you using Web services? What are the usage patterns you see emerging, the interoperability problems that you're seeing?" That is totally WS-I driven. We're trying to get a better sense, within the organization, of how the technology is being employed, what the problems are, so we can be proactive and go out from the WS-I to say, "Here are the issues we see that need to be resolved." Not simply as a response to analyzing existing standards, but at some point down the road, "Here are the usage patterns that users might like to implement; here are the reasons they can't."

I think the two concepts that describe the work the WS-I is doing are 1) implementers forum, and (2) standards integrator. One of the things we've been doing is developing effective relationships with W3C, OASIS, and IETF. Our primary focus has been on the organizations that own the standards that represent our profiles, and I think that will continue. But we're also working with industry vertical groups and other groups that are trying to be consumers of our work, or are trying to contribute their requirements and shape what we do. Focusing for a minute on feedback, however, obviously the first and clearest feedback mechanism is as we release draft and final versions of our profile. We're encouraging the organizations that own the standards we're referencing to review the profile, look at the interoperability issues we found, and insofar as it's possible, incorporate our recommendations in whatever form they feel is appropriate into follow-on versions of their standard.

This gets a little convoluted because we're working the other way as well. If you look at the clauses and basic profile that deal with SOAP, you'll find that many of our clauses are closely aligned with the work that went into SOAP 1.2 within W3C. We attempted, as we developed the profile, to minimize the costs associated with moving from a Web service that was built based on SOAP 1.1 and the Basic Profile into SOAP 1.2. The differences between the two technical definitions are as small as we can make them, but at the same time there are some differences. We're sending the profile back to W3C and encouraging them to read it, and if they see comments in it they think are good ones, we're encouraging them to weave our recommendations into future versions of the SOAP stack, or the WSDL stack, or the UDDI stack, or down the road into WS-Security. There's going to be some stuff in the profile that they simply can't find a home for because in addition to commenting directly on the standards, we also comment on how two or more standards should be used together. Because we're taking a view that's different from a

single standards developer view, some of this will just be hard to incorporate directly into a standard. It's probably down the road that you'll see the majority of the content of our profile. It will be, "How do you deal with boundary conditions between SOAP and WSDL; between SOAP, WSDL, and UDDI?"

Rhody: *I think I understand the deliverables process. What about the organization itself? Can you speak about the recent board of directors election, and any formal relationships you have with other standards bodies?*

Astor: The WS-I was created about a year and a half ago and there was a lot of inertia that I think had to be gotten over by the people who created the organization. Once momentum was established, it was important to start looking around and ask, "Okay, now that we have something here, is it the right something and what might need to be changed? What do our members think; what does the public think?" There have been several changes, and my presence on this call is certainly one of them. Back in late 2002, due to interest from the rest of the community in participating on the board of directors, which was previously open only to founding members, an election was held and two additional board members were elected, Sun Microsystems and webMethods. It was in that process that I got onto the board, and I now chair the WS-I marketing committee.

In terms of other organizations, it became clear some time ago that, aside from the original nine founding and contributing members, there was another category of people that are important to our organization as well. Those are other standards organizations, not just the W3C and OASIS, but also many of the vertical standards groups like RosettaNet, Cydex, UCCNet, and others, such as ACORD in insurance. Assuming that the membership agrees, there will be a new category of associate membership established that will be for organizational-type members to join the group as well.

Rhody: *That would be like system integrators, consultancies, etc.?*

Astor: No, more like RosettaNet, ACORD, and groups like that. We're talking primarily about non-profit organizations that don't have much money.

Glover: We have an established mechanism for bringing companies in. It doesn't matter if they are a services company like Accenture or a middleware company like BEA. This new provision will allow industry verticals or standards organizations such as W3C and OASIS to participate without paying dues. Primarily, these organizations are not accustomed to paying dues and are not set up for that sort of financial exchange. Many of them run on very small budgets. It's also a mechanism that we could use to bring academics in if there were specific cases where we felt that was necessary.

Rhody: *That makes a lot of sense.*

Astor: Another change is that the board has created several subgroups – we call them board committees – to deal with various issues. For example, the marketing committee, or the liaison committee. There was a lot of interest from the membership in participating in those committees so we've created a policy where whenever possible we open up those committees. The marketing committee is the first one that is open not just to board members but to contributing members as well. We are now in the process of bringing new members into the marketing committee.

The theme that I noticed about WS-I that I think is evident is that it has always been an inclusive organization. Certainly, I have found it so since I joined and got involved. But its policies are being broad-

ened to be even more inviting to nonboard companies to participate in new and interesting ways.

