

# SOA Web Services JOURNAL

NOVEMBER 2006 / VOLUME: 6 ISSUE 11

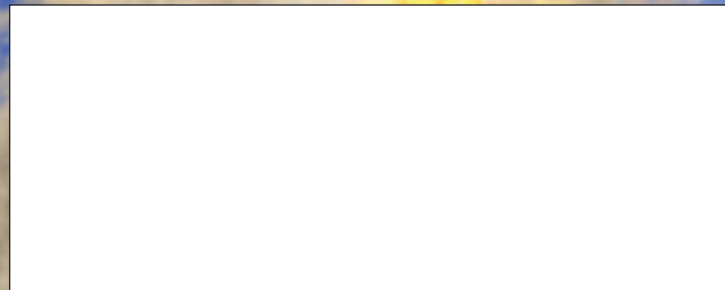
## THE SKYROCKETING ESB MARKET

### THE ESB IN YOUR SOA

Specifications  
**Web Services,  
WS-\* Specifications,  
and Interoperability**

Data  
**Complete Data Integration  
Through XQuery**

Governance  
**The Technologies Behind  
SOA Governance**



**AJAXWORLD  
UNIVERSITY  
BOOTCAMP**

**Jan. 22, 2007**

Coming to New York City

**Hands-On AJAX Training!**

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



# Gear up for XML excellence

Take off with the Altova® XML Suite, and  
save ½ off the top tools for XML development.



## Included with the Altova XML Suite 2007:

- Altova XMLSpy®, MapForce®, and StyleVision® Enterprise or Professional Editions
- Plus Altova SchemaAgent™, SemanticWorks™, and DiffDog® with Enterprise Edition
- Also get a FREE copy of Altova DatabaseSpy™ 2007 for a limited time\*

The Altova XML Suite 2007 delivers the latest releases of world's leading XML development tools all in an unrivaled deal. It contains Altova XMLSpy, the industry standard XML development environment; MapForce, the premier data integration and Web services implementation tool; and StyleVision, the ultimate visual stylesheet designer. What's more, the Enterprise Edition also includes XML Schema management, Semantic Web, and XML-aware differencing tools. Save a bundle!

Download the Altova XML Suite today: [www.altova.com](http://www.altova.com)

Join Altova at  
Microsoft Connections,  
Las Vegas  
Booth #407

\*Special offer: Now until Dec. 24, 2006, purchase or upgrade to the Altova XML Suite and get the NEW Altova DatabaseSpy database query and design tool for FREE! Conditions apply, see Web site for details.

**ALTOVA®**

Visit us online at [WebServices.SVS-CON.com](http://WebServices.SVS-CON.com)

# Inside This Issue

## ENTERPRISE SERVICE BUS

16

James Pasley



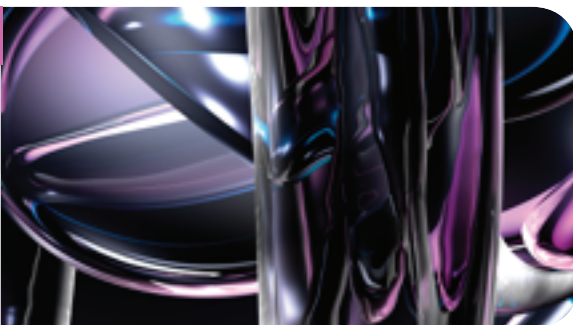
## The ESB in Your SOA

Why the ESB market is skyrocketing

## SPECIFICATIONS

20

Sanjay Narang



## Specifications and Interoperability

Achieving interoperability is neither simple nor straightforward

## BEST PRACTICES

28

Dan Foody



## SOA Worst Practices

When the "right thing" goes wrong — a safety ladder

## FROM THE EDITOR

### SOA for SaaS

By Sean Rhody

5

## DATA

### Complete Data Integration Through XQuery

By Jonathan Robie

8

## GOVERNANCE

### The Technologies Behind SOA Governance

By Gary So

12

## C/C++ FUNCTIONALITY

### Exposing C Apps as Web Services

By Mohit Chawla and Vijaya Bhaskar Peddinti

32

## ENTERPRISE

### Keep to the Original Intent of SOA

By David Besemer

36

## CODE

### Is It Done Yet?

By Dr. Adam Kolawa

38

## NEWS

### SOA Web Services and more!

42

## ENTERPRISE ARCHITECTURE

### Considering the SOA Reference Model

By Michael Poulin

46

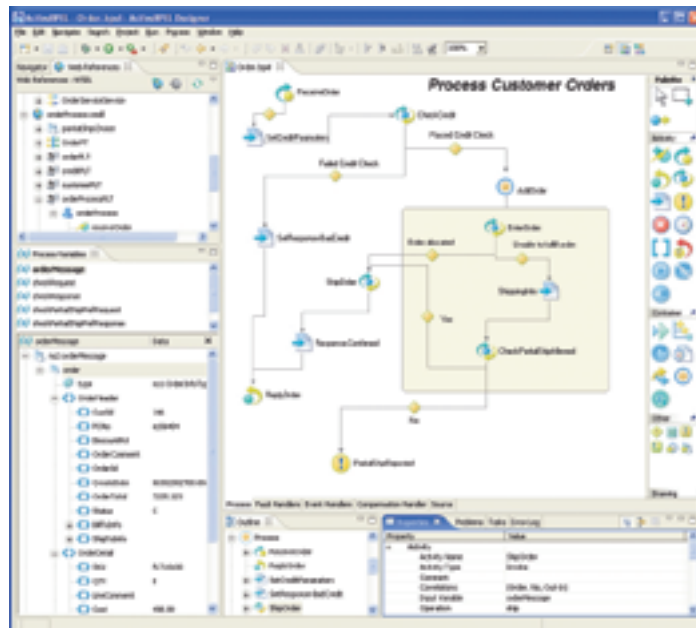
## STANDARDS

### Open SOA Collaboration

By David S. Linthicum

50

# Get Started with BPEL 2.0



**Build next-generation SOA applications  
with the leader in BPEL technologies**

**Download BPEL tooling & server software today**

**[active-endpoints.com/soa](http://active-endpoints.com/soa)**

**BPEL consulting and training.**

**BPEL design tools, servers and source code for Eclipse, Apache Tomcat, JBoss,  
WebSphere, WebLogic, BizTalk and Microsoft .NET.**

**activeBPEL**

**active  
endpoints**

**INTERNATIONAL ADVISORY BOARD**

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

**TECHNICAL ADVISORY BOARD**

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

**EDITORIAL****Editor-in-Chief**

Sean Rhody sean@sys-con.com

**XML Editor**

Hitesh Seth

**Industry Editor**

Norbert Mikula norbert@sys-con.com

**Product Review Editor**

Brian Barbash bbarbash@sys-con.com

**.NET Editor**

Dave Rader davidr@fusiontech.com

**Security Editor**

Michael Mosher wsjsecurity@sys-con.com

**Research Editor**

Bahadir Karuv, Ph.D Bahadir@sys-con.com

**Technical Editors**

Andrew Astor andy@enterprisedb.com  
David Chappell chappell@sonicsoftware.com  
Anne Thomas Manes anne@manes.net  
Mike Sick msick@sys-con.com  
Michael Wacey mwacey@csc.com

**International Technical Editor**

Ajit Sagar ajitsagar@sys-con.com

**Executive Editor**

Nancy Valentine nancy@sys-con.com

**PRODUCTION****ART DIRECTOR**

Alex Botero alex@sys-con.com

**ASSOCIATE ART DIRECTORS**

Abraham Addo abraham@sys-con.com  
Louis F. Cuffari louis@sys-con.com  
Tami Beatty tami@sys-con.com

**EDITORIAL OFFICES**

SYS-CON MEDIA  
577 CHESTNUT RIDGE ROAD, WOODCLIFF LAKE, NJ 07677  
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  
Woodcliff Lake, NJ 07677 and additional mailing offices  
POSTMASTER: Send address changes to:  
WEB SERVICES JOURNAL, SYS-CON Publications, Inc.  
577 Chestnut Ridge Road, Woodcliff Lake, NJ 07677

**©COPYRIGHT**

Copyright © 2006 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.

# SOA for SaaS

WRITTEN BY SEAN RHODY



If I were a lot more paranoid than I am (well, perhaps at least a little more than I am), I might suspect that the various free e-mail programs were a social engineering attempt by the big software coalition (yes, I know, it doesn't exist) to ultimately change the way we use our computers.

What does this have to do with service-oriented architecture? I'll get there, don't worry. But first, as to why would this be social engineering, let's take a look at the concept of Software as a Service (SaaS, for acronym fans). Software as a Service is not an entirely new concept, but it's one that has always faced an uphill battle. Previously, and even now to a certain extent, continuous connectivity was a serious issue. Reliance on software provided across the Internet can be problematic if the connection is slow or interruptible. While broadband access has solved this problem for many, there are still many challenges to be faced with software that doesn't reside on your local hard disk.

This challenge is something that faces both consumers and businesses – while a business may be able to secure faster, more reliable communication through redundant network access, slow response time can be a fact of life in such a scenario. Even then, there's no guarantee that an upstream DNS server may not be under attack or down. In the case where the software is local, this is not a problem, and might even increase productivity slightly by preventing employees from surfing the Net for a period of time.

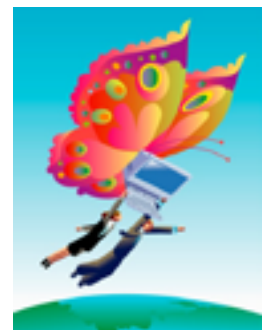
However, when the software sits in cyberspace (say that three times quick), it becomes a different matter. Downtime costs money, as do delays. I've heard numbers relating to call centers that say saving one second of call time results in a savings of a million dollars over a year. So, turning that around, delays introduced due to network latency could cost millions of dollars – not something a thin margin business wants to consider.

So social engineering comes into play. Free services, such as e-mail, start to address some aspects of this issue. First of all, with steady broadband connections, we can see that by and large, the service provided is acceptable. Not perfect, I think we'd all admit – and in many cases the interfaces provided are inferior to those of a thick client e-mail application. But, for the most part, you get reliable e-mail access, as well as reasonable functionality, without having to have a home e-mail server or any installed software. You begin to see that Software as a Service is possible.

There are other examples of services that aren't free, such as salesforce.com, that are successful in providing Software as a Service. Depending on your definition of software and of service, we could identify many more.

Service-oriented architecture, founded on the principle of loose coupling and the distinction between a service and an application, is the perfect driver for Software as a Service. As organizations begin to decouple their services from the silos that an application focus has placed them in, they begin to recognize the intrinsic value of the service and its relationship to their business processes. The value of an application does not become less, but the nature of an application changes to one where services form the core of the undertaking, and the presentation aspects of the application become the unique value.

With that in mind, it becomes easier for Software as a Service to become a reality. An organization already making use of services finds it simple enough to integrate an external service. The reverse is also true. An organization using services finds it easier to expose its own services to external customers – in essence to take its services to market by providing Software as a Service. ■

**About the Author**

Sean Rhody is the editor-in-chief of SOA Web Services Journal. He is a respected industry expert and a consultant with a leading consulting services company. sean@sys-con.com

**IBM®**





**WebSphere®**

\_INFRASTRUCTURE LOG

\_DAY 18: Came to work and found everything frozen. Icicles are everywhere. It's our processes. They're inflexible. Hard coded so we can't respond to change.

\_Why did we lock ourselves in like this? Brrrr.

\_DAY 19: A way out. IBM WebSphere middleware for Business Process Management. It lets us streamline business tasks and optimize performance. We can simulate and test our processes so we understand the impact they'll have, then monitor performance once they're deployed. And because it's based on a service oriented architecture, it's easy to reuse and connect existing process-based services.

\_Everything's unfrozen now. Wow, it's good to feel my toes again.

Take the BPM with SOA Assessment at:  
[IBM.COM/TAKEBACKCONTROL/PROCESS](http://IBM.COM/TAKEBACKCONTROL/PROCESS)

# Complete Data Integration through XQuery

## Vastly simplifying SOA implementations

WRITTEN BY JONATHAN ROBIE

➤ Most businesses have an urgent need for up-to-date, accurate information based on data from multiple data sources. It would be much easier if all your data were stored in one database so it can be queried as a whole, but this is rarely practical. In the real world, data integration is required. You need a simple, efficient way to query data found in various data sources.

Suppose you use information about customers in your Service Oriented Architecture (SOA). Your company might use an external CRM system like salesforce.com for leads and customer data, an internal ERP system like PeopleSoft for processing orders, a dedicated software solution to track technical support calls, and one or more databases to store customer information not captured by these other systems. Each of these data sources has a different representation of a customer, a different API and data model, and perhaps a different query language. Nevertheless, you have to be able to combine this data intelligently to get an overview of a customer.

For instance, you may want to generate one report with the overall status of a customer, or you may want to find all customers with outstanding tech support issues who are deciding whether to make a major purchase this quarter. If all of your data were in a single database, you would retrieve the information with a simple query. Because the data is in many different sources, you have to write a good deal of code to get the same result, and the code is quite different for each data source. This is time-consuming, error prone, and complicates security and auditing. With XQuery, you query each data source as though it

were XML, no matter how the underlying data is physically stored.

XQuery is the World Wide Web Consortium (W3C) standard XML query language, designed for both XML processing and data integration. Using XQuery for data integration vastly simplifies SOA implementations, making your developers more productive and improving the performance of your systems. An XML Integrated Development Environment (IDE) that supports XQuery makes it much easier for you to visualize data sources, generate and test queries, and debug. The queries you develop can be exposed via a data access layer, which is accessed using SOAP or HTTP, so that they can be reused in different SOAP message formats or in other applications.

### XQuery Simplifies Data Integration

XQuery simplifies data integration in two ways. First, it provides native support for XML and for the operations most frequently needed when working with XML. Today, XML is at the heart of most data integration, and this is certainly true for SOA environments where every SOAP message is expressed in XML. Most languages don't support XML natively. In general, programming languages are based on objects or



structures where query languages are based on relational tables and scripting languages are based on text. XQuery is based on XML, and XML is the only data structure in XQuery. In the same way that SQL queries tables to produce tables, XQuery queries XML to produce XML — and the XML produced by an XQuery can be used directly in XML applications. For example, a query result might be the payload of a SOAP message. XQuery provides direct support for querying, creating, and transforming XML.

One frequently used expression in XQuery, the FLWOR expression, is similar to SQL's SELECT-FROM-WHERE. Because XML structures are more complex than SQL tables, XQuery provides path expressions that can identify any item in an XML structure. To create structures in query results, it also provides constructors, using a syntax that looks like the XML to be constructed. A typical XQuery might use path expressions to locate data, FLWOR expressions to perform joins and combine data, and constructors to create the structures of the query result. These tasks are much more tedious with conventional programming languages. For instance, to achieve the same result with the Java DOM API, this would require parsing, navigating object structures, casting values from XML into Java data types, creating a

result tree structure, and appending nodes to that result tree. In general, conventional programming languages require seven to 20 times more code than an equivalent XQuery. Not only are XML applications harder to write in conventional programming languages, performance can be much better in a good XQuery implementation, because XQuery is a declarative language that allows the implementation to do many useful kinds of query optimization.

The second way XQuery simplifies data integration is by eliminating the need to work with different APIs and data models for each data source. The XQuery language is defined in terms of XML structures, but since almost any data can be mapped into XML structures, an XQuery implementation can use XQuery to query just about anything. For instance, an XQuery implementation can provide support for relational data, implementing queries by generating efficient SQL for the database, but allowing a user to query the data as though it were XML.

By treating all data sources as XML, this kind of XQuery implementation lets a developer query relational data, Web message calls, and other data sources together, with a small amount of declarative code, in one uniform data model, without mastering the idiosyncrasies of each system.

Consider the customer example in the introduction. With an XQuery implementation that supports all of the underlying data sources, a developer can write a simple query to do a join among the different systems that represent different aspects of a customer. This dramatically simplifies software development in most business environments. The developer focuses on the information that's needed, not on the representation used in each system. Typically, the code savings in data integration environments is even greater than in pure XML environments.

The available data sources and the implementation strategy vary widely among XQuery implementations. For relational data, an implementation may translate an XQuery into SQL then translate the SQL result sets to XML when returning results to the query engine. For flat file formats, an implementation can provide XML converters that actually convert data to XML on-

the-fly when it's queried. Web Service calls may be supported using functions that can be called from within a query. When choosing an XQuery implementation, make sure that it fits in your computing environment and can handle the data sources needed in your architecture. The XQuery implementations from most database vendors are designed to query only data stored in their database; most companies have more than one database, and data not found in a database.

The XQuery implementations from application server vendors or XML integration server vendors can query a wider range of data sources, but require the adoption of their server, which may not fit in your architecture, or may increase the footprint of the system. If you're writing Web Services in a Java environment, make sure your implementation supports the XQuery API for Java (XQJ), which is the standard Java interface for XQuery – it lets your servlets use XQuery the same way that JDBC lets servlets use SQL. Also, the performance of XQuery implementations varies dramatically – make sure that you test performance for the data you work with, especially if you're using XQuery for relational data or very large XML files. Because XQuery is declarative and can be optimized, a good implementation will provide performance better than you normally achieve with hand-coded Java, JDBC, SQL, and an XML API.

Using XQuery vastly simplifies data integration, offering loosely coupled access, and providing one way to query any data source supported by the query engine. And because an XQuery implementation can talk directly to the original data source, it can do optimizations that are no longer available once the data is extracted and converted to physical XML. As a result, what is easier for the developer also results in better performance.

## XML Development Environments for Data Integration

It's hard to understand the relationships among data without some way to visualize the data. This is particularly true when working with data from multiple sources. When doing data integration, look for an IDE that lets you visualize as many of your data sources as possible, supports general

## CORPORATE

### President and CEO

Fuat Kircaali fuat@sys-con.com

### Group Publisher

Jeremy Geelan jeremy@sys-con.com

## ADVERTISING

### Senior VP, Sales & Marketing

Carmen Gonzalez carmen@sys-con.com

### VP, Sales & Marketing

Miles Silverman miles@sys-con.com

Robyn Forma robyn@sys-con.com

### Advertising Manager

Megan Mussa megan@sys-con.com

### Associate Sales Managers

Kerry Mealia kerry@sys-con.com

Lauren Orsi lauren@sys-con.com

## SYS-CON EVENTS

### Associate Event Manager

Lauren Orsi lauren@sys-con.com

## CUSTOMER RELATIONS

### Circulation Service Coordinator

Edna Earle Russell edna@sys-con.com

## SYS-CON.COM

### VP information systems

Robert Diamond robert@sys-con.com

### Web Designers

Stephen Kilmurray stephen@sys-con.com

Richard Walter richard@sys-con.com

## ACCOUNTING

### Financial Analyst

Joan LaRose joan@sys-con.com

### Accounts Payable

Betty White betty@sys-con.com

### Accounts Receivable

Gina DeTemple gina@sys-con.com

## SUBSCRIPTIONS

SUBSCRIBE@SYS-CON.COM

1-201-802-3012 or 1-888-303-5282

For subscriptions and requests for bulk orders, please send your letters to Subscription Department

Cover Price: \$6.99/issue

Domestic: \$69.99/yr (12 issues)

Canada/Mexico: \$89.99/yr

All other countries: \$99.99/yr

(U.S. Banks or Money Orders)

### Worldwide Newsstand Distribution:

Curtis Circulation Company, New Milford, NJ

### For list rental information:

Kevin Collopy: 845 731-2684, kevin.collopy@edithroman.com;

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

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

# Businesses need up-to-date information that comes from a variety of data sources, but the proper tools and development methods have lagged behind

XML functionality, and has good support for XQuery. Some of these tools let you establish database connections, drag-and-drop from data sources to create XQuery code, run queries and see their output, and run a debugger to help find bugs. These tools make developers more productive.

When choosing an IDE for data integration with XQuery, consider related functionality that you may need. For instance, some IDEs also provide support for developing XML pipelines and publishing. Several IDEs can generate XQJ code to run an XQuery as part of a program. One XQuery IDE is implemented as an Eclipse plug-in, which is very convenient for Java developers who use Eclipse. Several IDEs also provide good support for writing and testing XSLT stylesheets, W3C XML Schemas and DTDs, and related XML development.

## The Data Access Layer

In most companies, several data consumers need to access the same information. For instance, if one of your Web Services needs a description of a customer, this same description might also be useful for other Web Services, and also for dynamic Web sites, AJAX clients, publishing applications, or any other application that needs customer data. Frequently companies design for a single project, coding very similar interfaces for each data consumer, an obvious waste of programming effort. And if the data sources change, each of these interfaces has to be rewritten. In environments where security and auditability are important, much more code must be audited.