Another interesting thing that happened recently that got an enormous amount of energy is that a special interest group for Japan was created. There are 15 participating member companies now in Japan who are actively translating all of the WS-I deliverables and evangelizing and

promoting WS-I there. And many of the things that Tom alluded to, such as SOAP with Attachments and security, are all driven by the membership. So the changes going on at WS-I are extraordinarily member driven. That is why we can't tell now what we will be working on in six months. As Tom said, if SOAP 1.2 makes it in the marketplace, terrific, and in all likelihood it will be a candidate for a profile, but it's really driven by the members and their concerns.

Rhody: *That all makes sense to me, but this is the first chance we've had to talk with anyone from the WS-I in depth. I'd like to get an idea of the "state of the union." Where do you think the next big things will be in Web services, not necessarily things WS-I will focus on immediately, but what is driving the industry.*

Glover: First, I think you want to talk to the various vendors because you're asking about a space where I believe there will be a difference of opinion. And the difference won't be so much over the technical areas where we have work to do. It will be over timing, over prioritizing, and of course at this point there will be a difference of opinion over which standard – or which effort that might produce standards – will prevail in certain areas. If you look at reliable messaging, there's more than one effort underway right now to produce an RM spec. If you look at workflow, there's no consensus on a standard for workflow. If you look at business rules, same thing. These areas are still, to some extent, hotbeds of research and discovery and discussion, which is really good news. Some people look at the fact that there is more than one effort to produce a reliable messaging spec and shake their heads and say, "This is a bad thing; why can't we settle on one?"

I guess I look at it and say, "Obviously this work is important; otherwise there wouldn't be so many people engaged." A little bit of debate early in the cycle is a good thing because it lets us look at var-

"the changes going on at WS-I are extraordinarily member driven. That is why we can't tell now what we will be working on in six months" —Andy Astor

Comdex

www.comdex.com

ious ways of solving reliable messaging, or business rules or security for that matter. It lets us explore different pathways for a while. The reality is that at some point we have to make up our minds and I think that's happening. Five years ago we didn't have SOAP and we do now. Five years ago we didn't have WSDL and we do now. We didn't have UDDI and we do now. WS-Security I'd say, and this is a personal opinion, is emerging as one of the key specs for Web services security and that's happening because the industry is coming to consensus on it. I really believe that in a couple years you're going to see that Web services is the next area where the industry says we have to get this sorted out. Let's come to an agreement on which specification or specifications we're going to base our reliable messaging implementations on. And frankly, this is where WS-I comes into play.

I think people join WS-I for pretty much one reason – they have decided that we need interoperable Web services. It's important to all of us for different reasons, depending on who you're looking at. Some of us are producing middleware; some of us are producing tools; some of us are consuming technology. We have heterogeneous IT environments; we're tired of not being able to integrate applications without a custom solution. We see Web services as a technology that will help us do application integration more quickly and for less money.

These are all good reasons for why people have joined WS-I. We all believe we have to settle on specifications that are deployed industry-wide in a common way. And that's really what WS-I is all about – it's a growing community of companies saying this is important to us; we want Web services to become a viable technology. The only way it will become viable is if we have a common understanding of how to interpret these specifications. That's what we're here to do. That's the really good news, and I think that over the next couple of years you're going to see WS-I members starting to say, "Okay, this is the next nut we have to crack. Now it's security. Let's all sit down and talk about our security problem."

That conversation is ongoing. We have Basic Security going on right now. If you look at what the Basic Security Profile is based on, you'll know a number of things about it. Security tokens are important for the security community. We've listed some token types that we are going to address. There's already a healthy discussion under way on whether that list of token types is sufficient. Maybe it is, maybe it isn't. It may be very possible that even as we defined the Basic Security working group's charter we did so over the types of tokens that should be included. SAML was one that didn't make it into the version 1 charter. It's very possible that we're going to incorporate SAML into a subsequent iteration of that profile and that's just

part of the conversation that's going on. When SAML settles down, and we understand whether or not it's important, we'll make a decision on whether we have to do something with it.

One of the challenges we have is in making sure we have this conversation going. It's obvious through the conversation we've had so far that it's really important, so WS-I has to pull into the organization the companies that represent what's going on in terms of the Web services user community. Particularly, we have to pull in the companies that are technology consumers using Web services, not just the companies that are using platforms or writing tools, and this is a real challenge. If you look at W3C, at OASIS, typically the companies that participate in standards

development are the hard-core technology companies that produce platforms, that produce middleware, that produce the component solutions that people use. Not the companies that rely on those solutions. In fact, when you look at groups like the Liberty Alliance, one of the things they have said is that unlike many standards efforts, they have a lot of technology consumers in their ranks. The WS-I needs that, and that's a challenge that we've got – trying to grow

"If Web services is going to become a technology that helps us all do what we want to do, namely application integration, it's going to take a lot of us participating in this effort" —Tom Glover

within our membership an ever-larger percentage of technology consumers that can come in – like United Airlines and a number of others – and say "Right, we don't build the middleware; we don't write the tools; we use this stuff. Let us tell you the problems we're having." We're succeeding, thanks to the work that this group is doing. Of course, the work that the liaison team does is key here too because these companies often participate in organizations such as ACORD, such as OMA; organizations that cater to the problems particular to their industry. We've spent a lot of time talking to the finance community, to health care, to any number of other organizations, saying "Tell us what your particular needs are – what it is you're trying to do that you think is unique to you. Looking down through your industry at your basic infrastructure requirements, what do you need?"

Rhody: *I have one last question. If you had to say one thing to our readers about what's going on, the relevance of the work you're doing, what would you tell them?*

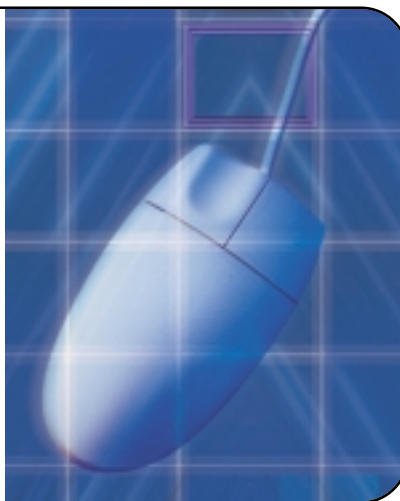
Glover: Participate. If Web services is going to become a technology that helps us all do what we want to do, namely application integration, it's going to take a lot of us participating in this effort. Whereas there is probably a limited section of the industry that is interested in writing the next SOAP spec, there is a much greater segment of the industry that has something valuable to say about what they need from Web services if the technology is going to go partway toward meeting their needs. And we need to hear from them. ☺

Wow Gao

www.wowgao.com

Building Your Own SOAP Client and Reporting Tool

Moving from evaluation to adoption



If your company is like most, it is likely that your suppliers, vendors, distributors, and partners have exposed dozens of Web services for your use.

What do you do if you want to try several Web services before deciding whether to integrate them with your IT environment, and you don't have the time to write a client for each? And what is the best way to handle verbose results that make it difficult to extract the information that is meaningful to your company?

Although some companies let you download free tools to invoke Web services, they generally don't give you much understanding of how a request is generated and usually won't go beyond showing you a raw response. You may want to roll up your sleeves and explore the many standards and APIs such as SAAJ and XSLT that are available to see how you can

invoke Web services and generate useful reports. During this process you can begin building a framework for integrating Web services into your environment.

In this article we'll describe how to develop a generic tool that allows you to select the WSDL file of any Web service, provide the required input when prompted, run the Web service to view the result, and construct a report containing only the data of interest to you. Our generic tool, which we will call WSClnt for the purposes of this discussion, uses industry standards and readily available Java APIs such as WSDL4J, SAAJ, Castor, JDOM, and XSLT.

WSClnt Overview

Our WSClnt will allow us to select a WSDL file, regardless of whether it resides on our local machine, a private intranet, or the Internet. After loading and analyzing the file, WSClnt will display the available operations – such as

GetFundQuote, GetHistoricalQuote, and so on. For each defined operation, WSClnt will display the structure of the corresponding input and output messages.

When we select an operation, our tool will create a dummy XML message that conforms to the operation schema. We can then fill in the input values and click the Execute button to run the Web service. WSClnt will

then send a SOAP message to invoke the Web service and display the response, which will be a valid result or a SOAP fault describing a failure.

Let's look at how we'll load the WSDL file, create sample XML input, invoke the Web service through SOAP, and generate a report from the response.

WSDL Parsing and Analysis

WSDL (Web Services Description Language) defines an XML grammar that is used to describe Web services. WSDL separates the abstract definitions of messages and endpoints from their concrete data formats and deployments.

WSDL describes Web services by providing the details of the communication requirements necessary for a client to invoke the Web service. Specifically, WSDL is an XML model that describes the messages that need to be exchanged between a client and a service provider. Additionally, WSDL describes how and where a Web service is invoked.

A WSDL document uses the elements shown in Table 1 for defining a Web service.

Each element will be used to supply the information we need to invoke a Web service. WSClnt will use IBM's Web Services Description Language for Java Toolkit (WSDL4J) to programmatically analyze the structure of the WSDL and identify the operations available for consumption. WSDL4J is the open source reference implementation of the Java APIs for WSDL (JWSDL) being developed under the Java Community Process.