A data access layer lets many data consumers access data using the same well-defined interface. For each request, the data access layer calls a data service. Data services should represent the business model, hiding underlying systems and the data integration task from data consumers. For instance, you might write a data service that provides the data for a single customer. A data service can be parameterized – a parameter might

identify the customer ID or the name of a particular view of the customer.

Many data services do nothing more than query data from one or more data sources to produce XML. These data services can be written directly in XQuery, using external variables to allow queries to be parameterized. In other data services, an XQuery may be part of a Java program that performs business logic or interacts with other systems, or it may be part of an XML pipeline. A small focused team can be responsible for writing the queries to implement data services, and for documenting available services, allowing data consumers to access these services using standard Web and XML interfaces.

## Summary

XQuery provides a simple way to query data across data sources, providing simple, efficient data integration. With a good XQuery implementation, any data source can be queried as though it were XML, and any desired XML structure can be created as the result of a query. For instance, a query can take relational data and other data sources as input, and return the payload for

a SOAP message. XQuery increases productivity by freeing developers from the need to learn a different API and data model for each data source, and provides direct support for the operations commonly used in XML. Depending on the XQuery implementation, data sources might include relational data from one or more databases, XML files, Web Service calls, EDI, and legacy file formats among others. An XML development environment that allows visualization of multiple data sources and provides support for XQuery can further enhance developer productivity. When many data consumers need access to the same kinds of data, data integration can be done in a data access layer that provides a set of data services, representing the business model, that hide the details of data integration and allow reuse of data integration code.

Because businesses need up-to-date information that comes from a variety of data sources, but the proper tools and development methods have lagged behind, today's software systems are often needlessly complex and ad hoc. Modern data integration tools are the solution. Using XQuery, an XML IDE, and a data access layer simplifies development significantly, improves performance, increases code reusability, and makes systems more maintainable. ■



## About the Author

Jonathan Robie is the XQuery Technology Lead at DataDirect Technologies, which specializes in data integration, data connectivity and mainframe integration products. He is a lead designer of XQuery, the W3C XML Query language, and an editor of many of the specifications which define the XQuery language. He is also a co-inventor of Quilt, a predecessor of XQuery, and XQL, a predecessor of XPath. Jonathan has been significantly involved in several other W3C Working Groups, acting as an editor for documents produced by the XML Schema and Document Object Model Working Groups, and has also participated in the W3C XML Information Set and XML Stylesheet Language (XSL) Working Groups. He is well known in the XML world, both as an innovator and as a speaker. Jonathan's Blog can be found at [http://blogs.datadirect.com/jonathan\\_robie](http://blogs.datadirect.com/jonathan_robie).

# Facing a few barriers on the road to SOA?



## **JackBe's Rich Enterprise Application (REA) platform clears the road to SOA business benefits.**

There's an abundance of products and vendors to help you *create* your SOA. Now, *consume* those SOA services with JackBe REAs to achieve the business productivity and results that led you to SOA in the first place. Our new REA platform combines the power of Ajax and SOA with reliable, secure communications to extend your SOA directly into powerful business applications.

A fully visual IDE reduces manual coding and accelerates the development process. And our lightweight, vendor-neutral platform easily integrates with existing middleware and services while maintaining server-side governance and control--unlike products that leave governance to the browser or create middleware platform dependencies.

Join over 35 industry leaders like Citigroup, the U.S. Defense Intelligence Agency, Sears, Tupperware, and Forbes who are already optimizing their business activity with JackBe solutions. Call or visit our website—let us help you remove the barriers on the road to achieving real business value from your SOA investment.



**Web:** [www.jackbe.com](http://www.jackbe.com)

**Phone:** (240) 744-7620

**Email:** [info@jackbe.com](mailto:info@jackbe.com)

# The Technologies Behind SOA Governance

**SOA governance is not a shrinkwrapped product that you can simply implement off-the-shelf**

WRITTEN BY GARY SO

➤ In last month's article, we discussed the motivation for SOA governance and the areas where governance should be applied. We also pointed out that, while SOA governance is not a shrink-wrapped product that you can simply implement off-the-shelf without also addressing important organizational and procedural issues, putting the right software mechanisms into place enhances the ability to automate the enforcement of policies and controls.

**T**his article outlines the key moving parts of an SOA governance system that performs such policy-related functions.

At a basic level, an SOA governance system should facilitate governance across the service lifecycle from design time to run-time to change time. It should allow policies to be defined and created, and provide mechanisms for these policies to be enforced at each phase of the service lifecycle.

The main components of this system include:

- A registry that acts as a central catalog of business services
- A repository for storing policies and other metadata related to the governance of the services
- Policy enforcement points, which are the agents that enact the actual policy enforcement and control at design time, run-time, and change time
- A rules engine for managing the declaration of policies and rules and automating their enforcement
- An environment for configuring and defining policies and for managing governance workflows across the service lifecycle

## Registry

A registry is usually identified as one of the first requirements of SOA adoption and registries play an important role in governance. In simple terms, a registry is a catalog or index that acts as the "system of record" for the services in an SOA. A registry isn't designed to store the services themselves rather it indicates their location by reference. Having a centralized catalog of services is significant from an organizational perspective because it enables the easy discovery, reuse, and management of services.

An SOA registry typically fulfills the following functions:

- Stores service descriptions, information about their end-points (the network resource where the service functionality is implemented), and other technical details that a consumer requires to invoke the service such as protocol bindings and message formats
- Allows services to be categorized and organized
- Allows users to publish new services into the registry and to browse and search for existing services
- Maintains service history, allowing users to see when a service was published or changed



As the place where services are made known in the SOA, a registry is also a natural management and governance point. For example, compliance requirements — such as conformance with the WS-I Basic Profile or the use of specific namespaces and schemas — might be imposed on services before they can be published in the registry. Or, as services are registered or changed, the registry can also trigger approval and change notification workflows so that stakeholders are alerted to changes. As such, a robust registry is an important component of any SOA governance solution.

Another important factor is the interoperability of the registry with other components of the SOA infrastructure. OASIS provides a platform-independent standard for registry interoperability known as UDDI (Universal Description, Discovery, and Integration). UDDI defines a Web Services-based programming interface that allows different consumer applications, tools, and runtime systems to query the registry, discover services, and interact as required to provide management and governance capabilities. While it's not a prerequisite for an SOA registry to be based on UDDI, it is the most commonly adopted standard and ensures

the greatest degree of compatibility with other products in the environment.

## Repository

While the registry plays a central role in policy enforcement, the registry itself doesn't provide sufficient context for the breadth of SOA governance requirements described in this article. For example, policies — in the form of rules and restrictions that are enforced on services — and consumer/provider service-level agreements are generally not constructs that are stored in a registry (for one reason, they aren't constructs that are known to UDDI). So another data store, usually referred to as a repository, is needed to store governance-related artifacts and support the full complexity of managing service metadata throughout the service lifecycle.

The term “repository” is used in many different contexts — for example, a data repository (SQL database), a document repository (file system), or a source code repository (version control system) — but in the context of SOA governance, the repository can be thought of as a centrally managed policy store.

Among other things, a governance repository should support the following capabilities:

- An information model or taxonomy for representing and storing organizational and regulatory policies that can be translated into rules that are enforced by the SOA governance system. It should be possible for policies and rules to be interpreted by people or machines (and sometimes both) as appropriate.

- Audit capabilities for tracking the trail of changes and authorizations applied to assets in the repository context.
- Identity management capabilities and role-based access controls to ensure that only appropriate parties have access to policies.
- A notification system and content validation capabilities to provide additional assurances that policies are well-formed, consistent, and properly applied.

The requirement for a logically centralized repository is particularly important for codifying and enforcing a single “official” set of policies across the organization. However, the actual repository itself may have a federated architecture for scalability and for using the repository across different geographic regions, multiple lifecycle constituencies, and corporate boundaries.

## The Advantages of an Integrated Registry/Repository

While the registry and repository have been discussed as separate concerns, in practice there are many benefits to combining the two functions into a single entity. To function properly together as a system, the registry and repository should maintain a consistent view of service definitions, service versions, consumer and user identities, and other information. Implementing them as separate products creates the burden of duplicate data entry, sets up the need to synchronize information, and increases the risk of inconsistencies between the two.

When the registry and repository are unified with a single normalized and standards-based information model and underlying datastore, the integrity of both governance-related metadata and the information model can be assured. The unified approach also eases the enforcement of policies that apply across the boundary between the registry and the repository.

Standard registry capabilities can be offered by an integrated registry/repository with a UDDI interface to the policy repository that allows access to the relevant data.

## Policy Enforcement Points

So far the role of the registry and repository has been established and the case made for an integrated registry/repository to create a consistent policy context. These policies, in turn, need to be enforced. In an SOA governance system this function is fulfilled by policy enforcement points.

The places where policies are actually applied and enforced — the policy enforcement points — change depending on the lifecycle stage. During design time, the registry/repository itself is the point of enforcement. During runtime, policies are generally enforced by the underlying message transport system that connects service providers with consumers. Finally, during change time, policies are typically enforced by the IT management system.

## Design-Time Enforcement: Registry/Repository

Since the registry/repository is the system of record for both service interfaces as well as the assets, attributes, and metadata associated with them, it provides a logical point at which to enforce policies about the design of these particular artifacts. Design-time policies are typically applied as artifacts — which could include WSDL files, schema definitions, process models, and project documentation — are checked into the registry/repository.

The following features are desirable in the design-time policy enforcement point:

- **Identity management:** To establish rights and responsibility in the registry/repository it's first necessary to identify users (for example, via an LDAP-based identity systems or other directory), service consumers, and other participants. Identity is also important for metering usage, logging for audit purposes, and applying approval requirements and other governance processes on an individual or role-based basis.

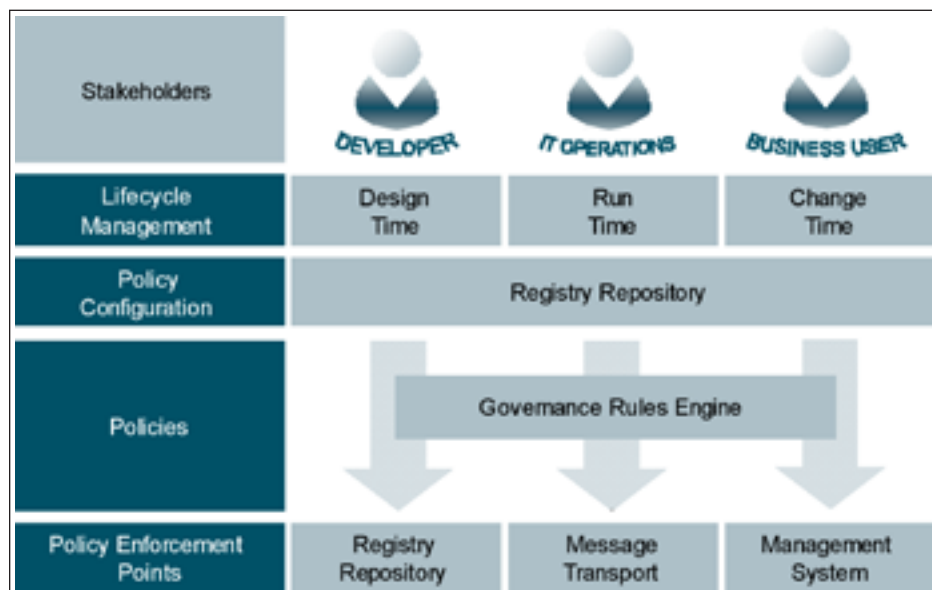


Figure 1: Key components of an SOA governance system

# By establishing the right balance between strategy, organizational practices, and supporting technologies, companies will be able to turn the concept of SOA governance into a practical reality

- **Access control:** Coupled with identity management, the system should offer fine-grained access configurations over all aspects of registry/repository assets. This includes the ability to secure assets as well as policies, governance processes, and asset classifications.
- **Automated notifications and approvals:** The ability to trigger events in response to Create, Read, Update, or Delete activities in the registry/repository allows alerts, approval processes, content validation scans, and other actions to be automated. These triggers might be applied either before or after the interaction in question. For example, a policy might be established that a design review approval is needed before an object is created in the registry/repository.
- **Content validation:** Content (in the form of assets that are checked into the registry/repository) should be scanned and validated according to their type and pre-configured compliance checks. Common validations include WSDL validation, XML schema validation, testing for namespace violations, schema validation, and other interoperability-related scans. For example, service consumers expect interfaces to be well formed and interoperable, so the registry/repository should automate the process of scanning and assuring that WSDL documents are well formed and conformant with WS-I interoperability profiles.
- **Audit trails:** A fundamental capability for establishing accountability is the ability to track interactions between participants and the registry/repository, along with the sequence and details of those activities. This record can be used for governance enforcement “after the fact” and to establish usage patterns for guiding process improvements.

## Runtime Enforcement: Message Transport

Runtime policies are enforced by the SOA mediation layer — which will typically be a SOAP/HTTP-based message transport — along with the registry/repository that fulfills the function of service discovery and policy store. As an intermediary between service providers and consumers, the message transport can exercise policy controls transparently to the underlying services and to the message flows between them.

Without SOA, the ability to control applications in this manner is restricted both by the scope and capabilities of the underlying platform. When different applications are integrated, it is generally difficult to apply a common policy context to the integrated result. A typical challenge is enforcing access security when two applications with different user communities are integrated. For example, application A automatically draws data from application B, but application A users aren't authorized to use application B. How do you now control the data that users have gained access to in application A? With the intermediation provided by the message transport, it becomes possible for a distributed network of services to share a common policy-managed context. This is a powerful capability that emerges as a direct result of SOA.

Since runtime policies are typically applied to messages that flow across the message transport system, the specific types — and level of sophistication — of runtime policies that can be defined and enforced depend on the capabilities of the underlying intermediary.

Desirable runtime policy enforcement capabilities include the following:

- **Consumer identification and security:** Identifying consumer applications to prevent unauthorized access to services; configuring the security of services at

runtime and enforcing policies such as encryption (for message confidentiality), digital signatures (for message integrity and nonrepudiation), and logging for tracing and tracking.

- **Routing rules:** Configuring runtime routing rules to address performance, version management, and other operational requirements. Variations include: content-based routing (examining a message's content to determine where to route it according to specific rules, for example, processing certain types of orders via a different process); version-based routing (to support version management and the deprecation of services); and preferential quality-of-service routing (giving consuming applications different processing priorities based on business priorities and defined service levels).
- **Transformation rules:** Translating between different message transports and technology protocols to facilitate application connectivity, or transforming data between consumer and provider.
- **Service Level Agreement (SLA) management:** Policies for managing performance and availability to match the requirements of an SLA, for example, routing a request to a backup service in the event of a failure of the primary service provider, or balancing the request load across additional back-end service to improve performance.
- **Logging, monitoring, and alerting** — Collecting service-level data and establishing rules based on aggregate counters for response time, throughput, errors, and other transaction data so that alerts can be generated when there are violations to predefined SLAs.

Finally, while the intermediary and registry/repository are logically decoupled, a

dependency exists to the extent that the intermediary has to understand and interpret the policies defined in the registry/repository. As such, it's advantageous to have a message transport system and registry/repository that are interoperable out-of-the-box; otherwise, this is an integration issue that the implementer has to address.

**Change-Time Enforcement: IT Management System**

Whereas design- and runtime policy have "hard" gates – approval for access to the registry/repository and connectivity via the message transport – change-time enforcement relies to a greater extent on IT change management practices and procedures than on enforced control points. Nevertheless, the IT management system — collectively, the set of systems management software tools and services that the IT organizations uses to manage, monitor, and control their business applications and software infrastructure — and the registry/repository combine to play an important role in change-time governance.

Unlike previous software paradigms where an application package enters a support or maintenance phase once put into production, SOA involves a dynamic network of interdependent services that are in an ongoing state of adaptation and optimization. Since services, transactions, and SOA events of interest can be monitored by the IT management system, it's a logical source of runtime information that can be fed back into the registry/repository to facilitate the orderly evolution of the SOA environment.

This information might include:

- SLA-related metrics, such as the average response time, availability, or throughput of a specific service
- Process-related metrics in the form of Key Performance Indicators (KPIs) that associate services with user-defined business process metrics (e.g., average order amount)
- Business activity monitoring (BAM) alerting and notification events related to business-level exceptions.

Information such as this can be used to optimize service delivery during the change-time cycle by guiding adjustments in policies, service levels, or in the services themselves. Changes to services will require the change-time governance practices described earlier to be put into effect, for example, performing an impact analysis to

assess the implications of changing a service and dealing with the resulting version management issues.

As with integration between the message transport and the registry/repository, it's beneficial to have out-of-the-box linkages between the registry/repository and the management system so that data flows seamlessly between the two without needing additional integration.

## Governance Rules Engine

A rules engine isn't strictly a requirement of an SOA governance system, but incorporating rules-engine technology in the registry/repository enables a significant degree of flexibility and automation, while reducing the reliance on humans to perform mechanical governance tasks (and the associated risk of error).

Rules are typically associated with events, while the rules engine handles the firing and chaining of rules. For example, a company's policy might be that access to services in production is strictly limited to staff belonging to an IT operations role, whereas in the test environment, access is also granted to developers. The rules engine could automate the process of setting and resetting access control switches at lifecycle milestones such as when a service is promoted from development into testing or production. A rules engine also provides the basis for creating complex policies based on reusable templates.

Besides automating governance tasks, the rules engine can help deal with policy federation, or the ability to allow multiple policy authors and authorities. This is an important use case for enterprise SOA adoption where governance policies might not be authored and controlled by a single department or organization. A more robust model — that is the basis for policy federation — is to enable both centralized as well as distributed policy creation. Policy federation requires the establishment of guidelines and rules for reconciling policies that come into conflict, and the rules engine assists in the execution of these rules.

## Lifecycle Management

The final key ingredient of an SOA governance system is the user environment that presents the human interface to the registry/repository and incorporates the governance lifecycle processes and workflows. Typically, the process workflow includes the following steps:

- Publishing a service by an authorized provider
- Discovering a service by a potential consumer
- Requesting use of the service by the consumer
- Agreeing on the terms of delivering the service
- Authorizing the consumer
- Provisioning of the service
- Monitoring of the service delivery

Related to each of these steps, organizations might define approval and notification workflows, exception alerts, and a variety of other process steps. The SOA governance lifecycle management capability will manifest itself in the forms of screens — with information customized to the user's role in the governance framework — reports, and integration with e-mail to communicate notifications and approval requests. An important feature when services are extended to business partners is the ability to make these lifecycle management capabilities available securely across the firewall.

## Conclusion

Ultimately, SOA governance is about maintaining control over the environment to engender the level of trust required for ongoing SOA adoption and success. While effective governance rests in how it's institutionalized in the organization, having the right technology framework makes it easier — and in some situations is the only sustainable way — to enforce policies and controls. This framework should include mechanisms for defining policies and service contracts and enforcing them through the service lifecycle through workflows, intermediaries, and other automated mechanisms.

By establishing the right balance between strategy, organizational practices, and supporting technologies, companies will be able to turn the concept of SOA governance into a practical reality. ■

---

### About the Author

Gary So is vice president, Office of the Chief Technology Officer, at webMethods, Inc, where he is responsible for advancing the company's status as a recognized industry thought leader. Gary has over 10 years experience in the integration field, serving previously as a system architect in corporate IT and as a director of professional services at Active Software, Inc., before joining webMethods in 2000. Gary has a masters degree in computer engineering from the University of Toronto.

# The ESB in Your SOA



Why the ESB market  
is skyrocketing

WRITTEN BY JAMES PASLEY

➤ Service Oriented Architecture (SOA) projects have evolved. A couple of years ago it would have been sufficient to demonstrate connectivity between systems that were previously isolated. Nowadays these connections must provide guarantees of reliability, security, and performance. Delivering on such requirements presents a number of challenges.

**N**ot all of the challenges faced during an SOA project are technical. However, it's the technical requirements that are best understood and most often explicitly stated as requirements. Other issues relate to the changes required in the way IT operations are organized in a company. While it's often the technical challenges that are focused on in the early projects, many organizations are discovering that this approach doesn't make for a successful SOA deployment.

The use of an Enterprise Service Bus (ESB) to implement an SOA addresses many of the technical challenges encountered by such projects, allowing organizations to concentrate on the other chal-

lenges that they'll face. A recent report by Gartner Dataquest showed that the Enterprise ESB market is the fastest growing application integration middleware market segment – with growth rates exceeding 100% year-over-year. So why are so many enterprises turning to ESBs to deliver their SOA projects?

This article discusses some of the technical issues solved by the ESB. It then goes on to look at the other SOA challenges relating to organizational changes that are seldom acknowledged at the start of an SOA project and often deserves the most attention.

## **The Problems Addressed by the ESB Connectivity**

The systems in any large organization will use a variety of transport layers. For example, there will be at least one message queue system. FTP, e-mail, files dropped into directories, and of course HTTP may also be used. An ESB can connect to all of these transports in a consistent manner, isolating the developer from the specific issues associated with each one. Appropriate use of an ESB can make the selection of a transport a deployment time issue rather than a development one.

## **Data Transformation**

Systems will invariably use different message formats. SOA must first standardize on the use of XML. The goal then is to use a common vocabulary of XML elements across most messages. Legacy systems are unlikely to use XML, so each format will have to be transformed to and from XML. The ESB provides tools and runtime components to handle this data transformation.

An initial SOA project may struggle to define a set of common XML elements. This will result in the need to perform additional XML-to-XML transformation on the ESB. In some cases, this may seem like surplus overhead. However, it's a necessary step on the way to achieving the goal of common message formats.

## **Reliability**

The reliability and traceability of messages flowing through the ESB is now a fundamental requirement of most SOA deployments. This means that the ESB should be based on a reliable message layer – typically one of the JMS implementations. Alternatively, the ESB can provide reliability on top of unreliable transports (typically HTTP) by using protocols such as WS-RM.

A reliable transport isolates the application from communication errors that occur at runtime. This reduces the error recovery code required in the application. It also eliminates any application code that attempts to redeliver failed messages since this is done automatically at the transport layer. Failure to adopt a reliable transport can result in the logic to catch faults and perform retries being implemented in the application layer. This is not a wise use of developer time.

## **Security**

Many ESBs provide security by using protocols such as HTTPS. However, using these protocols tie you to a single transport. Using WS-Security provides a transport-independent security mechanism. WS-Security can be added at the message-processing layer without needing modifications to the applications developed. This lets security requirements be considered relatively independently of the functional requirements for a project.

In SOA, there's increasing emphasis on the use of policies. Features that would have previously been developed as part of the application are being separated out and provided by the ESB. Secu-

rity and reliability are two areas where a policy-driven approach is being used. The ESB is configured using a policy document to both provide and enforce the level of security required.

## **Asynchronous**

Systems should be designed from the start to communicate using asynchronous messaging. A reliable asynchronous messaging system can isolate systems from faults relating to the temporary unavailability of a single service. Asynchronous messaging also provides a way to handle an increase in traffic more gracefully than synchronous messaging. The use of asynchronous messaging introduces the need to address issues such as message correlation. The ESB performs this task by using protocols such as WS-Addressing or by correlating features of languages such as WS-BPEL.

## **Orchestration**

A fundamental aspect of an SOA is the need to provide services that perform specific tasks as defined by business need rather than the technology. Unfortunately, there's seldom a one-to-one relationship between these. After exposing a number of legacy systems on the ESB, it will be necessary to orchestrate these services into a new service that reflects the business task. WS-BPEL provides a powerful way to achieve this level of orchestration. BPEL can easily deal with the correlation necessary to combine multiple asynchronous message exchanges into a single service. It also provides a reliable environment for process execution, thus ensuring that the reliability offered at the transport layer isn't lost due to problems in the application layer.

## **The Issues Enterprises Need To Address**

The initial reaction of any skilled development team would be to say that it can satisfy any of the requirements above without having to use an ESB. That may be true; however, it may be a case of reinventing the wheel and while a development team is focused on that, it may not be addressing the real issues. This section looks at some of the non-technical issues that can cause problems during the first SOA projects.

## **Training Existing Staff**

Seldom is an SOA built on a green field site. The need to reuse existing assets inevitably means that most real-world SOA deployments include a number of legacy applications. While SOA promises simplicity, integrating a number of legacy systems into an SOA can prove to be a complex task. This requires expertise in both SOA and the legacy systems. In short, this means that most of an SOA will be built by existing IT staff. Therefore, they will need to be trained in the use of the Web Services technologies required by an SOA.

## **Driving the Adoption of Shared Message Formats**

As part of adopting an SOA, many organizations will need someone who is responsible for maintaining XML Schemas. SOAs are deployed incrementally. Many initial projects struggle to produce data structures, which are reusable. Reuse of XML message formats is an important milestone in the adoption of SOA. Failure to adopt common data structures will result in the need for transformations every time a service is reused. Many standard XML Schemas exist, however it's not always possible to find an appropriate standard, so each organization will need to create XML Schemas for their own use.

# The value of an ESB is in its ability to free up your IT resources to concentrate on the higher-level issues associated with rolling out a SOA

Where two or more teams are involved in an integration project, there's an additional upfront cost involved in adopting a common set of XML Schemas. Often, the greatest difficulty is in coordinating changes to these schemas in a timely fashion. Once these XML Schemas are in use, the ability to modify them in a backwards-compatible way becomes important. Designing these XML Schemas requires a good understanding of XML Schema itself and also the ability to balance the requirements of several teams.

Reuse of entire message formats is unlikely to be possible during initial projects, so the goal should be to create a number of element definitions that can be reused in a variety of message formats. This will ease integration and cut down on the need for data transformations.

Issues such as versioning should be planned from the start. At a basic level, this can take the form of ensuring that each message on the ESB contains version information. The selection and implementation of an appropriate versioning strategy is another responsibility of the person who maintains the XML Schemas.

## Moving the Boundaries of Division of Labor

Any integration project involving a number of development teams will result in some level of negotiation on who does the work. Traditional division of labor can work strongly against the goals of an SOA. A successful SOA requires the creation of services that will be reused. In a traditional integration project, there's always the temptation to do the minimum amount of work to make the service accessible. This pushes some effort onto the client; perhaps an original text based message format will be used. SOA moves this work from the client to the service. The service must be created in such a way that's as convenient as possible for clients to access. Organizational changes may be required to ensure that the extra effort involved is acknowledged and rewarded.

## Coordinating Team Activity

The task of coordinating multiple development teams shouldn't be underestimated, especially when they're working together for the first time. The touch points between the teams such as transports, protocols, and message formats are often areas that cause difficulties. Changes to the various touch points have to be planned and coordinated to prevent a continual dripfeed of modifications into the system.

## Testing

Testing in an SOA project has to be approached in a very deliberate manner. Integration projects involving a number of applications are very sensitive to changes in any one of those applications. For example, late delivery of an application used as part of an orchestration can delay the start of testing for the overall project. Testing strategies need to plan for this eventuality. Testing needs to start early and be done in parallel with development. Stub versions of each service should be developed to facilitate isolated testing. This will limit the probability that delays relating to a single service will delay the overall project.

## Conclusion

Many of the technical issues solved by the ESB may appear at first glance to be straightforward. However, the cost of delivering quality of service features such as reliability, security, and performance shouldn't be underestimated. Don't fall into the trap of attempting to implement these from the ground up. As many others are now doing, use an ESB to deliver these features. The true value of an ESB is in its ability to free up your IT resources to concentrate on the higher-level issues associated with rolling out a SOA. ■

## About the Author

James Pasley is CTO of Cape Clear Software, where he is responsible for ensuring that the company's Enterprise Service Bus (ESB) technology supports the evolving needs of its global customer base. Pasley, who joined Cape Clear in 2001 as a lead developer for Cape Studio, is a technology visionary boasting several years of experience implementing messaging and middleware standards. Prior to joining Cape Clear, Pasley worked for Siemens Nixdorf developing secure X.400 messaging and public key infrastructure (PKI) solutions for a range of military and civilian products. He also served as a primary developer of the CORBA conformance test suite for the Object Management Group (OMG). He holds a computer science degree from Trinity College, Dublin.

## SOA WSJ Advertiser Index

Advertising Partner	Web Site URL	Phone #	Page
ACTIVE ENDPOINTS	ACTIVEBP.ORG/SOA		4
AJAXWORLD	WWW.AJAXWORLDEXPO.COM	201-802-3022	41
AJAXWORLD BOOTCAMP	WWW.AJAXBOOTCAMP.COM	201-802-3022	31
ALTOVA	WWW.ALTOVA.COM	203-929-9400	2,27
CROSSCHECK NETWORKS	WWW.CROSSCHECKNET.COM	888 276 7725	23
FORUM SYSTEMS	WWW.FORUMSYSTEMS.COM	801-313-4400	51
IBM	WWW.IBM.COM/TAKEBACKCONTROL/PROCESS		6-7
JACKBE	WWW.JACKBE.COM	240-744-7620	11
PARASOFT	WWW.PARASOFT.COM/WSJMAGAZINE	888-305-0041 (X-3501)	52
SOFTWARE AG	WWW.SOFTWAREAG.COM/TAMINO		19
SYS-CON EMAIL NEWSLETTERS	WWW.SYS-CON.COM	201-802-3022	37

General Conditions: The Publisher reserves the right to refuse any advertising not meeting the standards that are set to protect the high editorial quality of. All advertising is subject to approval by the Publisher. The Publisher assumes no liability for any costs or damages incurred if for any reason the Publisher fails to publish an advertisement. In no event shall the Publisher be liable for any costs or damages in excess of the cost of the advertisement as a result of a mistake in the advertisement or for any other reason. The Advertiser is fully responsible for all financial liability and terms of the contract executed by the agents or agencies who are acting on behalf of the Advertiser. Conditions set in this document (except the rates) are subject to change by the Publisher without notice. No conditions other than those set forth in this "General Conditions Document" shall be binding upon the Publisher. Advertisers (and their agencies) are fully responsible for the content of their advertisements printed in SOA Web Services Journal. Advertisements are to be printed at the discretion of the Publisher. This discretion includes the positioning of the advertisement, except for "preferred positions" described in the rate table. Cancellations and changes to advertisements must be made in writing before the closing date. "Publisher" in this "General Conditions Document" refers to SYS-CON Publications, Inc. This index is provided as an additional service to our readers. The publisher does not assume any liability for errors or omissions.



**tamino**  
XML Server

Dear IBM,  
Congratulations on  
your first native  
XML database.

We know this business  
inside out\*, so:  
Welcome to the club!

Kind regards,  
Software AG

\*For seven years we have fulfilled customer needs with  
our powerful high-performance Tamino XML Server.  
**Welcome to the club, IBM.**

>> [www.softwareag.com/Tamino](http://www.softwareag.com/Tamino)

 **SOFTWARE AG**

# Web Services, WS-\* Specifications, and Interoperability

Achieving interoperability is  
neither simple nor straightforward

WRITTEN BY SANJAY NARANG

➤ Interoperability is an important factor in the success of solutions that are based on Web Services and Service Oriented Architecture (SOA), along with other key factors such as contracts, loose coupling, and reuse. Interoperability is generally accomplished by developing your Web Services using the well-established guidelines for implementing Web Services and by following industry standards such as XML, WSDL, SOAP, and UDDI. However, just following Web Services standards and guidelines during the development phase of a project isn't sufficient to achieve interoperability.

**T**he different products used for development also have to comply with many requirements such as the need to have similar implementations (data types, formats, and schemas) of the standards that you want to use. As different products are provided by different vendors, developed by several sets of people, and employ various types of underlying technologies, achieving a

common understanding often becomes very difficult, which makes the products likely to be non-interoperable with each other.

Over the last few years, the basic Web Services standards like XML, WSDL, and SOAP have matured a lot and WS-I has released a Basic Profile (described later) that contains implementation guidelines for basic Web Services standards. Today, most vendors provide products that comply with the Basic Profile and support the standards included in the profile. With the wide adoption of the Basic Profile, software vendors have been able to make their products interoperable to a great extent.

As the Web Services industry evolves, it embraces new specifications like WS-Security, WS-ReliableMessaging (WS-RM), and WS-AtomicTransactions (WS-AT) to provide advanced functionalities such as security, reliability, and transactions that are not provided by the basic specifications. These specifications are generally referred to as the WS-\* (pronounced WS-Star) specifications. As they are relatively new and have not been so widely agreed on by the industry, achieving interoperability between Web Services that use WS-\* specifications is much more difficult and the WS-\* specifications may not even be supported in many products.

This article provides a set of guidelines and best practices that you can follow to accomplish interoperability when developing web services that make use of the WS-\* specifications across products provided by different vendors. It also provides insight into the Web Services specifications situation that contains a large number of WS-\* specifications that are being developed by different groups.

*Author's Note:*

1. This article uses the term “product” as a common term to refer to platforms, technologies, or tools provided by software vendors for developing Web Services.
2. It uses examples of interoperability between J2EE- and .NET- based Web Services. However, the given guidelines and best practices can be applied to other platforms too.
3. This article assumes that the reader is aware of basic Web Services concepts and knows the different steps involved in developing Web Services.

## Basic Web Services Interoperability

Achieving interoperability for scenarios involving only basic standards is relatively easy if you follow the guidelines set by the Basic Profile (BP) 1.0 or 1.1 of the Web Services Interoperability Organization (WS-I). The Basic Profile consists of implementing guidelines recommending how a set of core Web Services specifications should be used together to develop interoperable Web Services. The guidelines address technologies that cover four core areas: Messaging, Description, Discovery, and Security. BP1.0 covers the following core Web Services specifications and provides constraints and clarifications to these base specifications, along with conventions about how to use them together:

- SOAP 1.1
- WSDL 1.1
- UDDI 2.0
- XML 1.0 (Second Edition)
- XML Schema Part 1: Structures
- XML Schema Part 2: Data types
- RFC2246: The Transport Layer Security Protocol Version 1.0
- RFC2459: Internet X.509 Public Key Infrastructure Certificate and CRL Profile
- RFC2616: HyperText Transfer Protocol 1.1
- RFC2818: HTTP over TLS Transport Layer Security
- RFC2965: HTTP State Management Mechanism
- The Secure Sockets Layer Protocol Version 3.0

You can see detailed information about the Basic Profile at its home page: <http://www.ws-i.org/deliverables/workinggroup.aspx?wg=basiprofile>.

If you can ensure that all the products being used in your solution are compliant with the Basic Profile, you are very close to achieving the interoperability between those products. Additionally, if you build your Web Services by following the Basic Profile guidelines and by using the specifications listed above, you can achieve complete interoperability. However, the guidelines for using basic Web Services specifications aren't within the scope of this article. To find out more on the topic, you can either look at resources listed in the section “Web Services Interoperability Resources” or search for the keyword “Web Services interoperability” on any major search engine. You'll find numerous articles, guidelines, and samples.

## Land in the WS-\* Space

The basic Web Services specifications such as XML, SOAP, and UDDI have been around for a long time now and have been ratified by one standards body or another (described later in this section) as standards. These specifications have also found wide adoption in the industry and in leading Web Services products. However, the case is not the same for advanced Web Services specifications

or WS-\* specifications. Figure 1 below graphically represents how different categories of specifications fit in the context of a Web Services framework.

Note that Figure 1 provides high-level categorization of Web Services specifications; it shouldn't be considered a underlying architecture for Web Services implementations. The basic specification provides the core functionalities for Web Services such as messaging, discovery, and description, whereas WS-\* specifications provide advanced functionalities such as security, reliability, and transactions. A specification is generally given the status of a standard when it's approved by a standards body. There are two main standard bodies dealing with Web Services standards:

1. **W3C:** The World Wide Web Consortium (<http://www.w3.org/>)
2. **OASIS:** The Organization for the Advancement of Structured Information Standards (<http://www.oasis-open.org>)

W3C was founded mainly to focus on Web-based protocols and standards such as HTML, HTTP, and others. It also plays a core role in standardizing the basic specifications for Web Services including SOAP, XML, and WSDL. It's expanded beyond that and has been involved in overseeing higher-level ones such as WS-Choreography Description Language.

On the other hand, OASIS focuses on developing and adopting e-business and higher-level Web Services specifications such as ebXML, UDDI, and many of WS-\* specifications including WS-Security, WS-Reliability, Web Services Business Process Execution Language (WS-BPEL), etc.

Besides to these two major standard bodies, there are two other groups involved in Web Services-related standardizations:

1. The Web Services Interoperability (WS-I) organization (<http://www.ws-i.org/>) was founded by Microsoft, IBM, and other vendors primarily to promote interoperability across platforms. It focuses on developing profiles of Web Services standards that enable interoperability. It also provides usage scenarios, sample applications, and testing tools to help you develop applications conforming to the profiles. Its profile, BP1.0, mentioned in the section “Basic Web Services Interoperability” above, has gained wide acceptance in the industry.
2. The Liberty Alliance (<http://www.projectliberty.org/>) was co-founded by Sun with the mission to develop Web Services specifications for identity management using the Security Assertion Markup Language (SAML). It focuses exclusively on identity management and security issues.



Figure 1: Relationships between WS-\* specifications

### **Stages of a Specification Before Becoming a Standard**

This section describes the process followed by W3C and OASIS to adopt a specification as a standard. The exact process followed by the two standards bodies might be different, but on a high level, they follow the process described here.

Generally, some vendors (such as Microsoft and IBM) get together and create a draft version of a specification that they want widely adopted. They publish that draft specification to get feedback and support from the industry. The authoring vendors make changes to the specification per the feedback and, if they choose to, submit an agreed version to one of the standards bodies. The standards body then enters its process for deciding whether to form a working group (WG) in the case of W3C or a technical committee (TC) in the case of OASIS. If it does form the WG or TC, it calls for participation from industry members to work on the specification. The WG or TC works on enhancing the original submission. At this stage, the specification is called a “working draft.” When they are done with enhancements, the WG or TC members publish the specification for public review and take inputs. If the specification gets approved in the public review, the TC recommends to the standard body that it ratify the specification as a standard. In the case of the W3C, the process for promoting a specification to the status of standard is more rigorous.

This process and the availability of numerous Web Services specifications complicate the building of Web Services using WS-\* specifications. The complications are described in the next two sections.

### **Multiple Statuses**

The specifications listed in Figure 1 are at different “statuses” or “maturity levels” of the standards process. For example, WS-Security (SOAP Message Security v1.1) is an OASIS standard and WS-Addressing v1.0 is a W3C Candidate Recommendation (almost equivalent to a standard). WS-Policy is only a formal submission to W3C and WS-AtomicTransaction (WS-AT) isn't even submitted to a standard body; it's a joint publication by BEA, IBM, Microsoft, and few other vendors.

*Author's Note: The status of the various specifications mentioned in this document is their status at the time of writing. But, their status keeps changing and may be different when you read this.*

### **Competing Specifications**

There are some areas where there's more than one specification to address the same functionality; the multiple specifications are supported by different consortia of vendors.

For example, the WS-Choreography Description Language v1.0 is a candidate recommendation (close to becoming a final recommendation or a standard) at W3C for Web Services choreography and was submitted initially by Novell, Oracle, and others. On the other hand, Web Services Business Process Execution Language (WS-BPEL) v2.0 is a committee draft at OASIS for very similar requirements (i.e., Web Services orchestration); it's supported by a mostly different set of vendors like BEA, HP, IBM, and Microsoft.

Similarly, WS-Reliability v1.1 is an OASIS standard, initially submitted by Sun, Oracle, and others. But another group of vendors made up of BEA, IBM, Microsoft, and TIBCO has published a similar but different specification called WS-ReliableMessaging. In some instances, you'll also find a vendor supporting competing specifications.

Because of these complications, achieving interoperability using WS-\* specifications becomes much more difficult compared to using basic Web Services specifications. The following section describes some guidelines that should help you increase the interoperability success rate of your implementations and reduce the time spent in debugging and investigations.

### **Guidelines for Interoperating Using WS-\***

You need to look from a number of perspectives to ensure the interoperability between Web Services (using WS-\* specifications) that are built on different products. This section offers insight into these perspectives and provides examples wherever applicable.

#### **Support for the Same Specifications**

As mentioned above, there could be more than one specification for the same functionality such as reliability or orchestration. Before you start, ensure that the products on which the Web Services are developed support the same specification.

For example, if you want reliable delivery of messages between Web Services developed on two different products, both products should support the same specification, that is, both support WS-Reliability (WS-R) or both support WS-Reliable Messaging (WS-RM). Otherwise, you may not be able to interoperate between the two products.

Another example is about identity federation, for which there are two different specifications available: Liberty Alliance's Identity Framework ID-FF and WS-Federation. In both specifications, federation is implemented through a Security Token Service (STS) that provides security tokens to the requesting clients. But the two specifications use different kinds of security tokens; Liberty Alliance uses extended SAML assertions, whereas WS-Federation uses WS-Security's profiles of X509 and Kerberos. When two parties (or organizations or security zones) plan to implement identity federation between them, they must ensure that the common security token is supported by the two STSes; otherwise, they won't be able to implement the federation.

#### **Versions of the Specifications**

Web Services specifications are evolving and new versions keep moving through both the draft and final stage. But the products that implement these specifications don't change at the same pace. Because of that, you find products from different vendors implementing different versions of the specifications and the different versions of the same specification may not interoperate with each other. In such cases, check if the product that supports the newer version of the specification supports the older version too. Generally, products support the older version to maintain backward compatibility. You should configure a product to use an older version of the specification, if necessary, so that both products use the same version.