Our demonstration Web service is named "XigniteQuotes" and is available to the public from Xignite, Inc. The XigniteQuotes Web service returns a full month of historical quotes for any U.S. Equity. (You can view the WSDL for this Web service at

www.xignite.com/xquotes.asmx?WSDL.) WSClnt will be able to read this service description to get the details required to consume this Web service.



Author Bio
Bikash Behera and Jim Winfield are senior software architects at Metaserver, Inc., a provider of business process integration software.

TABLE 1: Defining a Web Service

WSDL Element	What the Element Describes
Types	Describes the data types used by a Web service using a type system such as XML schema.
Message	The abstract format of a particular message that a Web service sends or receives
Operation	The abstract description of an action supported by a Web service.
Port Type	A named abstract collection of operations.
Binding	Concretely defines the protocol and data format specification for a specific port type.
Port	A single endpoint defined by a combination of a binding and a network location
Service	A collection of ports that a Web service provides

Finding the Services and Operations

Our tool defines a class named `ComponentBuilder` that uses `WSDL4J`, the `Castor` Schema Object Model, and `JDOM` to analyze and compose an in-memory model of the Web service. The `ComponentBuilder` class creates an instance of `WSDL4J`'s `WSDLReader` interface to load the WSDL definition. The `WSDLReader` interface defines the necessary methods for us to turn any WSDL into an in-memory model of a service description. We pass to the `readWSDL()` method the URI of our WSDL document and receive an instance of the `Definition` interface.

The `Definition` interface defines the methods we need to begin analyzing the WSDL definition and makes it possible to programmatically discover the defined services, their operations, the data types, and the service's endpoint URI.

Now that we have an in-memory definition of the WSDL document, we ask for the services that are defined. The `Definition` interface's `getServices()` method returns a collection of `Service` instances that our tool will iterate. Each `Service` instance will contain a group of related ports, and each port is represented as an instance of the `Port` interface. From the `Port` instance we can discover the binding referred to by the defined port. A binding is represented as an instance of the `Binding` interface, which gives us information on the binding operations that are represented as instances of the `BindingOperation` interface (see Listing 1; due to space limitations, the code for the article can be found online at www.sys-con.com/webseries/sourcec.cfm).

Finding the Parts

Using the `BindingOperation` we can find details about the concrete implementation of an abstract operation defined in WSDL. Our tool currently supports only SOAP operations. `WSDL4J`'s support for extensions to WSDL include the `ExtensibilityElement` interface. `WSDL4J` defines the `SOAPOperation` as a type of `ExtensibilityElement` to give us the information we need about the concrete implementation. From the `SOAPOperation` instance we can determine the encoding style and the SOAP action URI if one is defined.

The defined operation is represented as an instance of the `Operation` interface that is obtained by calling the `BindingOperation`'s `getOperation()` method. The message structures of the operation's input and output are accessible as `Message` objects. Message definitions are obtained by calling the `getMessage()` method defined on the `Input` and `Output` interfaces.

The `Message` interface defines a method named `getParts()` to retrieve the SOAP parts that have been defined for this message. A message part is represented as an instance of the `Part` interface. The `Part` interface defines methods for retrieving a part's name, element name, and data type name (see Listing 2).

We can use this information to get the associated schema for each message part. This allows us to construct the request message to be sent to the Web service and to process the response generated by the invocation.

Creating Sample XML Input

Now that we know our message parts, we can create sample XML to use as request data when invoking the Web service. The `Definition` interface defines a method called `getTypes()` that we call to get the instance of the `Types` interface. The `WSDL4J` `Types` interface represents the `<types>` element defined in a WSDL document. We read the schema defined by the `types` element into an in-memory model using the `Castor` API.

In our sample WSDL document our `<types>` element contains the schema shown in Listing 3.

For each complex message part, we will refer to this schema to build an XML message that will be passed as part of the SOAP message sent during the Web service invocation. If we expect that the response part will be of a complex type we can refer to the type defined in the schema to process the response message.

The `WClient`'s `ComponentBuilder` class defines a method named `buildMessageText()` that takes as its input a `WSDL4J` `Message` instance. From the `Message` object we can obtain the list of parts that define the message. The parts are iterated and sample input text is built for the message parts. For each part processed we check to see if there is a complex type defined for it in our `Castor` Schema Object Model.

For each message part, we generate an initial XML instance that can be used to invoke the Web service. For example, our demonstration Web service takes as the request message the element named "GetQuotesHistorical". Using the element defined in the schema above an appropriate input message would be:

```
<GetQuotesHistorical
xmlns="http://www.xignite.com/services/">
  <Symbol>SUNW</Symbol>
  <Month>12</Month>
  <Year>2002</Year>
</GetQuotesHistorical>
```

This generated message is then saved to our `OperationInfo` instance and is then used as the initial message to be sent when we invoke the service (see Listing 4).