For example, BEA WebLogic Server (WLS) 9.0 only supports SOAP 1.1 whereas Microsoft Windows Communication Foundation (WCF) February 2006 Community Technology Preview (CTP) supports both SOAP 1.2 and SOAP 1.1. WCF provides out-of-the-box configuration settings for different kinds of bindings. All of the bindings except basicHTTPBinding are configured to use SOAP 1.2 by default. If your WCF Web Services use any of these bindings, they won't be able to communicate with WLS 9.0 clients and vice versa. To enable communication, you need to configure the WCF Web Services with a binding that uses SOAP 1.1. You can do that by creating a custom binding in WCF.



## *The Art of Web Services Testing*



### Build Your Applications on 4 Pillars of SOA Testing

- I. Automated Functional Regression
- II. Performance Profiles
- III. Interoperability Compliance
- IV. Vulnerability Risk Posture

Download SOAPSonar Enterprise Edition  
<http://www.crosschecknet.com>

# WS-\* specifications haven't matured yet so achieving interoperability using them is relatively difficult

You also have to ensure that versions of the other specifications that are being used by the specification you're implementing are the same. For example, WS-RM makes use of WS-Addressing. By default, the WCF Feb 2006 CTP uses the version of WS-Addressing given below:

```
wsa10=http://www.w3.org/2005/08/addressing
```

Whereas the WLS 9.0 supports only the older version of WS-Addressing as given below:

```
wsa=http://schemas.xmlsoap.org/ws/2004/08/addressing
```

In this case also, you have to configure the WCF Web Service with a custom binding that uses an older version of WS-Addressing. If you don't want to create a custom binding, you can manually change the generated WSDL of the WCF service and use the changed WSDL for generating a proxy that the WLS client uses to call the WCF service. This will work only when the two versions are compatible with each other.

Sometimes, vendors provide service packs or refresh packs to overcome discrepancies in the released version of the product. These packs might bring in changes in the supported specifications. These changes might break the existing interoperability or enable interoperability where it was absent earlier. For example, IBM WebSphere Application Server (WAS) 6.0 supports the element `<CoordinationContextType>` for WS-AT, but when the refresh pack 6.0.2.9 is applied over it, the supported element changes to `<CoordinationContext>`. You should analyze all such changes in the refresh packs before deciding to apply these packs.

## Schemas, Namespaces, and WSDLs

In some instances, the two products might use the same version number of a specification, but the namespaces for the specification might refer to different schemas. For example, both WCF Feb 2006 CTP and WLS 9.0 use the February 2005 version of the WS-RM Policy specifications but refer to different namespaces. The WCF Feb 2006 CTP generates WSDLs that refer to the following namespace for WS-RM Policy Assertions:

```
xmlns:wsrm=http://schemas.xmlsoap.org/ws/2005/02/rm/policy
```

Whereas the WLS 9.0 generates WSDLs that refer to this namespace:

```
xmlns:wsrm=http://schemas.xmlsoap.org/ws/2005/02/rm
```

The namespace above is actually for WS-ReliableMessaging specification and not for WS-RM Policy specification (which is different from WS-ReliableMessaging). But for some reason, WLS 9.0 uses this namespace to refer to RM Policy assertions.

In such cases, you can change the WSDL of a service manually to refer to a namespace that is understandable by the client. Subsequently, when you generate the client proxy from that changed

WSDL, the client proxy would refer to the changed namespace. Hence the clients can communicate with the service successfully.

There's also not much standardization on the location of WS-Policy references in WSDLs. For example, WLS 9.0 adds the policy expression reference in the `<operation>` element, whereas WCF adds the policy file reference in the `<binding>` element. In such cases, you have to change the WSDL of a Web Service according to the client requirements.

*Author's Note: In WebLogic Server 9.2, the namespace has been corrected and the WSDLs generated refer to the WS-RM Policy namespace for policy assertions.*

## Specifications & Actual Implementations

Although products may claim to support a particular specification, they may not support all the features in the specification that aren't mandatory. For example, WS-RM specifies four kinds of delivery assurances: `AtMostOnce`, `AtLeastOnce`, `ExactlyOnce`, and `InOrder`. However, WCF Feb 2006 CTP supports only two options: `ExactlyOnce` and `InOrder`. Hence, you need to make sure that both the products that you're trying to interoperate support the option you're choosing for implementation.

## Special Cases

You can come across many issues, which can't be categorized in a particular type of issue like the ones listed above. Such issues might be because of some problem in the configurations of a service, or client, or might also be because of some bug in a product. For example, there's a special problem between WLS 9.0 and WCF Feb 2006 CTP.

When a client running in WLS 9.0 calls an operation defined in a WCF Feb 2006 CTP service using WS-RM, the WLS requires at least one "two-way" (Request-Response or Solicit-Response) operation to be defined in the WCF service. Otherwise, WLS can't use WS-RM for communicating with WCF. The "two-way" operation is required even if the WLS client is only calling "one-way" operations in the WCF service.

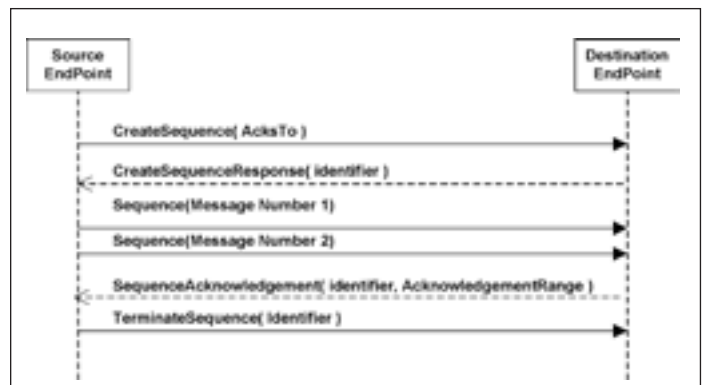


Figure 2: A typical Reliable Message exchange pattern

To understand this special problem, you have to know the kinds of messages that get exchanged in a typical WS-RM-based communication, which is shown in Figure 2:

According to the WS-RM specification, a source end-point sends a message with a <CreateSequence> element to a destination end-point to start a reliable session. The destination end-point responds with a message having a <CreateSequenceResponse> element. When the destination end-point is deployed using WCF CTP, WCF CTP sends the message with a <CreateSequenceResponse> element to the location that's provided in the <ReplyTo> element (defined in the WS-Addressing specification), which is present in the header of the CreateSequence message. Whereas, when WLS 9.0 hosts a destination end-point, it sends the response message to the location provided in the <AcksTo> element (defined in the WS-RM specification), which is present inside the <CreateSequence> element of the CreateSequence message.

When a WLS 9.0 client sends WS-RM-based messages to a WCF CTP service and there's no two-way operation defined in the service, the <ReplyTo> element in the CreateSequence message sent by WLS client contains the URI for an anonymous end-point (a well-known URI defined in WS-Addressing) as given below:

```
<wsa:ReplyTo xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">  
<wsa:Address>http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous</wsa:  
Address>  
</wsa:ReplyTo>
```

The WCF CTP service uses the address provided in the <ReplyTo> element to send the response message with the <CreateSequenceResponse> element. Because the anonymous end-point address provided here is used to indicate no response processing, no CreateSequenceResponse reaches the client and hence a reliable session can't be initiated. However, when a WCF CTP service has at least one two-way operation defined, WLS 9.0 fills the <ReplyTo> element of the CreateResponse message with the valid URI and hence a reliable session can be initiated successfully.

Such problems are difficult to identify and can be found only through careful troubleshooting and reading the information available in the product documentation and on the Internet.

### Development Best Practices

The WS-\* based development is a complex activity because of all of these issues. You can follow the best practices given below to reduce the complexity to some extent.

1. Start implementing using only the basic Web Services specification. If your Web Services aren't interoperating at the basic level, implementing WS-\* specifications will only make the issues harder to identify.
2. For basic Web Services specifications, always use WS-I's Basic Profile 1.0 or 1.1 guidelines. Most of vendors provide Basic Profile-compliant products and have tested those products comprehensively for interoperability. Hence, by following the Basic Profile, you can easily avoid the issues that WS-I has already addressed in the Basic Profile. You can verify if your Web Services are following the Basic Profile by testing your WSDLs in WSDL testing tools. There are many free tools, including those provided by WS-I, as well as commercial ones, available to test WSDL.
3. Once your Web Services are interoperating with basic specifications and BP1.x, start implementing the required WS-\* specifications one by one. If you implement all the required specifications

(e.g. WS-Security, WS-RM, and WS-AT) at the same time and come across an issue, it would be difficult to identify the culprit.

4. Although this is a very common best practice that can be applied to any scenario, I feel it's important to mention here. Search on the Internet for other findings and samples. As WS-\* specifications haven't matured, the documentation for the available products is either insufficient or incorrect in some areas. In such a situation, it becomes important to learn from others' experiences. Look at the blogs of a vendor's key employees involved in interoperability, post into forums like ws-builders or soapbuilders (described later), or just do a simple Web search. Many times, you'll find someone who has already encountered the issue you're facing and he or she can either help you or can tell you that the problem can't be addressed in the current version of a product.

### Troubleshooting Best Practices

Most of the time you won't get successful interoperability in the first try. You can follow the troubleshooting best practices given below to identify the root cause of the issues you come across:

1. When a problem occurs, identify the step where the problem is happening. The exact process followed to make a call from a Web Service or a client to another Web Service may be different from one product to the other, but the high-level steps remain the same:
  - a. Get the WSDL of the target Web Service that the client needs to call
  - b. Generate a "stub" or "proxy" using various tools available in Web Services toolkits or client development environments (e.g., Add Service Reference in Visual Studio 2005).
  - c. Use the generated proxy in the client code to send messages (or make calls) to the target Web Service.

The problem could exist in any of these steps. There could be instances where the WSDL of a target service isn't compatible with your client environment. The examples of such scenarios and their solutions have already been explained.

When the problem exists in the proxy generation, you can look into the output of the tool used for generating the proxy to find out the exact error(s). If the error description doesn't provide much help, you can try other best practices described in the following bullets.

1. When the problem happens in the messages exchanged, you need to analyze the actual SOAP message exchanged to find the problem and fix it. Many vendors provide tracing tools with their products that generate traces of all messages exchanged between two communicating Web Services. For example, Microsoft provides a TraceViewer tool with WCF that helps you easily view, group, and filter traced messages. You can also use the tools that provide traces for TCP packets such as the TCP Monitor provided with Apache AXIS. To analyze the messages, you should have adequate knowledge about the specification you're implementing. You should know the different kinds of SOAP messages, their structure as well as sequence, that get exchanged between the two interacting components implementing that specification. For example, to diagnose an issue in a WS-RM implementation, you have to know the different types of messages that are exchanged during a reliable session: CreateSequence, CreateSequenceResponse, Sequence, SequenceAcknowledgement, etc. as shown in Figure 2. In this step too, if the message analysis doesn't provide much help, you can try other best practices described in the following bullets.

2. Try implementing a WS-\* specification between two Web Services developed on the same product. Do this on both the products you're trying to interoperate. If you're successful then take the same configurations to implement the specification between the Web Services developed on different products. By doing this you can identify if the issue lies in a particular product or in the general configuration of that specification. When both of these options are ruled out, you can look for other interoperability-related issues.
3. When you can't identify the reason for an issue that exists while implementing a WS-\* specification between your application's Web Services, try implementing the specification between simple "hello world" types of Web Services. This will help you separate business logic-related problems from the specification's configuration-related problems. If that succeeds then apply the same configurations to your application's Web Services.

## Web Services Interoperability Resources

There's a huge amount of information available on Web Services interoperability and WS-\* specifications. I provide some selected resources and brief descriptions here that I found useful.

### Community Groups

- **SOAP Builder Yahoo Group:**  
<http://groups.yahoo.com/group/soapbuilders/>  
It's a community group to discuss cross-platform implementations and interoperability issues primarily about core Web Services specifications. You'll find messages from employees of key vendors such as BEA, IBM, and Microsoft who are involved in building Web Services and SOAP-based products. You'll find different types of messages: general discussions, announcements of new events (such as Plug Fests), product versions and features, and clarifications and solutions to problems that product consumers have.
- **WS-Builders Yahoo Group:**  
<http://groups.yahoo.com/group/ws-builders/>  
It's another community group similar to the SOAP Builders group, but focused more on WS-\* specifications.

### SOAP Interoperability Labs & Plug Fests

As interoperability gains in importance, industry vendors are getting together to make their products interoperate. Interoperability labs are examples of industry's effort towards interoperability. In these events, many vendors get together to test their products (application servers, Web Services frameworks, SOAP toolkits, etc.) with the products of other vendors.

Microsoft hosts similar events called WCF Plug Fests. At these events Microsoft invites all industry vendors to its campus for a three- or four-day workshop, where vendors test their products against WCF.

You can get very useful information from such events that may include interoperability scenarios descriptions, scenario testing scripts, live end-points, and sample WSDLs. Here are links to get information about the past events:

- **WCF Plug Fest:**  
<http://msoapinterop.org/ilab/wcfinteroplab.htm>
- **SOAPBuilders Interoperability Lab Round 2 Home Page:**  
<http://www.whitemesa.com/interop.htm>
- **SOAPBuilders Interoperability Lab Round 1 Home Page:**  
<http://www.xmethods.net/ilab/>

To find out about future events, check the announcements at community groups listed above.

### Vendor's Home Pages on Web Services Specifications & Interoperability

Here's a list of a few vendor Web pages that are dedicated to Web Service specifications and interoperability

- **Microsoft's Web Services specifications page:** Provides a comprehensive list of all the Web Services specifications that Microsoft supports in a logically structured Web Services protocol stack. Here you can download actual specification documents, check their status, and find white papers, and other resources about the specifications.  
<http://msdn.microsoft.com/webservices/webservices/understanding/specs/default.aspx>
- **MSDN page on Web Services interoperability:**  
<http://msdn.microsoft.com/webservices/webservices/building/interop/default.aspx>
- **IBM's Web Services specifications page:** Like Microsoft's page, IBM's page also provides a comprehensive list of all the Web Service specifications that IBM supports in a clearly categorized structure. Here too you can download actual specification documents, check their status, and find white papers, and other resources.  
<http://www-128.ibm.com/developerworks/webservices/standards/>
- **BEA's Web Services standards page:**  
<http://dev2dev.bea.com/webservices/standards.html>
- **Sun's Web Services Interoperability Technology (WSIT) for Java and .Net page:**  
<http://java.sun.com/webservices/interop/index.jsp>

## Summary

Web Services have evolved over the past few years and their support is now considered part of the core platform of most SOA-based solutions. With this evolution, industry is coming up with new specifications to support advance requirements like security, reliability, and transactions. These specifications are commonly referred as WS-\* specifications and are relatively new compared to basic Web Services specification such as WSDL, and UDDI. The products providing support for the WS-\* specifications haven't matured adequately. As a result, achieving interoperability using them is relatively difficult. However, if you select the products after a detailed analysis about the specifications they support and you follow the guidelines listed in this article, you can achieve an interoperable implementation in much less time. ■

### About the Author

Sanjay Narang is a senior technology consultant at the Global Delivery India Center (GDIC) of Hewlett-Packard. He is based in Bangalore, India and has around eight years of IT experience. Sanjay has done a Post-Graduate Diploma in IT (PGDIT) and has MCSD for .NET Framework and SCJP certifications. He has been involved in designing and developing solutions based on Microsoft technologies for a long time and has worked on various solutions and products related to software process automation and software quality standards such as SEI CMMi. He is currently working on SOA-related projects around Microsoft technologies and their interoperability with other technologies and has published papers about IPv6, VSTS, SOA, and Web Services. He writes extensively about VSTS in his blog: <http://sanjaynarang.wordpress.com/>. [sanjay.narang@yahoo.com](mailto:sanjay.narang@yahoo.com).

# Evade constraining query tools

Move up to DatabaseSpy 2007, and manage  
all your databases from one elegant interface.

## BRAND NEW DATA MANAGEMENT TOOL!

- Connects to all major databases
- Project Manager organizes connections, and Database Browser clearly presents tables, views, and data
- SQL Editor facilitates query writing with code completion, syntax coloring, drag & drop construction, and more
- Design Editor enables graphical design and visualization of database structures

Altova DatabaseSpy 2007 is the unique multi-database query and design tool from the creators of XMLSpy. A real steal, DatabaseSpy liberates data management, delivering advanced, coherent capabilities at a fraction of the cost of single-database solutions. Write accurate queries with confidence and get clear, easily negotiable results.

Create and edit database structures visually by dropping tables on the design pane and dragging relationship links between them. Even duplicate existing structures in diverse database types. Do data differently! **Download DatabaseSpy™ 2007 today: [www.altova.com](http://www.altova.com)**

Join Altova at  
Microsoft Connections,  
Las Vegas  
**Booth #407**



# SOA Worst Practices

When the “right thing” goes wrong—a safety ladder

WRITTEN BY DAN FOODY

➤ The world would be a better place if everyone would just do the right thing. Unfortunately, doing the right thing is often easier said than done. Especially in business ideas that seem to have the most promise can actually yield the worst results.

**F**or example, applying SOA “best practices” to a business can be tricky because these practices don’t always relate to real-world situations or business models. Ideas that look good on paper are often not so good in practice. In fact, they may turn out to be “worst practices.” Learning from mistakes is a valuable exercise; however, the learning process is even better when someone else makes the mistake.

In the following article, you will learn some of the most common mistakes that IT teams make as they begin a SOA deployment process. With any luck, this insight will teach you what not to do as you begin to implement your own SOA. This way, you can be assured that your organization is on the path to success.

## Rip and Replace

Migrating an entire system to SOA in the short term is a recipe for disaster. Theoretically, it may seem like a good idea to jump right into a SOA implementation, ripping out and replacing all existing systems at once. SOA technology is new, exciting, and hugely beneficial, and it’s easy to get carried away.

However, SOA can be overwhelming when you try to wrap your mind around it all in one sitting; especially when you begin implementing a massive project that involves thousands of compo-

nents. The best way to sidestep this issue is to be selective. Start with applications that are not working – migrating or rebuilding these applications will deliver immediate benefits.

Doing too much too soon will overwhelm your IT team and lower their level of confidence in the project. SOA gives you the opportunity to use a best-of-breed approach, as well as to leverage open standards. A SOA implementation plan should cover migrating your entire system to a SOA, but remember this process can take years. Focusing on applications that aren't working properly provides an opportunity for immediate payoff while functioning applications continue to deliver value.

The strategy of every organization should be to only undertake tasks that add value to the organization as a whole. With this in mind, make sure that your SOA strategy focuses on leveraging the business processes that deliver the most benefits to the company.

## Randomly Wrapping Services

Enterprises without a business strategy to dictate what services to wrap and how they should be governed end up with security and performance problems – both inside and outside the organization. Businesses have undertaken practices known as wrapping services to permit the flow of data from one source to another by allowing a client to send complete SQL statements in a request to the accessed data, such as a database. This protocol has left the relational database susceptible to SQL injection attacks, where attackers bypass the SQL statements defined by the Web server and input their own.

The IT team at one particular company faced this problem when its Web server endeavored to use an anti-injection attack feature already present in its current security bundle. The feature was meant to detect and sanitize SQL statements that were scripted to perform only destructive operations such as “Drop” or “Delete.”

While utilizing the wrapping concept in a Web-based service approach isn't a bad idea or dangerous in itself, the implementation must be more specifically defined than in the previous example where users can make “dumb” SQL requests. Wrapping services demands a tightly coupled approach and a requestor that knows the implementation details to make the SQL call.

Overall, this approach lacks a business context and makes the database vulnerable to problems such as bad data input to SQL statements and invalid or bad SQL statements. An IT team investing time in manufacturing a Web-based wrapped SOA needs to disallow the input of “dumb” requests by setting up rules that define who has a right to request to the SQL database and in what situation that's appropriate. This requires that the user be authenticated before being granted access. That prevents the database from giving out information blindly and thwarts destructive requests without having to resort to security software measures.

When communicating directly to the database, applications and processes have too much authority. By offloading policy management and governance, rules and policies are easier to manage. The SQL injection problems can be solved by applying SOA and Web Service standards. This approach helps to achieve reuse and integration goals quickly but must be part of an overall SOA strategy or there could be problems on the application level.

## Tweaking Standards

Organizations that modify standards – for instance, an open language like SOAP – lose a lot of value. By splitting SOAP messages into more than one part, such as using an envelope and a body, applications that employ the standardized SOAP stack become

incompatible with those in the unorthodox configuration and this creates a proprietary form of messaging.

By adding these proprietary “extensions,” the system limits interoperability dramatically limiting the scope of the SOA effort and drastically increasing the cost of operation. Because of the proprietary nature of the SOAP stack, specific code must reside on both sides of the communication and all applications and software in use through the SOA must be customized. This also prevents the use of developer tools, which are designed to work with standard SOAP stacks, creating problems with versioning software that must then be updated later for an ISV's revised platform.

To combat this predicament, IT teams must design a SOA that avoids too many components, thereby adding to the load. Using a universal language and making sure that components are loosely coupled is the best way to accomplish this objective. In this way, appropriate information is able to reach its endpoint in significantly fewer steps.

## Poorly Designed Firewalls

Organizations that do not take the time to understand all the interdependencies, as well as the potential business impact, of their firewall will encounter communication problems that frustrate customers and partners. The interoperability that SOA presents can be easily stalled by the misapplication of firewall technology. It's easy when implementing an SOA to confuse the performance and UI requirements of a firewall with those of the SOA. This can lead to a restriction on the data that the SOA seeks to make available to consumers and partners.

When building an SOA be sure to consider the specific reasons – based on business and IT objectives – for its implementation. It's also important to find a way to clearly communicate this strategy to partners and customers. Finally, be aware that interoperability is a requirement for SOA, but it doesn't happen automatically. It's important to take into consideration all the interdependencies between partners and customers, as well as any other business impact, since these factors all have a bearing on your SOA.

## Reconfiguring Existing Applications

On the surface, cloning and reconfiguring existing applications may sound like a great way to reduce development costs and save on maintenance. In reality, doing so can result in numerous issues including replicating bugs, creating redundant infrastructure, and testing nightmares.

Some common misconceptions about cloning apps are that by doing so the organization will have more developers who are familiar with a larger percentage of the applications and will lead to the creation of simple architecture and make it easy to plan and compute future costs. What ends up happening is quite the opposite. For instance, if there's a bug in the core code, it will end up being replicated. If the code is altered when you clone an application and then customize it, often there's no history of the alterations in the code. Therefore, before a team can fix problems, it needs to spend time determining what caused the problem in the first place. Finally, by embedding rules or policies as code into the application rather than abstracting them, you can add to the level of redundancy and make changes even more difficult to make.

There are ways to reduce your maintenance and development costs that don't involve reconfiguring existing applications. For example, you can distill a small set of core applications from hundreds of duplication applications. You can then use those reusable

# I once heard someone give a keynote at a conference and tout the fact that his organization had 300 services and anticipated having 1,000 by the end of the year. That's not a measure of success; it's a train wreck in the making

services across applications and business processes, achieving greater operating efficiency and lower maintenance costs.

The beauty of an SOA is that it enables you to construct new business solutions faster than by cloning applications. That means you can spend more time customizing solutions for each of your customers, while leveraging your core set of services.

## Sharing WSDLs

When implementing an SOA, you need to set clear guidelines about how services can and should be reused. By not doing so, you can open your organization up to serious security, privacy, and compliance issues. In addition, you can end up missing a crucial opportunity to show the measurable value of your SOA. Sharing Web Services description languages (WSDLs) may leave your IT department wondering whether the right departments are being supported, whether the data is protected or encrypted, and what the possible ramifications of a system failure are.

There's a happy medium that would allow services to be reused and guarantee that proper safeguards are in place. When implementing a detailed process regarding the reuse of service you should:

- Identify who in your organization will be using the service.
- Implement proper security measures that ensure that your IT policy is enforced and your business rules are applied.
- Put a mechanism in place that can help alert you to when a service is being used by unauthorized users. That mechanism should also be able to help determine whether there's inappropriate use of services.
- Set up management and runtime governance technology.
- Validate that the service is running as designed and meeting the business objectives for which it was designed.

Don't fall into the trap of believing that sharing WSDLs is an innocuous act. On the contrary, uncontrolled sharing can open your services to countless individuals and put your data at risk.

## Invoking External Schema Validation

While this may sound like added protection, invoking external schema validation may actually leave the Web-based server more vulnerable to attacks. It's generally easy for a hacker to gauge the external validation by monitoring the transaction response and then set up a spoofed IP and change the location of the schema check. The hacker can then initiate transactions with a dummy schema and have it verified with a fake schema validation set up at the spoofed IP address and cause legitimate transactions to fail. While the information present in the server was protected by the schema, business can still be gravely impacted by the wrong hands.

With two simple steps, the schema developed for the SOA can be safeguarded. First, provide the schema to partners directly since

anyone attempting a hack would not have access to direct contact with your partners. Then put a schema validation on an internal table that's secured by traditional perimeter defenses and end-point security. This will prevent anyone from being able to track the responses through the transactions and keep the data flowing between only you and your business partners.

## Judging Success by the Number of Services

Many organizations mistakenly associate the number of services they have with the success of their SOA. In fact, it's actually the opposite. Since one of the primary business benefits of SOA is reuse of services, a high number of services can be a prime indicator that your SOA isn't successful. The more services your organization has, the greater the likelihood that most of those services aren't being used.

When one organization researched the value of its SOA initiative, it learned that it had 25 services in production. However, out of those 25 services, only five were being used by other applications and other business lines. Most of the services were only being used as individual applications. I once heard someone give a keynote at a conference and tout the fact that his organization had 300 services and anticipated having 1,000 by the end of the year. That's not a measure of success; it's a train wreck in the making.


Not only is reuse a key benefit of SOA, it's also a useful tool for organizations looking to quantify the business value of their SOAs. By looking at reuse, you can determine how many times a service is being used, how many processes it's supporting, and the number of items being reused. You can use that data to calculate the costs savings for each instance of reuse, such as the saved architecture and design time and the saved development time.

## Conclusion

It's easy to be overwhelmed when embarking on an SOA initiative. While there are pitfalls to be aware of, it's important to focus on the positive business and IT benefits a SOA can have for your organization. Being aware of the "worst practices" to avoid can help you create an environment where doing the right thing is easy. These worst practices can help you create a blueprint for the processes necessary to make your SOA initiative a success. ■

### About the Author

*Daniel M. Foody, chief technology officer of Sonic and Actional products at Progress Software, leverages his extensive experience in enterprise systems software to design robust and manageable Service Oriented Architectures. Daniel's experience with distributed systems technologies including middleware, integration, and Web Services gives him a broad knowledge of the complexities and requirements for managing real-world enterprise software deployments. He's the author of various standards and contributed significantly to the OMG standard for COM/CORBA interworking. Most recently, Daniel was the recipient of InfoWorld's 2005 CTO 25 award. He holds a BSEE and MSEE from Cornell University.*



**ALL ATTENDEES RECEIVE:**  
Certificate of Completion **PLUS** Course Materials on DVD!\*

# AJAX ONE-DAY HANDS-ON INTENSE TRAINING!

AJAXWorld University's "AJAX Developer Bootcamp" is an intensive, one-day hands-on training program that will teach Web developers and designers how to build high-quality AJAX applications from beginning to end. The AJAX Developer Bootcamp is intended to be the premier AJAX instructional program presently available anywhere.

## AJAXWORLD UNIVERSITY BOOTCAMP

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

Roosevelt Hotel  
New York, NY  
January 22, 2007

Roosevelt Hotel  
New York, NY  
March 18, 2007

## What You Will Learn...

### Overview of AJAX Technologies

- HTML vs. DHTML
- Network Concerns
- Asynchronous Conversations with Web servers
- The characteristics of high-quality AJAX applications
- The Web page is the application
- What the server provides
- User interaction

### Understanding AJAX through the basics of AJAX

- Asynchronous server communication
- Dynamic HTML
- Javascript Design patterns
- User interface strategies for building elegant, highly addictive Web sites and applications
- The Essential AJAX Pieces
  - Javascript
  - Cascading Style Sheet (CSS)
  - Document Object Model (DOM)
  - XMLHttpRequestObject
- The AJAX Application with Javascript
- Using CSS
- Structuring the View Using the DOM
  - Applying Styles with Javascript
  - Communicating with the Web Server in the Background
  - Designing AJAX Applications
  - Design Patterns
- Introduction to AJAX Frameworks
  - Dojo, script.aculo.us, Prototype
  - Overview of framework capabilities
  - Examples of frameworks in use
  - Best Practices

### Hand-On Development The Fundamentals:

#### Building the Framing for an Ajax Application

- Review the courseware code with the Instructor
- Begin building a working AJAX application and start applying technique and technologies as introduced in class
- Create the basic AJAX application by creating HTML, Javascript, and CSS files
- Learn Best Practices and Validation
- Learn and add script.aculo.us effects
- Learn and add the Dojo Framework

### Adding Basic Ajax Capabilities to a Web Page:

#### Going Deep Into the AJAX User Experience

- Elements on the Rich Internet Experience
  - Interactivity
  - Robustness
  - Simplicity
  - Recognizable Metaphors
  - Preservation of the Browser Model Bookmarks/Back Button
- Background operations
- Building a AJAX Notification Framework
- Provenance and Relevance
- Rich Experience Support with Third-Party AJAX Client - Framework
- Using AJAX layouts, containers, and widgets
- Patterns for Animation and Highlighting
- User Productivity Techniques
- Tracking Outstanding Network Requests

### Hands-On Development:

#### Expand the Application with more Advanced Ajax

- Review the courseware code with the Instructor
- Expand the Mural Application
- Add Features using Dojo
- Add specific Dojo Libraries to support Ajax widgets

### Work on server side communications in the background

- Create a tabbed layout
- Create a submission form to upload to the server, all without reloading the page
- Create an Ajax submission form that will take uploads on one tab
- Create a form validation that ensures only the right information is submitted.
- More Stretch work for those who want to learn additional concepts

### Advanced AJAX Concepts

- Review Ajax Concepts
- SOA and Mashups
- Current state of Ajax Frameworks
- Web 2.0 and the Global SOA
- Ajax Constraints
- Design Patterns
- Javascript Timers
- Ajax Programming Patterns
- Performance and Throttling

### Hands-On Development:

#### Working with Advanced Ajax Capabilities

- Review the courseware code with the Instructor
- Work with the Accordion control
- Learn how to use the Tree control
- Explore Dojo's animation capabilities
- Explore how the debug output can be used in <div> elements
- Tour Dojo and RICO demos
- Experiment with new Dojo features from the Dojo demos
- source code and attempt to add them to various parts of the Mural application
- Overview of Future of AJAX and Rich Internet Applications

## What Attendees Are Saying...

“The trainer was excellent.  
The material too!”

“The hands-on, although long,  
was useful and educational!”

“The instructor was good. He  
answered questions thoroughly!”

“Well designed and organized.  
Good mix of lecture vs lots of  
hands-on!”

\*ALL RELEVANT COURSE MATERIALS WILL BE OFFERED ON DVD AFTER THE EVENT

**HURRY! REGISTER NOW FOR EARLY-BIRD DISCOUNT!**



For more great events visit [www.EVENTS.SYS-CON.com](http://www.EVENTS.SYS-CON.com)

# Exposing C Apps as Web Services

**SOA enabling C based legacy assets**



WRITTEN BY MOHIT CHAWLA AND VIJAYA BHASKAR PEDDINTI

➤ Most companies, especially from the banking domain, develop a large amount of their software in C/C++, and generally, they are not interested in redeveloping the code in any other new generation language, such as .NET or Java, due to cost and performance reasons. One well known advantage [1] to sticking with the C++ legacy application is performance. Most scientific applications are intentionally developed in C++ for the same reason. But then, integration of this existing functionality with the new application is a big issue. The solution to this problem is incubating service orientation into the application architecture and accessing the legacy code by exposing it as a service, based on open standards so that it can be used across various platforms.

One of the ways of implementing Service Orientated Architecture (SOA) is Web Services; it provides a strong foundation for software interoperability through open standards such as Simple Object Access Protocol (SOAP), Web Services Description Language (WSDL), eXtensible Markup Language (XML), etc. Web services can be readily used for leveraging legacy code which can then be seamlessly integrated with any application across platforms, allowing portability and reusability. Exposing the current legacy code as a Web service enables us to take advantage of the performance of the existing legacy application while working with modern-age technology.

## Tools/Frameworks

There are various tools/frameworks available, both open source and proprietary, through which we can create and deploy Web services written in C++. They also allow exposing existing application functionality written in C++ as a Web service. Some of the tools in this domain are listed below:

- gSOAP [2]
- Apache Axis C++ [3],
- Rouge Wave's LEIF [4]
- Systinet server for C++ [5]

Of these, gSOAP and Axis C++ are the open source tools, while LEIF and Systinet Server for C++ are proprietary tools. We have used gSOAP for exposing legacy code as a Web service, which I explain later in this article.

## gSOAP

GSOAP tool is an implementation of a SOAP protocol using C/C++ language, which helps in the development of SOAP Web services and client side development in C/C++. It provides a C/C++ SOAP API that hides SOAP-specific details from the user which eases the development of a service/client. The gSOAP compiler maps C/C++ data types to equivalent XML data types and vice-versa, hence, providing full interoperability.

The gSOAP package includes pre-build tools [6]:

- **WSDL/schema parser tool (wsdl2h.exe):** Imports one or more WSDL files and XML schemas to generate a header file with the service prototypes and the data types used.
- **Stub/skeleton compiler (soapcpp2.exe):** Using a header file, it generates (de)serialize routines for data types, stub and skeleton code. This tool will generate either client-stub or server-skeleton or both.

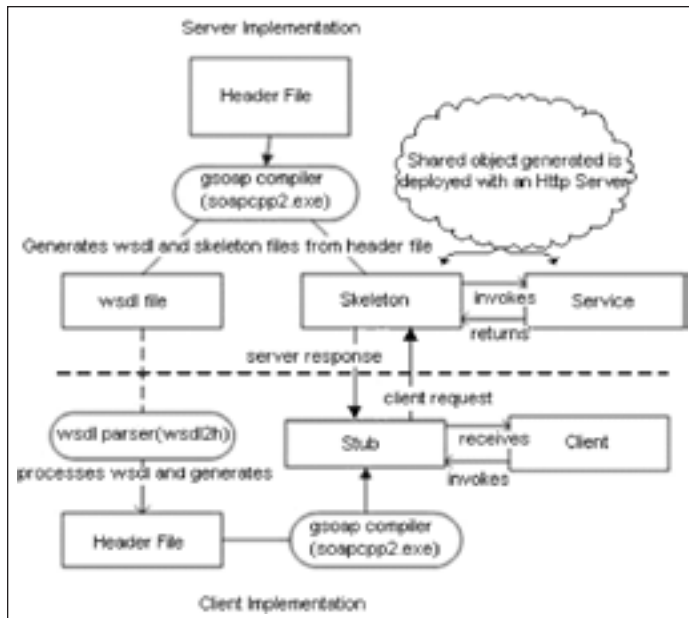


Figure 1: Creating the Web service using gSOAP

There are two ways to create and deploy a Web Service, as shown in Figure 1:

- A top down approach where you start with a WSDL.
- A bottom up approach where you start by writing a web service.

### With a WSDL

This approach is mostly used whenever a client wants to use an existing Web service or if a user starts by writing a WSDL file. A WSDL file consists of the descriptions of all service methods, in terms of data types used, their inputs and outputs, their location, namespaces used, etc. These descriptions in the WSDL file have to be C/C++ equivalent, so as to use/implement the Web services, i.e., “wsdl to header file” conversion is required. In order to generate a WSDL equivalent C/C++ header file, gSOAP provides a WSDL2H tool.

As explained earlier, a wsdl2h.exe is used to convert the .wsdl file to a header file, consisting of C/C++ equivalent data type mapping and function prototypes. Then this header file should be processed using the soapcpp2 stub/skeleton tool. This creates C/C++ source files that provide (de)serialize functions for the data types, stub and skeleton routines.

Using pre-build tools:

- wsdl2h -o out.h inputfile.wsdl
- soapcpp2 out.h

the wsdl2h command generates out.h from inputfile.wsdl. The header file is then processed by the gSOAP compiler to generate the stubs and skeletons as explained in the next section.

### By Writing a Web Service

Create a header file, say “service.h.” The SOAP service methods are declared in the header file as function prototypes. Then the header file should be processed by the soapcpp2 stub/skeleton compiler tool in order to generate the stubs and skeletons. This can be achieved by:

Soapcpp2 service.h

(Or)

Soapcpp2 -c service.h (to generate the code using pure C)

The compiler generates stub and skeleton routines for the function prototypes in service.h. The generated stub routines allow C/C++ client applications to interact with existing Web services. The skeleton routines are generated for each of the service methods in the header file which can be readily used to implement one or more of the service methods as a new SOAP Web service. The Stub/Skeleton code takes care of insertion and extraction of SOAP-XML messages and also provides the (de)marshal of the data types between clients and servers.

### An Example to Demonstrate How a Report Generator Developed in C Can be Exposed as a Web Service

GenerateReport method, which is being published as a Web service, takes templateName as an argument, which is an XML Template file based on which the actual report is generated. This function will return the path of the report (file location on the server).

### Writing a Function Prototype

The service function prototype has to be in the following syntax:

```
int [namespace prefix_] methodname ([input param list], output param);
```

The following are observations based on the syntax:

- The namespace prefix to a method is optional.
- The parameter list can have zero or more input parameters but a mandatory output parameter (a pointer that holds the output value).
- Service methods always return integer values upon success (SOAP\_OK else error constant).

The Listing below shows the service method signature and usage of the directive “//gsoap”. These directives are used to set the properties of a service. The directive below specifies the location of the service:

```
//gsoap ns service location: http://localhost:80/ReportService
int Report__generateReport( char *templateName, char **result );
```

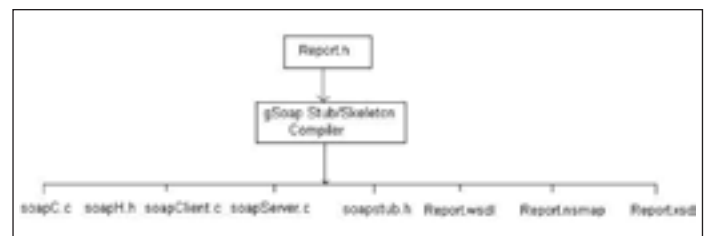


Figure 2: Processing with gSoap Stub and Skeleton Compiler

# Web Services provides a strong foundation for software interoperability through open standards such as Simple Object Access Protocol (SOAP), Web Services Description Language (WSDL), eXtensible Markup Language (XML), etc.

Now save the file and name it Report.h

Execute the soapcpp2.exe with the header file shown below:

`Soapcpp2 -c Report.h` (option -c is used for generating the files in pure c)

On successful execution, the compiler generates the following files:

- **soapH.h:** Has the function prototypes for the functions defined in soapC.c.
- **Report.wsdl:** Has the services descriptions. This file is published, so that client applications can use these remote service methods.
- **Report.nsmmap:** Provides namespace mapping table which is required to enable SOAP applications to validate the content of SOAP messages. This table is used at runtime to resolve namespace bindings and (de)serialize routines in order to validate the service request/responses.
- **soapStub.h:** Has the function prototypes of the soap wrapper methods, soap service dispatcher methods, and modified data types generated by the compiler.
- **soapC.c:** Contains the (de)serialization routine for the data types used by the service routines and also routines for generating SOAP-XML related code.
- **soapClient.c:** Has the functionality that creates the XML output with the header and the XML representation of the structure with the parameters of the function. This is responsible for sending and receiving the XML request and response messaging. Waits on the result and populates the result struct (output holding data type) with the elements of the received XML.
- **soapServer.c:** Provides lookup of service methods, dispatches a request to the appropriate service method, and ensembles the required input parameters from the received XML message and a reference to the output. On execution of the service, the output is populated, which is then serialized and an XML message is created and sent back to the client.

## Server Side Implementation

Listing 1 shows the service method implementation. The service method must be implemented in the server side application and the type signature of the method must be identical to the remote method specified in the header file (soapStub.h) shown below:

```
int Report_generateReport( struct soap[8] *soap, char *templateName,
char **outputFilePath);
```

The service function uses three SOAP API functions `soap_init`, `soap_serve` and `soap_receiver_fault`.

- **Soap\_init method:** Initializes the soap runtime environment.
- **Soap\_serve method:** Acts as a service dispatcher. It listens to client requests and invokes the appropriate service method through a skeleton routine. After the request is served, the response is encoded in SOAP and sent to the client.
- **Soap\_receiver\_fault:** Sets the fault string and fault details in the current runtime environment. Every remote method imple-

mented in a SOAP service returns an error code as a return value. SOAP\_OK denotes success and others denote an exception. A receiver error indicates that the service can't handle the request, but can possibly recover from the error. To return an unrecoverable error, use the function `soap_sender_fault`.

## Client Side Implementation

The client side implementation that uses the service can be done in any language, i.e., JAVA/C/.NET. Listing 2 shows how a C-Client uses the service. A client requests a service through the client stub routines(`soap_call_Report_generateReport`) generated by the stub compiler (see Figure 2).