SOAP Invocation

`WClient` uses the Java SAAJ API to invoke the Web service. SAAJ provides a flexible, yet fairly straightforward, way to consume a Web service. During our WSDL analysis we captured the elements necessary to make a Web service invocation using SAAJ.

The sequence of consuming a Web service from our SAAJ client will be:

1. Create a connection
2. Create the message
3. Add message content
4. Send the message to the destination
5. Process the response

Creating a Connection and Message

We create the connection by calling the `SOAPConnectionFactory`'s `newInstance()` method. We then create a new `SOAPMessage` instance using SAAJ's `MessageFactory` class. The `SOAPMessage` instance created by the SAAJ factory comes initialized with a pre-defined `SOAPPart` that contains a `SOAPEnvelope` (see Listing 5).

In addition, the `SOAPEnvelope` comes prebuilt with an empty `SOAPHeader` and `SOAPBody`. The `SOAPHeader` is optional. Our example doesn't pass a `SOAPHeader`, so we will remove it from the envelope when building the request.



The Historical Quotes				
Symbol = MSFT				
Year = 2003				
Month = January				
Date	Open	High	Low	Last
1/31/2003	47.45	48.35	47.03	47.46
1/30/2003	50.16	50.17	48.19	48.24
1/29/2003	48.73	50.04	47.93	49.91
1/28/2003	49.69	49.7	48.56	48.82
1/27/2003	49.32	50.6	48.41	49.17
1/24/2003	52.03	52.05	49.7	49.85
1/23/2003	51.95	52.54	51.46	52.28
1/22/2003	51.59	52.4	50.91	51
1/21/2003	51.87	52.15	51.29	51.33
1/17/2003	52.94	53	51.31	51.46
1/16/2003	56.32	56.65	55.11	55.35
1/15/2003	57	57.32	56.19	56.27
1/14/2003	56.33	57	56.19	56.97
1/13/2003	56.52	56.75	55.77	56.39
1/10/2003	55.1	56.3	54.9	55.92
1/9/2003	54.72	55.92	54.53	55.81
1/8/2003	55.37	55.55	54.11	54.24
1/7/2003	54.92	56.01	54.68	55.8
1/6/2003	54.02	55.23	53.8	54.77
1/3/2003	53.59	53.8	52.88	53.79
1/2/2003	52.3	53.75	51.71	53.72

FIGURE 1 The HTML report shows the relevant data in tabular form

Adding Content

We add content to the SOAPBody using SOAPElement instances added as child elements. We add to the SOAPBody a SOAPElement that contains the name of the service we are calling. Parts defined for the input message are then added as child SOAPElements (see Listing 6).

Invoking the Web Service

The request content is obtained from the sample input message that was built during WSDL analysis. This initial value could be displayed and edited by the user using a GUI-based tool and then used to populate the request message.

Once we've added the request content to the SOAPBody we can invoke the service. The final steps are to create the SOAPAction header and a URLEndpoint that points to the service's target URL. We give the Connection instance our populated SOAPMessage and the URLEndpoint and invoke the Connection's call() method. The

call() method returns the XML response as a SOAPMessage (see Listing 7).

Making Sense Out of the Response

Web service responses can be very verbose and may contain extraneous data that makes it difficult to find what you are looking for. Even though XML documents are very structured, they are best suited for machines. Most people find it far easier to view an HTML report. A portion of the response from the XigniteQuotes service is shown in Listing 8.

It is important to note that users do not have to write this script themselves. Using a generic XSLT script, the tool automatically generates the script for this particular Web service from the WSDL file. This generic script knows how to traverse a WSDL file and extract the structure of the response message. It is aware of the XML Schema elements such as ComplexType, Sequence, All, Group, minOccurs, maxOccurs, etc. Using this information it infers when to display items as text

fields and when to create tables.

The reporting process can be summarized as follows:

1. WSDL file for Web service + Generic XSLT Script = XSLT script for the Web service
2. Web service response in XML + XSLT script for the Web service = HTML report

The generic script processes the entire response and presents it in an HTML format, which may turn out to be very verbose. Because of this the tool displays the schema of the response, so users can visually select the elements of interest. This will generate a smaller XSLT script that will extract only the pertinent information from the response and present it as a report.

The generic script can be extended using richer XSLT semantics and formatting objects. This will enable us to transform the response into any format, including HTML, PDF, Word, or another XML document (see Figure 1).