The client code introduces us to few more SOAP related functions, namely, `soap_call_*` methods, `soap_print_fault`, `soap_end`, `soap_done`:

- **Soap\_call\_\* methods:** The method names would be of the form: `int soap_call_[namespace]_methodname(struct soap*,char*URL, char*action,parameters..)`
  - **struct soap:** Stores the soap runtime environment.
  - **URL:** Specifies where the service is deployed. The service location is specified as [http://...../mod\\_gsoap.dll?servicename](http://...../mod_gsoap.dll?servicename). The `mod_gsoap` allows `gsoap` shared libraries to be plugged into HTTP Servers. It will dynamically load the required `gsoap` servers at runtime.
  - **Action:** Specifies the SOAP Action. For each remote method declared, a client stub routine(`soap_call_method`) is generated. These routines are generated by stub/skeleton compiler. These functions act as wrappers and modify the parameters to and from the request/response message data structures, (de)serializes, and validates the data before sending and after receiving the result.
- **soap\_print\_fault:** During the processing of a request, if an exception is generated, then the appropriate error number is assigned to the current runtime environment. This method is used to print a descriptive message for the corresponding error to a specified stream (standard output or a custom file).
- **soap\_end:** This method internally calls the `soap_free` method which cleans up temporary and (de)serialized data created in the current runtime environment. This has to be used at the end of a transaction/messaging.
- **Soap\_done:** This function cleanly terminates a `gsoap` runtime environment, detaches callbacks registered to, and closes any sockets connection opened for message exchange. This has to be the last function called, as it detaches the soap struct attached to the runtime environment.

## Performance

The SOAP/XML services performance is highly driven by XML-Native Data type mapping, XML (de)encoding, XML Call Latency, floating point data type handling. `gSOAP` preprocessor translates XML directly into application data and vice versa. It uses an optimized two phase serializing algorithm [8] which reduces the runtime checking of object/pointer references. This, in turn, improves the latency/roundtrip messages achieved per second. Looking at the performance



**Figure 3:** Performance analysis using gSOAP, .NET, and Axis Java

of wrt, .NET, and Axis Java frameworks, it gives as much as 50 to 100% performance gain (depends on the network condition and machine configuration). We did performance analysis using gSOAP, .NET, and Axis Java by implementing the same basic Web services on each, and invoked the service from a Java client on the network. The Computer configuration used is :RAM 512MB, Windows XP, Intel® Pentium® 4CPU, 2.80GHz,2.79GHz. Results retrieved are as shown in Figure 3.

As you can see in Figure 3, gSOAP performed around 5% better than .NET and approx. 100% better than Axis Java on similar network conditions.

## Conclusion

The introduction of Web services and the advantages offered using gSOAP has made the migration of complex C/C++ business and scientific applications to a Web-based environment easier. Web service standards such as SOAP, Representational State Transfer (REST), and XML-RPC facilitate integration of such legacy applications over the Internet. SOAP/XML is a mechanism to integrate existing C/C++ programs with other client-side technologies. This article provides insight on how to expose an existing C/C++ functionality as a Web service using the open source gSOAP Library. ■

## References

- [1] Pistol, C., and Lungu, A. "Deploying C++ Grid Services: Options and Performance": [www.cs.duke.edu/~anda/cps296.5/final\\_projects/PaperWS.pdf](http://www.cs.duke.edu/~anda/cps296.5/final_projects/PaperWS.pdf)
- [2] gSOAP: <http://www.cs.fsu.edu/~engelen/soap.html>
- [3] Apache Axis C++: <http://ws.apache.org/axis/cpp/index.html>
- [4] Rogue Wave's LEIF: <http://www.roguewave.com/products/leif/>
- [5] Systinet Server for C++: <http://www.systinet.com/products/ssc/overview>
- [6] gSOAP 2.7.8 User Guide: [www.cs.fsu.edu/~engelen/soapdoc2.pdf](http://www.cs.fsu.edu/~engelen/soapdoc2.pdf)
- [7] Struct soap definition: <http://www.cs.fsu.edu/~engelen/stdsoap2.h>
- [8] Toward Remote Object Coherence with Compiled Object Serialization for Distributed Computing with XML Web Services: <http://www.cs.fsu.edu/~engelen/cpcpaper06.pdf>
- [9] Gupta: Masters Project Report.

### About the Authors

Mohit Chawla is a software engineer with the Web Services Center of Excellence at Infosys Technologies, Hyderabad. His primary area of interest is SOA, with specific focus on Web services implementations on various platforms. He has also actively participated in publishing in leading international conferences.

[mohit\\_chawla@infosys.com](mailto:mohit_chawla@infosys.com)

Vijaya Bhaskar Peddinti is a Software Engineer with Web Services Centre of Excellence in SET Labs, Infosys Technologies, Hyderabad. His experience has been in product development of various sizes based on SCA.

[vijayabhaskar\\_p01@infosys.com](mailto:vijayabhaskar_p01@infosys.com)

## Listing 1

```
#include "soapH.h"
#include "Report.nsoap"

struct xmlObject
{
    //..
    //..
};

typedef struct xmlObject XMLObject;
int main()
{
    struct soap soap;
    soap_init( &soap );
    soap_serve( &soap );
    return 0;
}

int Report_generateReport( struct soap* soap, char *templateName,
char** outputFilePath )
{
    int errorStatus = 0;
    XMLObject xmlObject;
    // CreateXMLObjectFromFile method create and Object Graph/tree
    // from the file templateName
    errorStatus = CreateXMLObjectFromFile( templateName, &xmlObject );

    if( errorStatus )
    {
        return soap_receiver_fault( soap, "File Read Error\n",
        "File Read Error\n" );
    }
    // renderReport render the actual report using xmlObject
    errorStatus = renderReport( xmlObject, outputFilePath );
    if( errorStatus )
    {
        return soap_receiver_fault( soap, "Error while Rendering
        Report\n", "Error while Rendering Report\n" );
    }
}
```

## Listing 2

```
#include "soapH.h"
#include "Report.nsoap"

const char* server =
"http://localhost/ReportService/mod_gsoap.dll?ReportService";
const char* action = NULL;
int main()
{
    int res;
    struct soap soap;
    char * templateName = "XML.xml";
    char *outFileName = NULL;

    soap_init( &soap );

    soap_call_Report_generateReport( &soap, server, action,
    templateName, &outFileName );

    if( soap.error )
    {
        soap_print_fault( &soap, stderr );
        return 1;
    }

    /*
    .. code to view the generated report
    */

    soap_end(&soap); // dealloc deserialized data
    soap_done(&soap); // cleanup and detach soap struct
    return 0;
}
```

# Keep to the Original Intent of SOA

## Best-of-breed versus proprietary approaches

WRITTEN BY DAVID BESEMER

➤ The original inspiration for Service-Oriented Architecture (SOA) was vendor neutrality and interoperability among best-of-breed technology components, comprising a cohesive system that is flexible and adaptable enough to meet ever-changing enterprise demands. Yet, recent news from well-regarded SOA technology vendors might lead one to believe SOA is heading in the opposite direction, toward a multi-faceted yet proprietary solution from a single source provider. What should enterprises consider as they plan SOAs, or do if their SOA implementations are already underway? This article reviews the best-of-breed versus proprietary approaches, and draws conclusions based on the original intentions for SOA.

**S**OA is used to provide standardized interfaces between enterprise systems so that one application can easily utilize the services (data or process logic) of another. SOA promotes interoperability and reuse of information assets, enabling IT professionals to build and change applications faster in order to keep up with rapidly changing business demands. But how exactly do you achieve an SOA to reap all of its promised benefits? There are a myriad of solutions and approaches available to help companies with their SOA implementations.

Let's first look at the proprietary approach whereby you obtain all the SOA pieces from one vendor. These pieces include the data services layer, the transaction services layer, the business process management (BPM) layer, and a services registry. The relative importance of each piece may be debatable, but many people will agree that these are undoubtedly four of the main pieces. IBM might be the only vendor in the market that can legitimately claim to offer all these capabilities, with perhaps one or two other vendors having credible claims to such a wide offering. The advantage of a proprietary approach is guaranteed compatibility among all the components. The disadvantage is that each

component may not be the best available in the industry, as each piece may have inferior functionality to other solutions provided by other vendors that specialize in a specific piece of SOA.

At the other end of the spectrum is the best-of-breed approach by which you can pick and choose the best products in the market place for the function required. These solutions can come from established vendors or from startups. If a suitable solution is not available, you can choose to develop the required software in house. Companies can choose from a wide range of available products for data services, transaction services, business process management, and registry. The advantage of this approach is that you can pick the best available technology designed to meet your specific needs. The downside is that you run the risk that the parts may not work together. But, as we'll see, if SOA lives up to its promises, this is actually not a concern.

The foundation of SOA is interoperability. This allows companies to leverage their existing assets, and permits them to plug in best-of-breed products as needed to support their businesses. They are no longer forced to sunset a perfectly useful system just because the technology is "outdated" nor to implement an entire suite of prod-



ucts for only one particular module. The value of this is fairly straightforward, but one of the benefits that stands out is that now business requirements drive IT and are no longer encumbered by it.

In a typical IT environment, you have a mixed set of tools and systems. You might develop web-based applications using IBM, BEA, or even open-source products like JBoss. You may also have tried-and-true systems like SAP and Oracle E-Business Suite running your back office. Merger and acquisition activities may bring numerous other disparate systems into your IT environment. The need for systems to talk to each other, or use data that resides within another system, plus combine information from these various systems to execute business processes that lead to effective business decision making, is at the heart of the need for an SOA. SOA promotes (if not assumes) mixed technology from multiple vendors and in-house developers interacting through agreed standards.

Could we realistically expect one vendor to provide all the needs for SOA? We described four major components for an SOA earlier. In addition, there are numerous supporting pieces that are pertinent to the architecture, a list that is likely to expand in the future. SOA is simply too big of an

initiative to be fully addressed by a single vendor. Even if we look at examples of a single vendor addressing a large need for its customers, we see that it cannot provide everything. For example, SAP is often cited as the single-source vendor for back-office management. However, loyal SAP users will admit that the system handles financials and manufacturing operations very well, but is less complete when it comes to the areas of supply chain management or customer relationship management, so they will implement complementary best-of-breed IT products. The customer is best served if these products work well together. The same is true of SOA.

There are many advantages to the best-of-breed approach, other than the right tool for the right job. With best-of-breed, by definition, you get what you need. You do not have to wait for any particular incumbent vendor to provide the functionality that you need in the "next release." Best-of-breed is also a practical approach, in that you only purchase what you need and nothing more. Why spend beyond your budget on tools and functionality that you don't need because it is included in an "enterprise deal"? You may believe that

you have a toolset that will allow you to meet your needs for the future, but, as you proceed in your SOA implementation, you discover unanticipated needs not covered in the "one-size-fits-all" tool set. If done correctly, the best-of-breed approach allows you to grow and evolve with your changing IT environment.

Let's take a closer look at an example of a best-of-breed component of SOA and how it can grow and evolve within an IT environment. One of the critical pieces of SOA is a data services layer. Data services access and combine information from existing systems and deliver the data as SOA services. These services are used in applications that require data from across the enterprise. Every SOA needs a data services layer to abstract the complexity and location of enterprise data, making the data more available and reusable. You can start by allowing the data services layer to abstract only a few of your existing systems, and, as you add more systems to your environment, the data services layer expands to tap into those systems as well.

SOA is not just about technology. SOA implementations also need to take into account the people and process realities

involved in the IT environment. A couple of things to consider are the ease of use of the technology and whether the tools leverage knowledge already in house. A tool that has all the features in the world but is not easy to use would be impractical because it would never be used, so promised benefits would never see the light of day. Introducing a technology that requires developers and users to learn it from scratch also faces resistance, so, in choosing the best-of-breed component, you should consider if it's the best fit for your people and processes.

Companies will go about their SOA implementations in a way that best fits their current IT environment. There are advantages to a proprietary approach, but there are far more advantages to a best-of-breed approach. Best-of-breed not only provides the right tool for the specific job, it allows growth as the SOA expands and adapts to changing business needs. Therefore, the best-of-breed approach is the most practical approach to SOA, and provides the most flexibility. ■

#### About the Author

David Besemer is CTO, Composite Software.



## Looking to Stay Ahead of the i-Technology Curve?

**Subscribe to these FREE Newsletters** >

Get the latest information on the most innovative products, new releases, interviews, industry developments, and i-technology news

Targeted to meet your professional needs, each newsletter is informative, insightful, and to the point.  
**And best of all – they're FREE!**

Your subscription is just a mouse-click away at **www.sys-con.com**














**SYS-CON MEDIA**  
The World's Leading i-Technology Publisher

# Is It Done Yet?

## Three steps to checking in your code with confidence

WRITTEN BY DR. ADAM KOLAWA

➤ It's difficult to determine how much time to spend reviewing and testing your code before checking it in to the team's shared code base. On the one hand, you want to complete and check in code as rapidly as possible so you can meet deadlines and move on to developing new code or getting started on other projects. After all, you went into software development to develop, not to test.



**Y**et, if you move too fast, you might end up checking in code that causes bugs—immediately upon integration, or later on when the code is reused, extended, or maintained. In that case, any time that you originally saved by checking in the code prematurely is significantly outweighed by the time you need to spend diagnosing the problem, correcting the responsible code (and possibly code that has been layered upon that code), and verifying the correction—not to mention the hassle of having to interrupt whatever you're currently working on and return to something you previously wrote off as “done” and forgot about.

This article explains three steps you can take to significantly reduce the risk that code will come back to haunt you after you check it in:

1. As you write code, comply with best development rules to prevent reliability, security, performance, and maintainability problems in the code.
2. Immediately after each piece of code is completed or modified, use unit-level reliability testing to ensure that it's reliable and secure.
3. Immediately after each piece of code is completed or modified, use unit-level functional testing to verify that it's implemented correctly and functions properly.

All three steps can be automated with commercial and/or open source tools so you can gain their benefit without disrupting your development efforts or adding overhead to your already hectic schedule.

### Step 1: Comply with development rules to improve code reliability, security, performance, and maintainability of the code

The first step in determining if your code is done is to ensure that it complies with applicable development rules as you write it. Many developers think that complying with development rules involves just beautifying code. However, there is actually a wealth of available development rules available for every modern development language that have been proven to improve code robustness, security, performance, and maintainability. In addition, each team's experienced developers have typically developed their own (often informal) rules that codify the application-specific lessons they've learned over the course of the project.

Even if your team is not already following a set of formal or informal development rules, we strongly recommend that you make rule compliance a team effort. If you are the only developer on the team following the rules, your code will undoubtedly improve. But if all your team members aren't on the same page, dangerous code could still enter the code base, and your own efforts might conflict with (or be overwritten by) those of your teammates. Having a development team inconsistently apply software development standards and best practices as it implements code is like having a team of electricians wire a new building's electrical system with multiple voltages and incompatible outlets. In both cases, the team members' work will interact to form a single system. Consequently, any hazards, problems, or even quirks introduced by one “free spirit” team member who ignores the applicable guidelines and best practices can make the entire system unsafe, unreliable, or difficult to maintain and upgrade.

### Why bother?

The key benefits of complying with applicable development rules are:

***It cuts development time and cost by reducing the number of problems that need to be identified, diagnosed, and corrected later in the process.***

Complying with meaningful development rules prevents serious functionality, security, and performance problems. Each defect that is prevented by complying with development rules means one less defect that the team needs to identify, diagnose, correct, and recheck later in the development process (when it's exponentially more time-consuming, difficult, and costly to do so). Or, if testing does not expose every defect, each prevented defect could mean one less defect that will impact the released/deployed application. On average, one defect is introduced for each ten lines

## Step 1: (continued)

of code (A. Ricadela, "The State of Software", InformationWeek, May 2001) and over half of a project's defects can be prevented by complying with development rules (R.G. Dromey, "Software Quality – Prevention Versus Cure", Software Quality Institute, April 2003). Do the math for a typical program with millions of lines of code, and it's clear that preventing errors with development rules can save a significant amount of resources. And considering that it takes only 3 to 4 defects per 1,000 lines of code to affect the application's reliability (A. Ricadela, "The State of Software", InformationWeek, May 2001), it's clear that ignoring the defects is not an option.

### ***It makes code easier to understand, maintain, and reuse***

Different developers naturally write code in different styles. Code with stylistic quirks and undocumented assumptions probably makes perfect sense to the developer as he's writing it, but may confuse other developers who later modify or reuse that code—or even the same developer, when his original intentions are no longer fresh in his mind. When all team members write code in a standard manner, it's easier for each developer to read and understand code. This not only prevents the introduction of errors during modifications and reuse, but also helps developers work faster and reduces the learning curve for new team members.

### **How do I do it?**

#### ***Decide which development rules to comply with***

First, as a team, review industry-standard development rules for the language and technologies you are working with and decide which ones are most applicable to your project and will prevent the most common or serious defects. The rules implemented by automated static analysis tools offer a convenient place to start. If needed, you can supplement these rules with the ones listed in books and articles by experts in the language or technology you are working with.

Next, consider practices and conventions that are unique to

your organization, team, and project (for instance, an informal list of lessons learned from past experiences). Do your most experienced team developers have an informal list of lessons learned from past experiences? Have you encountered a specific bug that can be abstracted into a rule so that the bug never occurs in your code stream again? Are there explicit rules for formatting or naming conventions that your team is expected to comply with?

### ***Configure all team tools to check the designated rules consistently***

To fully reap the potential benefits of complying with development rules, the entire development team must check the designated set of rules consistently. Consistency is required because even a slight variation in tool settings among team members could allow non-compliant code to enter the team's shared code base. Just one overlooked rule violation could cause serious problems. For instance, assume that your team member is not checking the same rules as everyone else, and consequently checks in code that does not comply with rules for closing external resources. If your application keeps temporary files open until it exits, normal testing—which can last a few minutes or run overnight—won't detect any problems. However, when the deployed application runs for a month, you can end up with enough temporary files to overflow your file system, and then your application will crash.

### ***Check and correct new/modified code as its written.***

Study after study has shown that the earlier a problem is found, the faster, easier, and cheaper it is to fix. That's why the best time to check whether code complies with development rules is as soon as it's written or updated. If you check whether each piece of code complies with the designated development rules immediately, while the code is still fresh in your mind, you can then quickly resolve any problems found and add it to source control with increased confidence.

## Step 2: Use reliability testing to verify that each piece of code is reliable and secure

The next step toward reliable and secure code is to perform unit-level reliability testing (also known as white-box testing or construction testing). This involves exercising each function/method as thoroughly as possible and checking for unexpected exceptions.

### **Why bother?**

If your unit testing only checks whether the unit functions as expected, you can't predict what could happen when untested paths are taken by well-meaning users exercising the application in unanticipated ways—or taken by attackers trying to gain control of your application or access to privileged data. It's hardly practical to try to identify and verify every possible user path and input. However, it's critical to identify the possible paths and inputs that could cause unexpected exceptions because:

### ***Unexpected exceptions can cause application crashes and other serious runtime problems***

If unexpected exceptions surface in the field, they could cause instability, unexpected results, or crashes. In fact, Parasoft has worked with many development teams who had trouble with applications crashing for unknown reasons. Once these teams started identifying and correcting the unexpected exceptions that they previously overlooked, their applications stopped crashing.

### ***Unexpected exceptions can open the door to security attacks***

Many developers don't realize that unexpected exceptions can also create significant security vulnerabilities. For instance, an exception in login code could allow an attacker to completely bypass the login procedure.

## Step 2: (continued)

### How do I do it?

#### **Design, implement, and execute reliability test cases.**

To identify potential uncaught runtime exceptions in newly added/modified code, you test each class's methods with a large number and range of potential inputs, then check whether uncaught runtime exceptions are thrown.

Manually developing the required number, scope, and variety of unit test cases that would expose exceptions is impractical. Achieving the scope of coverage required for effective white-box testing mandates that a significant number of paths are executed. For example, in a typical 10,000 line program, there are approximately 100 million possible paths; manually generating input that would exercise all of those paths is infeasible and practically impossible.

When trying to expose exceptions, a tool that automatically generates test cases is essential. If test design and generation is automated, the only user intervention required is to review the findings and address the reported exceptions.

### **Review and address all reported exceptions.**

After the first test run completes, review the coverage. If any classes received less than 75% coverage, we recommend that you customize the automated test case generation settings (for instance, by modifying automatically-generated stubs, adding realistic objects or stubs, or modifying test generation settings) so that the automated test case generation can cover a larger portion of that class during the next test run.

After you rerun the test, review all exceptions exposed by the tests, then address them before proceeding. Each method should be able to handle any valid input without throwing an exception. If code should not throw an exception for a given input, the code should be corrected. If the exception is expected or if the test inputs are not expected/permissible, document those requirements in the code and tell the tool that they are expected. This prevents most unit testing tools from reporting these problems again in future test runs. Moreover, when other developers who are extending or reusing the code see documentation that explains that the exception is expected behavior, they will be less likely to misunderstand the code and introduce bugs.

## Step 3: Use functional testing to verify that each piece of code is implemented correctly and operates properly

Next, extend your reliability test cases to verify each unit's functionality. The goal of unit-level functional testing is to verify that each unit is implemented according to specification before that unit is added to the team's shared code base.

### Why bother?

The key benefit of verifying functionality at the unit level is that it allows you to identify and correct functionality problems as soon as they are introduced, reducing the number of problems that need to be identified, diagnosed, and corrected later in the process. Finding and fixing a unit-level functional error immediately after coding is easier, faster, and from 10 to 100 times less costly than finding and fixing that same error later in the development process. When you perform functional testing at the unit level, you can quickly identify simple functionality problems, such as a “++” in a prefix notation substituted for a “++” in postfix, because you are verifying the unit directly. If the same problem entered the shared code base and became part of a multi-million line application, it might surface only as strange behavior during application testing. Here, finding the problem's cause would be like searching for a needle in a haystack. Even worse, the problem might not be exposed during application testing and remain in the released/deployed application.

### How do I do it?

#### **Add and execute more functional test cases as needed to fully verify the specification**

After you've worked through the exceptions reported by the automatically generated test cases, check if the code you've written actually functions as desired. Functional unit tests are meant to do just that. Without regard to internal function behavior, this means specifying function inputs and checking if the output is as expected. Such tests should be created based on the class API specification, or class use cases.

Functional tests can be written using whatever version of the xUnit framework is appropriate for the language you are using (JUnit for Java, NUnit for .NET languages, CppUnit for C++, etc.).

How do you know when you've completed sufficient functional testing on a piece of code? When 1) you have developed a functional test suite that is rich enough to verify that the specified functionality is implemented correctly and 2) the code passes that test suite with no failures.

## Can I Check It In Now?

Yes! Of course, there's no guarantee that if you follow these three steps before you check in your code, you will never see an annoying bug report again. But you will notice that you eventually spend less time finding and fixing bugs, which means fewer interruptions, less “crunch time” at the end of the project, and more time for more challenging and interesting tasks such as developing and implementing new technologies. The key to saving time in the long run is to automate these steps as much as possible so flushing errors out of your code before check in can become an essential yet unobtrusive part of your normal day-to-day work. ■

### **About the Author**

Dr. Adam Kolawa is cofounder and CEO of Parasoft, a vendor of automated error-prevention software and services based in Monrovia, CA. Dr. Kolawa, who is the coauthor of *Bullet-proofing Web Applications* (Wiley, 2001), has written and contributed hundreds of commentary pieces and technical articles for publications such as *The Wall Street Journal*, *CIO*, *Computerworld*, *Dr. Dobbs' Journal*, and *IEEE Computer*. He has also authored numerous scientific papers on physics and parallel processing. He holds a PhD in theoretical physics from the California Institute of Technology.

[ak@parasoft.com](mailto:ak@parasoft.com)

REGISTER TODAY AND SAVE!

***Rich Internet Applications: AJAX, Flash, Web 2.0 and Beyond...***

**www.AjaxWorldExpo.com**

# AJAXWORLD<sup>TM</sup>EAST

## CONFERENCE & EXPO



# NEW YORK CITY

THE ROOSEVELT HOTEL LOCATED AT MADISON & 45<sup>th</sup>

**SYS-CON Events is proud to announce the  
AjaxWorld East Conference 2007!**

**The world-beating Conference program will provide developers and IT managers alike with comprehensive information and insight into the biggest paradigm shift in website design, development, and deployment since the invention of the World Wide Web itself a decade ago.**

The terms on everyone's lips this year include "AJAX," "Web 2.0" and "Rich Internet Applications." All of these themes play an integral role at AjaxWorld. So, anyone involved with business-critical web applications that recognize the importance of the user experience needs to attend this unique, timely conference – especially the web designers and developers building those experiences, and those who manage them.

**BEING HELD MARCH 19 - 21, 2007!**

We are interested in receiving original speaking proposals for this event from i-Technology professionals. Speakers will be chosen from the co-existing worlds of both commercial software and open source. Delegates will be interested in learning about a wide range of RIA topics that can help them achieve business value.



## BlueTie Launches TieIn – a Web Services Platform for Centralized Management and Custom Development

(Rochester, N.Y.) – BlueTie, a provider of business e-mail and calendaring services, has launched TieIn, a Web services platform that gives resellers the ability to manage all of their customers in a single interface and develop new mash-ups. TieIn enables resellers to increase service revenues while reducing the time and costs to provide these services – giving partners an alternative to tedious Microsoft Exchange implementation and administration. TieIn combines a Web-based Extranet – for centralized provisioning, private-labeling and administration – with an XML-based Web services API that enables custom development and mash-ups. These tools give partners the ability to instantly manage all of their customers and extend BlueTie in new ways. TieIn is immediately available to BlueTie's 700 resellers worldwide, as well as new partners. By private-labeling BlueTie services directly through TieIn, partners can also strengthen their brand identity, increase customer satisfaction and generate additional professional services revenues. [www.bluetie.com](http://www.bluetie.com)



## Xignite Web Services Powered McDonald's Corporation ChipotleExchange.com

(San Mateo, CA) – Xignite, a pure-play provider of financial Web services for mission-critical corporate applications, has announced that it delivered live stock market data to a custom Website used to provide market information in connection with McDonald's recent exchange of its common stock for class B common stock in Chipotle Mexican Grill. The pricing mechanism for the transaction was based in part on calculation of "daily volume-weighted average price" (or VWAP) of McDonald's common stock and Chipotle class A common stock. Xignite's VWAP web service was used to provide the VWAP data via a custom website built and hosted by Xignite. [www.Xignite.com](http://www.Xignite.com)



## Solera Completes Acquisition of Insurance Web Services Organization in the Netherlands

(San Ramon, CA) – ABZ, the Solera company operating in the Netherlands, has announced that it has completed its acquisition of CATO. CATO operates the country's leading Web portal that enables efficient information exchange between insurance carriers, their corporate clients, and government agencies. The acquisition enables ABZ to enter the medical information reporting segment, an emerging business segment in the Netherlands with significant global growth potential. This is Solera's first acquisition since the purchase of Audatex from ADP in April 2006. [www.abz.nl](http://www.abz.nl)



## New Software AG Product Helps Customers Reduce the Costs of System Maintenance and Reuse

(Reston, VA) – Software AG has announced a new tool that allows organizations to reduce their IT system maintenance costs. In addition, Natural Engineer for Refactoring lets customers analyze, restructure and streamline existing applications so they can be more easily incorporated into new Web-based applications - often as part of a Service-Oriented Architecture. The initial release can modernize software applications written in Software AG's own development language Natural. This is a further step in the company's focus on providing new value to customers out of existing systems. In the future, Software AG will extend the scope of development languages that can be analyzed and restructured, and provide cost savings and reuse possibilities to a broader market. [www.softwareag.com](http://www.softwareag.com)



## Loomia Selects Mashery to Manage Developer Community

(San Francisco) – Mashery, a new company that helps online businesses build, promote, support and manage access to Web services and data, has announced that Loomia, an easy-to-use online personalization service provider, has selected the Mashery ProMash Solution to manage its online developer community. San Francisco-based Loomia helps retailers and media sites provide Amazon or Netflix style recommendations to their users driven by user behavior. Loomia counts iFilm, Learn out Loud, and GigaVox Media among its client base. According to market research firm Gartner, by the end of 2008, 75% of enterprise software providers will have a Web-based, software-as-a-service delivery model that includes Web APIs to create mashups. By 2010, Web mashups will be the dominant model for the creation of composite enterprise applications. [www.mashery.com](http://www.mashery.com) [www.loomia.com](http://www.loomia.com)



## ThinkFree Develops Free Integration Tools

(San Francisco) – ThinkFree, Inc., developer of an online office productivity suite, has announced the launch of ThinkFree Viewers, free tools that allow Web services, such as Coil ([www.coil-os.com](http://www.coil-os.com)), a virtual office application for work processing, to add office productivity functionality to their applications and Web sites. Viewers also provide Web users with an easy way to view office documents on the Web or from their desktop without having to purchase office suite applications. ThinkFree Online is a complete Internet-based office productivity suite that blends the best of the Internet's collaboration features with comprehensive Microsoft Office compatibility. ThinkFree provides easy integration of their tools for online creation, collaboration and editing of Microsoft Office-format documents with other Web services like Coil. The combination of ThinkFree and Coil brings together content creation services and a work-processing platform for end-to-end office productivity. [www.thinkfree.com](http://www.thinkfree.com)



## Intalio|BPMS Sets the Standard for BPM Performance

(Redwood City, CA) – Intalio, Inc., The Open Source BPMS Company, has announced the completion of a new performance benchmark for Intalio|BPMS. Using the newly released 4.3 version of its flagship BPM product, Intalio reached new levels of performance for the execution of BPEL processes. On a dual-CPU machine equipped with Intel Xeon processors and 2GB of RAM, the Intalio|BPMS executed over 3.5 million persistent processes in less than 24 hours. Persistence was offered by a single instance of the MySQL database engine deployed on a separate server. When persistence was de-activated, over 17 million transient processes were executed within the same 24-hour period. In a separate benchmark, Intalio demonstrated that round-trip calls made to external Web Services through an embedded Enterprise Service Bus (ESB) could be completed within less than 14 milliseconds. In yet another test, Intalio demonstrated that Intalio|Server could run at maximum capacity during several days without measurable decrease in transaction throughput. [www.intalio.com](http://www.intalio.com)



## LogicLibrary Introduces SOA FastPath

(Pittsburgh) – LogicLibrary, a provider of design-time SOA governance and software reuse, has introduced SOA FastPath. SOA FastPath leverages the knowledge and data that LogicLibrary Professional Services has acquired over multiple years of guiding organizations with their initial SOA deployments and the subsequent engagements that drove the maturation of these deployments. Now enterprises can implement an “out-of-the box” SOA project without being encumbered by configuring governance policies, user roles, and metadata. While organizations may realize the inherent value of SOA and the need for a design-time repository and associated SOA governance, they often struggle with how to get started – what governance processes to use, how and when to report to management, and the myriad of other details that must be decided upon. To address these issues, LogicLibrary has created SOA FastPath. Built upon Logidex coupled with pre-configured SOA Logidex templates, this product delivers everything an enterprise needs for a successful design-time SOA governance strategy. [www.logiclibrary.com](http://www.logiclibrary.com)



## New Xcalia Software Enables Cost Containment and Deployment Control for Applications in SOA Environments

(Palo Alto, CA / Paris) – Xcalia, a provider of dynamic integration software, has announced the availability of the Xcalia Intermediation Core (XIC) version 5.0. This new product enables enterprises to build composite applications with the ability to decide at runtime the way applications access data and service resources. XIC 5.0 also greatly reduces their overall SOA infrastructure costs, and, meanwhile, improves control over how the business applications are deployed and maintained. The “secret sauce” in XIC 5.0 is the ability for applications to dynamically access the appropriate data or service resources at runtime – even in heterogeneous environments – based on the pre-determined cost analysis of resource deployment options. The foundation for this new software infrastructure approach is “intermediation,” a dynamic alternative to static integration. With this approach, Xcalia utilizes a patented infrastructure layer that sits between applications and data, where all the business logic resides, so that no code changes are required in the base applications. As a result, Xcalia's intermediation approach leapfrogs the performance and scalability of the XML integration approaches used today. [www.xcalia.com](http://www.xcalia.com)



## Fiorano Launches Component Gallery – Pioneering iTunes-Like Experience in SOA Application Assembly

(Los Gatos, CA) – Fiorano Software Inc., a provider of business integration and middleware solutions, has launched the first Component Gallery to speed the delivery of modular business solutions in a world of distributed computing. The initial rollout of the gallery contains several useful components including connectors to databases, files, sophisticated XSLT transformers, HTTP sources, Web services, various flow components and middleware bridges. Experience with multiple complex production deployments indicates that over 75% of most common business processes can be deployed “out of the box,” without programmer intervention. Components in the gallery use the JCA (Java Connector Architecture) and JMS (Java Message Service) interfaces, allowing any programmer worldwide to contribute to the gallery. Fiorano has a certification program in place to control the quality of released components. Components in the gallery can be developed in a variety of supported languages, including Java, C, C++, C# and all .NET languages including Visual Basic. [www.fiorano.com](http://www.fiorano.com)



## AdventNet Upgrades QEngine with Support for Geographically Distributed Test Automation

(Pleasanton, CA) – AdventNet, Inc., a provider of affordable network and systems management software for enterprises, equipment vendors, and service providers, has announced the release of QEngine 6.6, a cost-effective, extensible and complete Web-based test automation tool for functional, performance, Web services, and regression testing of Web applications/Web services. The software helps QA professionals and test automation engineers to improve productivity and save significant time and resources, when compared to manual testing. QEngine is an enterprise class software that provides a client/server architecture to test from any location with just a browser and QEngine Toolbar installed at the client end. QEngine Server provides a uniform and centralized Web-based console to all the users connected via a toolbar to create, modify, schedule tests, analyze test results, and manage defects and issues thus enabling the seamless transfer of knowledge and test case data. [www.adventnet.com](http://www.adventnet.com)



## Tidal Announces Intersperse 6.0 to Simplify Management and Analysis of SOA Applications

(Palo Alto, CA) – Tidal Software, a provider of application scheduling and performance management software, has announced the general availability of Tidal Intersperse 6.0, the latest release of its comprehensive solution for monitoring and managing SOA solutions based on Java and .NET. With Intersperse 6.0, Tidal Software brings to SOA performance management detailed transaction tracing that can readily capture transaction flow inside components, between tiers, and across application servers for end-to-end monitoring and management in dynamic production environments. The release leverages detailed transaction tracing to build composite SLAs, enabling proactive performance management in the business context – across the multiple applications and servers that are often involved in composite SOA applications. Intersperse 6.0 delivers a readily customizable user interface to provide a visual, role-based display of the wide range of information needed for managing SOA deployments. With this release, Intersperse 6.0 now supports Tomcat and SAP NetWeaver and offers expanded coverage for WebLogic, WebSphere, JBoss, and .NET. [www.tidalsoftware.com](http://www.tidalsoftware.com)



## webMethods Brings SOA Governance to the Mainstream

(Washington, DC) – webMethods, Inc., a business integration and optimization software company, has unveiled two new offerings for service-oriented architecture (SOA) governance: webMethods Infravio X-Registry and webMethods Infravio X- Broker. Together, they form the industry's most comprehensive platform for addressing the entire SOA governance lifecycle, spanning design-time, runtime, and change-time environments, from the unique perspective of each user. Both webMethods Infravio X-Registry and webMethods Infravio X-Broker will be maintained on an independent basis as best-in-class products for SOA governance. They will also be offered as integrated components of the webMethods Fabric product suite, beginning with version 7.0, which is scheduled to begin shipping this December. [www.webmethods.com](http://www.webmethods.com)



### **TIBCO to Resell Systinet Registry**

(Palo Alto, CA) –TIBCO Software Inc has announced an agreement with Systinet, a division of Mercury Interactive Corporation, to resell Systinet Registry as a component of TIBCO's comprehensive service-oriented architecture (SOA) governance solutions. Governance is a key requirement to help mitigate risk and effectively align business and IT environments for a successful SOA implementation. TIBCO chose Systinet Registry because it provides a simple, standards-based means for publishing and discovering reusable business assets and offers enterprise organizations greater control and visibility into the services lifecycle. [www.tibco.com](http://www.tibco.com)



### **Cape Clear Announces General Availability of Cape Clear BAM**

(San Mateo, CA) – Cape Clear Software, an Enterprise Service Bus (ESB) Platform provider delivering on the promise of Service-Oriented Architecture (SOA), has announced the general availability of Cape Clear Business Activity Monitoring (BAM) as an embedded product family in Cape Clear 6.7's ESB Platform. Cape Clear BAM is a solution that monitors real-time and historical business performance across complex, heterogeneous environments through the use of ESB and SOA technologies.

Cape Clear BAM is comprised of two products:

1. Cape Clear BAM - SOA Monitor – Allows IT operations, and application and service delivery support to quickly assess application health, ensure SLA conformance, troubleshoot and diagnose problems, analyze impacts and provide predictive monitoring to identify IT incidents before they impact business operations.
2. Cape Clear BAM - Business Monitor – Provides business operations, process owners and executives real-time access to business key performance indicators across a 360-degree view of business processes, allowing improved responsiveness to business exceptions and events.

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



### **Envox Worldwide Introduces Envox CT Connect 7**

(Westborough, MA) – Envox Worldwide, a provider of voice solutions, has announced the availability of Envox CT Connect7, the latest version of the company's standards-based CTI software. Envox CT Connect 7 provides enhanced CTI capabilities for IP communication networks and integration with platforms based on Web services or service-oriented architectures. Envox CT Connect 7, which makes it faster and easier to build solutions to improve customer service and contact center operations, is helping to drive the transition from proprietary systems to communications solutions based on open standards, such as SIP and Web services. [www.envox.com](http://www.envox.com)



### **Sony Ericsson Releases UIQ 3 Web Services Package**

(Stockholm) – Sony Ericsson Developer World has released a complete package for Web services development, enabling over-the-air access directly from your Sony Ericsson mobile phone to Web server-based solutions by using the C++ or Java programming languages. Extending its support for Web services for the Java ME Platform to UIQ 3, Sony Ericsson Developer World introduces a new development tutorial and sample code for use together with the gSOAP open source toolkit. Web services have the power to offer vast, revenue-generating opportunities to the mobile industry by giving mobile users access to services like search engines, online auctions, bookstores, and gift shops as well as online payment services. Web services for the UIQ 3 software platform also make it easier to design sophisticated enterprise applications such as real-time mobile inventory systems or online banking services. The industry trend toward convergence has resulted in telecom operators looking to create value-added services that combine voice, video, and data services over fixed and mobile networks. Sony Ericsson has focused its efforts on removing the barriers to enter this highly lucrative market by enabling developers to use existing application programming interfaces (APIs) on today's phones to design not just client-specific applications but also advanced server-based Web 2.0 solutions, without worrying about interoperability between different platforms, networks, or service models. [www.sonyericsson.com](http://www.sonyericsson.com)

# Considering the SOA Reference Model

## Part I - Business Grounds

WRITTEN BY MICHAEL POULIN

➤ SOA RM: "...in SOA, services are the mechanism by which needs and capabilities are brought together"

**R**ecently OASIS voted the SOA Reference Model (SOA RM) into a standard. In spite of its high level of abstraction, this model emphasizes the business orientation of SOA. This two-part article will elaborate on the business aspects of the standard. Part 1 discusses one of the possible business grounds for SOA and design-for-changes uses cases. Part 2 will describe several technical design pillars of SOA in light of the standard.

### Business Agility

The SOA Reference Model (SOA RM) is the first standard that shifts SOA from a pure technical realm into the business world; you can see it right away from the SOA service description. You can wonder what "needs" and "capabilities" means; why an application in IT, which is considered to be an SOA service, becomes a mechanism; and, finally, where all the technical definitions of interfaces, clients, providers, QoS controls, etc., are. Don't worry, they're all in their places but now they have been repurposed toward business agility.

In particular, the SOA RM says: "the central focus of service-oriented architecture is the task or business function – getting something done." That is, in SOA, we're talking about a business function of the organization rather than about isolated business operations that implement the function today. The function is something the organization cannot exist without; the organization business model is the combination of such functions while the actual implementation of the function is the secondary category. Unfortunately, up 'til now, the majority of IT applications fit exactly into that category. Now SOA creates an opportunity for IT to become the business partner and perform as a function for its enterprise rather than just a support provider.

The SOA RM states that an SOA service operates in an "execution context," which is defined as a "set of technical and business elements that form a path between those with needs and those with capabilities." The standard expands the interpretation of "those with needs" and "those with capabilities"; "those" are not only IT applications and operations, they are the business elements as well, and, moreover, the business elements are first. If IT looks at its organization from the business perspective (versus operational/procedural ones), it can find that very few applications correspond directly to the core business tasks; other applications support a lot of just-this-moment operational requirements that were real years ago. If you add runtime and procedural application integration to this view, it becomes easier to see why IT has difficulties implementing and supporting frequent business changes required by the modern market.