Conclusion

Although our sample client is written using the Java language and Java-based APIs, a similar application can be written using tools and APIs available in Microsoft's .NET environment. The fact that WSDL doesn't force a particular programming model is actually one of the beauties of Web services.

WSClient is a useful mechanism for trying Web services. As we have seen you can use it to invoke any Web service and generate easy-to-read reports based on the response.

Our tool also can be used as a first step towards integrating Web services. Users can create a named scenario by specifying a WSDL file, the input XML (which is configurable), and an optional XSLT script. Through an API, the tool can be programmatically instructed to run the scenario identified by its name and return the result. This localizes the complexity of dealing with Web services in the tool and encapsulates the complexity from the invoking program. If used properly, this tool can help you make the transition from tentative evaluation of Web services to their complete adoption. ©

SYS-CON MEDIA

304,187 of the World's Foremost IT Professionals

DIRECT MAIL, EMAIL OR MULTI-CHANNEL

Target CTOs, CIOs and CXO-level IT professionals and developers who subscribe to SYS-CON Media's industry leading publications

Java Developer's Journal...

The leading publication aimed specifically at corporate and independent java development professionals



LinuxWorld Magazine...The

premier weekly resource of linux news for executives with key buying influences



Web Services Journal...The

only Web Services magazine for CIOs, tech, marketing & product managers, VARs/ISVs, enterprise/app architects & developers



XML Journal...The world's #1

leading XML resource for CEOs, CTOs, technology solution architects, product managers, programmers and developers



PowerBuilder Developer's

Journal...The only PowerBuilder resource for corporate and independent enterprise client/server and web developers



ColdFusion Developer's

Journal...The only publication dedicated to coldfusion web development



WebLogic Developer's

Journal...The official magazine for BEA WebLogic application server software developers, IT management & users



WebSphere Developer's

Journal...The premier publication for those who design, build, customize, deploy, or administer IBM's WebSphere suite of Web Services software



.NET Developer's Journal...

The must read iTechnology publication for Windows developers & CXO management professionals



Wireless Business &

Technology... The wireless magazine for key corporate & engineering managers, and other executives who purchase communications products/services

Recommended for a variety of offers including Java, Internet, enterprise computing, e-business applications, training, software, hardware, data back up and storage, business applications, subscriptions, financial services, high ticket gifts and much more.

NOW AVAILABLE!

**The SYS-CON
Media Database
304,187 postal
addresses**



For email information:
contact Frank at 845-731-3832
frank.cipolla@epostdirect.com

epostdirect.com 800-409-4443 fax845-620-9035

For postal information:
contact Kevin at 845-731-2684
kevin.collopy@edithroman.com

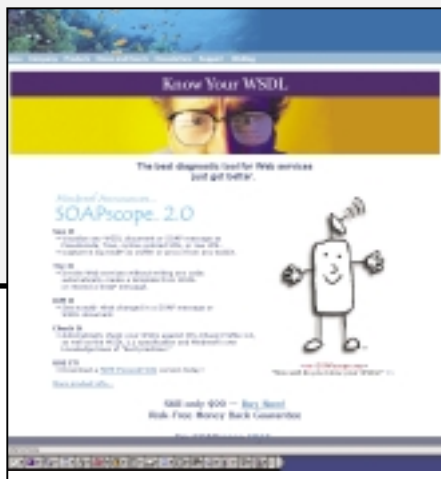
edithroman.com 800-223-2194 fax845-620-9035



Mindreef Releases SOAPscope 2.0

(Boston, MA) – Mindreef, Inc., has announced the availability of Mindreef SOAPscope 2.0, the industry's first Web services diagnostics system. This emerging product category addresses the challenges of isolating and solving problems of applications built with Web services throughout the development, testing, and management process. SOAPscope 2.0 focuses on the interfaces between Web services, described in a WSDL document. It makes WSDL approachable by automating the steps people use when solving Web services problems.

Mindreef SOAPscope 2.0 is priced at \$99 and is available immediately for download at www.mindreef.com



Kamiak Corporation Releases Omniopera Viewer 1.0

(Sheridan, WY) – Kamiak Corporation, which publishes tools for developing Web services, has announced the release of Omniopera Viewer, an Internet browser plug-in that allows users to view WSDL and XML Schema documents in a clear and concise format, but does not require them to have expertise in XML. Users can view a top-level summary of a document, and can drill down to see details.

Kamiak Corporation has also announced the release of Omniopera WS 1.5, their WSDL and XML Schema editor, providing improved support for validation of WSDL and XML Schema documents. www.omniopera.com

HP to Acquire Talking Blocks

(Palo Alto, CA) – HP has signed a definitive agreement to acquire Talking Blocks, a service-oriented architecture and Web services management software company. This acquisition furthers the HP Adaptive Enterprise strategy to help enterprises measure, architect, and manage change by creating a tighter linkage between business and IT. Talking Blocks' standards-based service-oriented architecture helps companies integrate disparate internal systems and the systems of external business partners as well as manage Web services' use in highly flexible environments.

The Talking Blocks' technology will also enable HP to quickly deliver a proven, feature-rich Web services management platform that will support the recently released HP Web Services Management Framework. www.hp.com/go/adaptive

CheckFree i-Solutions Announces i-Series 5 with Enhanced Capabilities

(Atlanta) – CheckFree i-Solutions, provider of e-billing and e-statement applications and part of CheckFree Corporation, has released the newest version of its flagship software platform, CheckFree i-Series 5. With i-Series, organizations can deliver dynamic, interactive invoice, bill, and/or statement content to business customers and consumers from a single, integrated platform.

The new platform delivers greater scalability, performance, and throughput than previous i-Series software platforms. www.checkfree.com

MyAmberPoint.com Provides Knowledge Base and Forum for Community Discussion

(Oakland, CA) – AmberPoint, Inc., has announced its newest support offering, MyAmberPoint.com, a personalized service, discussion, and technical support portal designed to give customers all the resources they need to successfully implement AmberPoint's management solutions and gain greater value from their Web services systems. AmberPoint's Customer Experience Group of consultants, trainers, and support technicians will manage and monitor the password-protected MyAmberPoint.com portal. www.amberpoint.com

Altova Enhances Its Line of XML Development Tools

(Beverly, MA) – Altova Inc., producer of XML-SPY, the world's leading XML development environment, has released the new Altova 2004 XML development tools product line, designed to meet the needs of software developers who are building advanced XML and Web services applications. The new Altova 2004 product line consists of complete updates to existing products, XMLSPY 2004, AUTHENTIC 2004, and STYLEVISION 2004, and introduces MAPFORCE 2004, a visual data mapping tool with code generation capabilities, essential for XML and database integration projects.

Altova's MAPFORCE 2004 is a visual data integration tool that auto-generates custom data mapping code in multiple output languages, such as XSLT and Java, to enable programmatic XML-to-XML or database-to-XML data transformations. It provides a two-step, XML-based approach to enterprise data integration that ensures compatibility and interoperability across different platforms, servers, programming languages, and database environments.

www.altova.com/download



Roving Planet Introduces WLAN Management Platform

(Boulder, CO) – Roving Planet has announced Central Site Director v2.1, its latest enterprise WLAN management, control, and integration platform. The platform includes the first Web services framework and XML-based application program interfaces (APIs), enabling IT managers to quickly and easily leverage existing systems and applications to administer changes to WLAN policies for bandwidth and security management. www.rovingplanet.com

The World's Leading Java Resource!

JAVA DEVELOPERS JOURNAL

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

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

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

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

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

WSJ ADVERTISER INDEX

ADVERTISER	URL	PHONE	PAGE
BEA dev2dev 2003	bea.com/dev2devdays2003	925-287-5156	35
Choreology	www.choreology.com		8, 9
Comdex	www.comdex.com		49
Edith Roman	www.edithroman.com	845-731-2684	55
Ektron	www.ektron.com/ws		11
IBM	ibm.com/websphere/seeit		60
JDJ Store	www.jdjstore.com	888-303-5282	27
LinuxWorld Magazine	www.sys-con.com	201-802-3020	37
Mindreef	www.mindreef.com	603-465-2204	5
Mindreef News	www.mindreef.com/hl0309		4
Parasoft	www.parasoft.com/ws10	888-305-0041	2
Quest Software	www.java.quest.com/jcsc/ws	800-663-4723	59
Strike Iron	www.strikeiron.com		12, 13
SYS-CON Media	www.sys-con.com/suboffer.cfm	888-303-5282	41
SYS-CON Media	www.sys-con.com	888-303-5282	45
The 2003 Borland Conference	connect.borland.com/borcon03		15
Web Services Journal	www.wsj2.com	888-303-5282	29
WowGao, Inc.	www.wowgao.com	416-292-2364	51

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

IN THE NEXT ISSUE OF WSJ...

Focus on Vertical Industries

RosettaNet Moves Ahead

RosettaNet is a consortium of information technology, electronic components, semiconductor manufacturing and telecommunications companies working on their own and with other organizations to create and implement open e-business process standards. We'll look at what they're doing, and why Web services will work for them.



ACORD: Leading the Insurance Pack

ACORD is a nonprofit insurance association whose mission is to facilitate the development and use of standards for the insurance and related financial services industries. This article looks at how the insurance industry is using Web services to meet its goals.