Finally, when explaining how SOA differs from other models, the SOA RM outlines that it reflects business-motivating considerations that are expressed "within service descriptions and service interfaces." This is what SOA brings to the table the first time because a service description in the form of the service contract includes policies that "may also express business-oriented policies – such as hours of business, return policies, and so on." That is, some services in SOA elevate to the level of business entities. Now that it's more clear what "capabilities" are, we are saying that an SOA service can implement much more than the interface "API"; it's responsible for the functional and non-functional aspects, many of which might be invisible to the consumer though the service interfaces. Nonetheless, the invisible capabilities are just as important to the service consumer as the visible ones, because consumers now depend on the



honest behavior of the service outside of the consumer's control. For example, your enterprise service engages a third-party service to publish your results on the Internet. Would you like to use that service if you find that it doesn't filter porn ads and they get or can get published with your data?

You are right, not every SOA service is that complex and there are a lot of simple and very useful services that IT can offer. However, the power of SOA is in the services that implement business services and, accordingly, get involved in the risks of real business life. In this article, I will go a step further and elaborate on how a business may be decomposed into elements useful for building SOA service applications, and what the IT managers and architects have to understand and deal with if they really want to modernize IT to be agile in business. That is, the article tries to position SOA as a business-centric application architecture driven by business (not by technology) and organized in a way that is oriented to support business changes, both today and tomorrow.

### Business Model Decomposition

Since the SOA RM standard attempts to facilitate an enterprise technical architecture to mind real business functions, it would be worth finding out what they are. This way, one of the working assumptions we make is that every organization operates on the basis of its business model. A business model definition is still evolving; in this article, we'll use the one below:

*A business model is "a conceptual tool that contains a set of elements and their relationships and allows expressing the business logic of a specific firm. It is a description of the value a company offers to one or several segments of customers and of the architecture of the firm and its net-*

*work of partners for creating, marketing, and delivering this value and relationship capital, to generate profitable and sustainable revenue streams."*

A business model also has instrumental and structural elements. For example, distribution channels and revenue models are instrumental elements while value configurations (the configuration of activities and resources) and core capabilities (the capabilities and competencies necessary to execute the company's business model) are structural elements. Talking about structural elements of the business model, we can recognize Business Services and Business Processes, the compositions of which constitute core capabilities. Both Business Service and Business Process have internal structures as illustrated in Figure 1 and described below.

The functional organizational decomposition of an enterprise model helps in recognizing its Business Services and Processes. Each Business Service is unique in the organization but may formally be described in the same way as other services. In particular:

- **The foundation of a Business Service is data:** The data becomes business data when its business meaning is defined via business metadata. A Business Service specifies methods, activities and/or rules that might be applied to manipulate the business data. Business methods, activities, and rules may be grouped for cooperative execution in different scenarios. The latter have their own methods and rules for composing data processing activities; those compositions are usually called business operations and workflows. The execution of data manipulation methods, activities, and/or rules in business scenarios produces results. The results become business results when corresponding business metadata is defined. The Business Service frequently includes a delivery mechanism of the results to the business consumers although it is optional. (Other important elements of a Business Service such as documentation and audit rules and procedures are skipped here for the sake of simplicity.)

A Business Service consists of at least one business unit of work, which encapsulates some business data and data processing activities. The entities of the data processing activities are different heterogeneous "things" representing additional data, applications, communication processes, human

activities, information containers (e.g., a spreadsheet), and more. The business unit of work defines how a particular business task of the Business Service is implemented. The implementation is not crucial for the enterprise and is the subject of change based on solution availability (including technical ones) and market trends.

Business Service changes, in its core, are infrequent and usually reflect industry and corporate policies or corporate restructuring. Instead, the Business Service may be extended or split into several new Business Services. What can and may happen to the Business Service is defined in the enterprise business model.

A Business Process is defined in Wikipedia as:

*"a collection of related structural activities that produce something of value to the organization, its stakeholders or its customers. A business process can be part of a larger, encompassing process and can include other business processes that have to be included in its method. In that context, a business process can be viewed at various levels of granularity."*

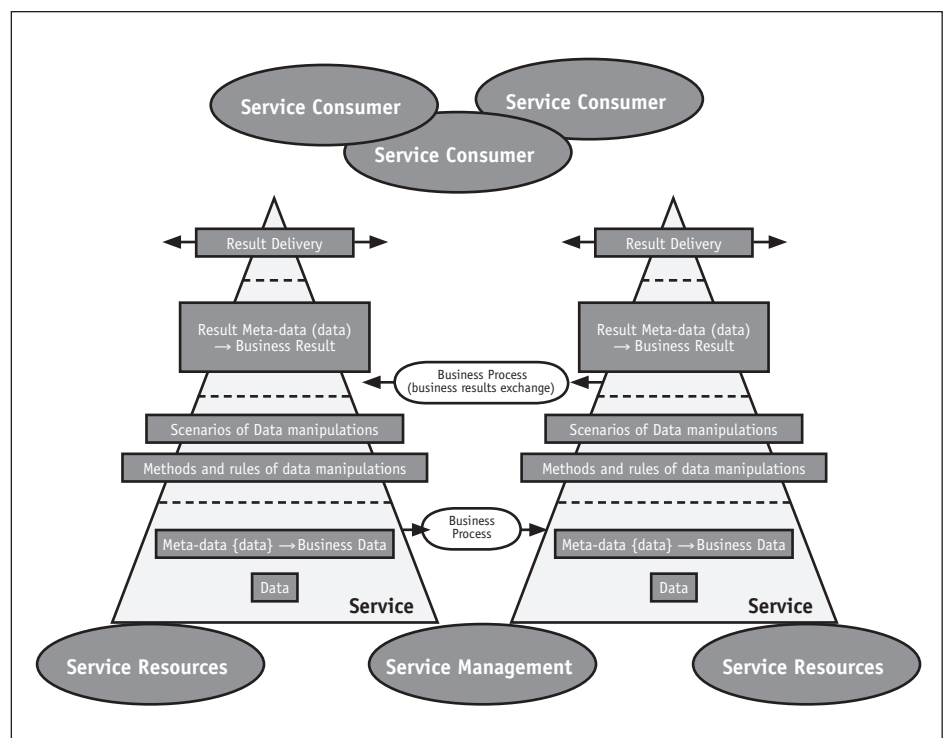
In the Business Process, Business Services interact with each other according to special methods, activities, and/or rules. The interactions may be viewed as an exchange

of the results (actual or logical) produced by the Business Services. That is, a Business Process consists of methods, activities, and rules for exchanging results between Business Services. Methods, activities and exchange result rules constitute the "things" the enterprise business usually changes to adapt to market needs. The richer the spectrum of changes available and the easier the changes are to do, define the market adaptability and business stability of the enterprise. SOA targets this exact area; however, without implementing Business Services (in addition to the technical services), SOA can't help in the Business Processes.

At the entrance point into a Business Service, the foreign results become new data and the Business Service's *stack* starts again. Sometimes, metadata transformation is needed to convert foreign result interpretation into the local one. The transformation may be done either by adding additional metadata (i.e., accepting metadata provided with the foreign results) or by replacing some or all of the foreign metadata with internal metadata – business data interpretation.

Thus, we can rephrase the goal of SOA:

- The creation and maintenance of such application architecture that can effectively support corporate Business Services and Business Processes with the highest effectiveness and can quickly adapt to the changes in them.



**Figure 1:** Business Services and Business Processes in the enterprise business model.

# The reality of the digital age is that your business is embodied in your technology

Thus, SOA may be viewed as a technical architecture built around an enterprise business model, not around isolated business procedures or just-this-moment operational needs. SOA is supposed to address current and upcoming business requirements, diversity, which is limited by a particular business model. If the business model is unclear in the organization, Services and Processes, SOA won't help but rather will confuse the company a lot.

## Design for Business Changes

Forrester Research states, "The reality of the digital age is that your business is embodied in your technology – you don't have a business until you have it implemented in your technology base, and your business can change only as fast as your technology can." SOA promises to make the changes faster and easier. Let's observe these changes.

Looking from the enterprise perspective, we can find three types of possible change in the business that the technology has to support:

- Changes in the business methodology of doing things (e.g., changes required by the Basel II regulations in financial credit management)
- Changes in the set of tasks performed by a business service and/or in a business process (e.g., adaptation to the market trends)
- Changes in the core of the business service, i.e., in the business model, dictated by the market dynamics (e.g., enterprise strategy change)

SOA can help IT in two ways by:

1. Offering services that can be easily combined in different compositions
2. Hiding change implementations behind the service interfaces

The first way is popular but it has significant pitfalls: to gain certain flexibility in the compositions, the services either have to be quite generic, which hits performance and risks losing control over details, or there should be a lot of fine-grained services

available, which makes management and maintenance difficult and you can end up with a significant ballast of in-definite non-reusable services.

The second way is the IT hope of SOA. However, those who hope are usually forgetting that the changes are not free due to the cost of retesting and revalidating all the dependencies (in the Service Contracts) and the actual support for multiple service versions – not all service consumers "are in need" of the changes at once.

The following examples illustrate the third way of designing SOA services – the design for changes. This approach is based on an a priori knowledge of the business model and its potential modernization. Certainly, this way does not help in converting the service to a completely new one, but it covers the majority of other, smaller change-cases.

## Access Control System

Assume we build an access control service based on roles. A user or subject in a role is permitted to operate on a resource – client's data – and perform the activities associated with or defined for the role. The role is equivalent to only one activity, e.g., "read," "write," or "edit."

Due to business responsibilities, many users may or have to perform several activities but individual role assignment demands costly operational support. The business requires reducing the cost of the access control management and maintenance.

## Traditional Application-Centric Design

IT decides to simplify management and maintenance by gathering users into groups and each group is assigned multiple roles. That is, all users in the group inherit all the roles from the group. The business objective of reducing costs is reached.

In a while the market has changed and the client's data has received new internal dependencies. For some users, their duties have become conflicted with the new data dependencies and those users had to be restricted from acting in one of the roles, but they still have to perform in others.

Since a group has been designed to provide inheritance of all the roles to all group members, the only possible way to restrict a user in the role is to create another group with a reduced set of roles and put those users into a new group. It's easy to imagine that if a group has a relatively small number of associated roles (three, for example), eventually we have to create seven groups to represent all possible combinations of the roles. Considering the realistic number of different clients and original groups in the firm, this becomes a maintenance nightmare. Plus, adding a new role to the initial group increases the number of group variations almost exponentially, and the operational cost rises correspondingly.

## SOA Design Style

We have started by studying the business model and found that users may be grouped (to make management cheaper) based on their business responsibilities that follow certain corporate policies. This means the grouping should be changed if user activities are in conflict with the policies. Another potential reason for the access change is a reassignment of the business responsibilities to a user. With this business knowledge, the access control service has to be designed in such a way as to accommodate access changes even if current business requirements don't say anything about it (which, actually doesn't mean there isn't any problem).

Instead of explicit role inheritance from the group to the users, each user becomes associated with a mask specified for the given group. The mask addresses all roles in the group and defines which concrete roles a particular user inherits from the group. As a default, all roles are inherited. If a user's access should be changed, the user's mask is modified and no new masks or groups are created. Plus, the modification may be based on corporate business policies. What we get is this: almost the same savings for the business but it works now and in the future.

This described case fits into SOA very well because the change may be done behind the service interface transparently for the service

# you don't have a business until you have it implemented in your technology base

consumers. However, if the service has not been designed for change from the beginning, it would be too risky adding the masks later (even behind the interface) because, in case of a mistake, it could create problems with resource access for many users and negatively affect corporate business service.

## Credit Risk Calculation Engine

A credit risk calculation engine calculates the risk exposures for certain types of financial transactions.

### Traditional Application-Centric Design

A credit risk calculation engine is designed as a self-contained financial application that:

- Utilizes financial transaction data from the operational data store with related details from the reference data, client reference, and money market data stores (supported by four different teams)
- Calculates risks according to the pre-defined formulas
- Stores credit exposures (results) in the result operational data store

The calculation engine team has a schedule of when to calculate intra-day and end-of-day risks and knows which data should be used in each case, when and how to check data accessibility in each of the data stores, which procedure to follow in the case of data store schema changes, which procedure to follow if security and compliance rules and regulations change, and when and what should be re-deployed in case of life-cycle changes in each of the consumer applications.

It's obvious the team has to perform a lot of activities that are unrelated to the credit risk calculation. Thus, any change in the calculation business rules or an appearance of additional calculations starts a laborious process of negotiations and synchronizations between the credit risk calculation engine team and all other resource and consumer teams, each of which has its own objectives and, sometimes, conflicting interests. This results in slow and sometimes a partial implementation of the change, i.e., higher cost and longer time-to-market.

## SOA Design Style

The credit risk calculation engine is designed as a service (CRCS) with a set of contracts with its consumers. There may be single contract for all or multiple contracts. The engine is obliged to perform credit risk calculations on the data that's totally described by certain metadata (descriptors) with predefined performances. This is it.

The CRCS receives a command from the Business Process Management (BPM) Service on when to perform this or that risk calculation task and which data to use. The CRCS engages the data access service to obtain all necessary data for the calculations. The exposures are returned to the data access service as well.

The CRCS doesn't know where the data comes from and where the results would go, and what policies are to be applied and how. If the CRCS gets household data instead of financial transaction data, it still has to be able to perform perfect calculations until the data meets the metadata definitions.

The CRCS team is only concerned with service performances, stability, reliability, and deployment/configuration manageability. It shares concerns of scalability with the Business Process Management Service because the latter may be able to invoke several CRCS, if needed.

Now, when a new type of financial transaction has to be processed in the CRCS (such a big change indeed), very little needs to be done. First, data for the new calculations should be provided by the Data Access Service in the form of a unified document-style data container. Then the CRCS has to add a new formula to the calculation module. If the module has an interface of the Command Pattern style, a new formula invocation isn't more complex than a new command request. Plus, the data for new calculations, provided by the data access service, is encapsulated in another unified document-style container.

Thus, without the knowledge of a real business service provided by the credit risk calculation function, it might be not obvious that the CRCS needed to be designed with the Command Pattern style and that the data needed to be encapsulated into

the document-style containers by the data carriers. In such a case, the processing of new financial transactions would require a lot of application and/or service interface modifications (including all dependents). You can imagine how costly it would be in production with many consumers in place.

## Summary

The SOA RM standard emphasizes the business aspects of SOA and provides a rich foundation for positioning enterprise architecture around the enterprise business model. This article has described one of possible interpretations of the enterprise business model in terms of business services and business processes. The observation was followed by two examples of how a SOA service may be designed for these changes based on the understanding of the enterprise business services and processes and their business scope. Indeed, if we have an enterprise architecture that's focused on real business functions (rather than on operations), why don't we take advantage of it? ■

## Resources

- **Reference Model for Service Oriented Architecture 1.0. OASIS:** <http://www.oasis-open.org/committees/download.php/19679/soa-rm-cs.pdf>
- **Ostenwalder, Y. Pigneur, and C.L. Tucci. Clarifying Business Models: Origins, Present, and Future of the Concept. Communications of AIS, Volume 15, Article 1:** <http://www.businessmodeldesign.com/publications/Preprint%20Clarifying%20Business%20Models%20Origins,%20Present,%20and%20Future%20of%20the%20Concept.pdf>
- **Digital Business Architecture: Harnessing IT for Business Flexibility. Forrester Research:** <http://www.forrester.com/ResearchThemes/DigitalBusinessArchitecture>

## About the Author

Michael Poulin works as a consulting technical architect for leading financial firms. He is a Sun Certified Architect for Java Technology. For the past several years, Michael has specialized in distributed computing, SOA, and application security. [m3poulin@yahoo.com](mailto:m3poulin@yahoo.com)

# Open SOA Collaboration

## The birth of another standard

WRITTEN BY DAVID S. LINTHICUM

➤ Last month an alliance of leading vendors announced progress on specifications to define a language-neutral programming model for application development in SOA environments. They call this specification Open SOA Collaboration. In essence, they are proposing a new standard to create and manage IT, making the process of integrating different third-party SOA technologies “less onerous,” they say. Or, we can call this a standard way of delivering services, making it easier to work and play well together.

**S**o, who's in the gang? BEA, IBM, Oracle, and SAP first got together last November to begin work on the common programming model, along with Iona, Sybase, Xcalia SA, and Zend Technologies Ltd. Others are joining the mix, including Software AG and Red Hat.

This group has concentrated its efforts on two projects – service component architecture (SCA) and service data objects (SDO). If this sounds familiar, it is. We've seen this type of standard with components, distributed objects, and, more recently, Java.

SCA is looking to provide a model for creating service components in a wide range of languages and a model for assembling service components in a business solution. In essence this is a standard that defines how services are created so they interact with each other without a lot of customization. This will benefit those who are looking to create composite applications that use these services.

SCA encourages an SOA organization of business application code based on components that implement business logic. It offers capabilities through services that consume functions offered by other components through services called references. SCA divides up the steps in building a service-oriented application into two major parts: The implementation of components that provide services and consume other services and the assembly of sets of components to build business applications by wiring references to the services.

SCA stresses decoupling the service implementation and service assembly from

the details of the infrastructure capabilities and the access methods used to invoke services. SCA components operate at a business level, according to the spec.

SDO is looking to provide a consistent way of handling data in applications, whatever its source or format may be. Okay, that would be data abstraction. Moreover, SDO provides a way to unify data handling for databases and services.

It's clear that SDO is designed to unify the way in which SOA applications handle data. Using SDO, application programmers can uniformly access and manipulate data from heterogeneous data sources, including relational databases, XML data sources, Web Services, and enterprise information systems.

SDO is based on the concept of disconnected data graphs or a collection of tree-structured or graph-structured data objects. Under a disconnected data graphs architecture, a client retrieves a data graph from a data source, mutates the data graph, and then applies the data graph changes back to the data source.

Databases are connected to the applications by data mediator services. Client applications query a data mediator service and get a data graph in response. Client applications send an updated data graph to a data mediator service to have the updates applied to the original data source, and this architecture allows applications to deal principally with data graphs and data objects.

New? No. Interesting? Sure. We've seen these types of standards before with the rise of



client/server, CORBA, and Java, all looking to provide standard mechanisms for developing SOA, or, the way we bind all of these things together to form applications. The SDA concept especially has been done to death, with some successes and some classic failures.

As always, the real battle to be won here is the developer's acceptance of these standards. For that, the vendors have to work together to implement the standards in the very same way...something that's been tough to do in the past. So they'll have to put aside their desire to stand out and focus on being the same...an unnatural act for most.

It will also be interesting to see where this standard goes in the context of BPEL and other standards that provide the same solution patterns. At the end of the day, standards are only useful if there's one for each problem pattern. So far in the world of SOA, we have three or more standards for each problem pattern. Those who consume the technology won't touch standards until the problem is solved. Once bitten, twice shy. ■

### About the Author

David S. Linthicum (Dave) is an internationally known application integration and service oriented architecture (SOA) expert. In his career Dave has formed many of the ideas behind modern distributed computing including EAI (Enterprise Application Integration), B2B application integration, and Service Oriented Architecture (SOA), approaches and technologies in wide use today. Currently, Dave is the CEO of the Linthicum Group®, LLC, ([www.linthicumgroup.com](http://www.linthicumgroup.com)) a consulting organization dedicated to excellence in SOA product development, implementation, and strategy. [david@linthicumgroup.com](mailto:david@linthicumgroup.com)

SOA  
MAKE YOUR ^ SECURITY MOVES WISELY...



**XWALL**

WEB SERVICES  
FIREWALL



**XRAY**

WEB SERVICES  
DIAGNOSTICS



**VULCON**

VULNERABILITY  
CONTAINMENT SERVICE



**SENTRY**

SOA SECURITY  
GATEWAY

PUTTING TOGETHER THE PIECES FOR THE WORLD'S MOST DEMANDING SOA SECURITY SYSTEMS

## FORUM SYSTEMS ENTERPRISE SOA SECURITY SOLUTIONS:

- ▶ TRUSTED SOA MIDDLEWARE
- ▶ WEB SERVICES SECURITY
- ▶ XML ACCELERATION

W W W . F O R U M S Y S T E M S . C O M



**FORUMSYSTEMS**

THE LEADER IN WEB SERVICES & SOA SECURITY

# Complex and evolving systems are hard to test...



## Parasoft helps you code smarter and test faster.

Start improving quality and accelerating delivery with these products:

Awarded  
"Best SOA Testing  
Tool" by Sys-Con  
Media Readers

**SOAtest™**

InfoWorld's 2006  
Technology of  
the Year pick for  
automated Java  
unit testing.

**Jtest™**

Automated unit  
testing and  
code analysis  
for C/C++ quality.

**C++test™**

Memory errors?  
Corruptions?  
Leaks?  
Buffer overflows?  
Turn to...

**Insure++™**

Easier Microsoft  
.NET testing by  
auto-generating  
test cases,  
harnesses & stubs

**.TEST™**

Automate  
Web testing  
and analysis.

**WebKing™**

 **PARASOFT®**

*We make software work.™*

Go to [www.parasoft.com/WSJmagazine](http://www.parasoft.com/WSJmagazine) • Or call (888) 305-0041, x3501