FpML: Moving Money Around the World

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



And...

Production Web Services: The Critical Need for Monitoring and Analysis

Web services are becoming the technology of choice today for solving pressing point-to-point integration needs because of the inherent benefits of the approach. Foremost among these benefits are cost reductions, business agility, and component reuse, which we examine here.



Using Web Services for Business – Delivering the Message

If a solution to problems has been developed in a standards group, then not only will you probably have the leading experts in the world involved, but they are more likely to develop a very good way of solving the problem. In short you should have a "better solution."



Web Services JOURNAL





Think Async

Edwin Khodabakhchian

Edwin Khodabakhchian is the CEO of Collaxa. Previously, he was the CTO of the AOL eCommerce division where he took an active part in the Merchant Integration, Shopping Search, Wallet and AOL 6.0 products. Before the acquisition of Netscape by AOL, Edwin was the chief architect of the Netscape/iPlanet Business Process Manager, a product he drove from inception to release.
edwink@collaxa.com

Within the IT industry, we're seeing a steady shift from an RPC (remote procedure call) integration style to asynchronous, document-based integration. More and more developers are realizing that asynchronous services are core to a new, loosely coupled and message-driven architecture. In practice, while most developers and architects I work with are sold on the value of an asynchronous architecture, it introduces a steep learning curve that makes it more difficult for them to take the next step. Having seen the good, the bad, and the ugly sides of many integration projects, I'll try here to pass on some experiences on the benefits, common hurdles, and best practices of implementing asynchronous services and the systems that support them.

The main benefits that are enabled by asynchronous services and the associated loosely coupled, message-based architecture are:

- **Reliability:** A synchronous integration style results in brittle applications where the weakest link in a chain impacts an entire application or business process. Reliable integration requires asynchronous interactions.
- **Scalability:** Message-oriented middleware vendors and developers have shown that queues provide for maximum throughput and efficiency when connecting systems with different processing and availability capacities. The same approach can be taken with all integration projects by leveraging asynchronous services.
- **Long-running services:** Integration projects frequently bring the greatest value when automating business processes that involve both manual processes and automated systems. Any service that supports or requires manual intervention must have an asynchronous interface due to the time it will likely need to complete.

Taking a real-world example, a developer needs to build a bidding service and wants the benefits of an asynchronous, message-driven architecture. What are the problems he or she might run into?

- **Designing an asynchronous interface:** The goal is to think about the interface to the service as a message-driven/document style interface. The first step is to use XML Schema to describe the documents that will be passed into and out of the service. Similarly, the output of the service should not be thought of as a return value and exception(s) but rather as callback messages. In the XML Web services world, WSDL and PartnerLinkTypes (which specify conversational callback messages) are particularly helpful.

- **Coordinating reliable/asynchronous conversations:** Asynchronous, conversational architectures with a service can be highly reliable but introduce some new problems that are typically not issues for synchronous, RPC-style integration. For example, when it is time to send a callback message to the client of a service, what address should it be called back at, and how should the callback message be correlated with the initiation message? The WS-Addressing specification, promoted by the same software vendors behind the BPEL specification, provides a standard answer to these questions, or a custom solution could be crafted using content-based correlation. Another problem surrounds the reliable delivery of messages over less fault-tolerant protocols like HTTP. A well-behaved asynchronous service should both support a reliable message delivery standard (such as WS-ReliableMessaging) to enable once-and-only-once guaranteed delivery of messages as well as idempotence (the characteristic that sending the same message multiple times has the same effect as sending it once).
- **Managing transactional integrity:** In an asynchronous world, services will need to support compensating transactions (also known as long-running transactions) rather than the traditional XA model of locking transactions. This means that services should implement pairs of operations (doXXX and undoXXX) so that a client of the service, or a transaction manager, as standards like WS-Transaction take hold, will be able to cancel a fully "committed" operation if necessary. A related facet of this problem is that of exception management and task services. In an asynchronous environment, people and manual tasks can be inserted in the workflow of a service or business process when appropriate. Frequently this enables support of existing manual processes for exception cases when automating a flow, and speeds adoption (and thus ROI) of the automated system.

As you can see, there are many standards (BPEL4WS, SOAP, WSDL, WS-Addressing, WS-Transaction, etc.) that assist in implementing asynchronous architectures. These standards provide a framework for an asynchronous architecture and bring significant value in both the design and implementation phases of an integration project by supporting asynchronous interactions as first-class citizens. But all of these standards and technology are the second step – the first step is for developers and architects to learn to be more comfortable designing asynchronous, loosely coupled services and systems. ©

Quest Software

www/java.quest.com/jcsc/ws

IBM

ibm.com/websphere/seeit