SYS-CON

THE WORLD'S LEADING MAGAZINE DEDICATED TO WEB SERVICES TECHNOLOGIES

FOR ALL THE LATEST NEWS AND INFORMATION VISIT:

www.SOA.SYS-CON.com

SOAWORLD™ MAGAZINE

AUGUST 2007 / VOLUME: 7 ISSUE 8

Service Provisioning via SPML in SOA

PLEASE DISPLAY UNTIL OCT 30, 2007

$6.99US $7.99CAN

08

0 71486 03420 9

_INFRASTRUCTURE LOG

_DAY 68: The business climate is constantly changing. Our IT environment is completely rigid. We can't align IT to meet the larger business needs. I told Gil we need an SOA so we can be proactive for once.

_Gil had an idea. He brought in contractors and made the entire office "modular" and "flexible." Gil, I am not a hamster.

_DAY 70: This should free us up: IBM SOA Solutions built with IBM WebSphere®, the leading integration platform. Now we have the hardware, software and services for a flexible IT infrastructure. IBM has helped 3,600 companies implement an SOA. And getting started was easy. Our business is built for change.

_I don't have to crawl with my coffee anymore. It's great.

IBM.COM/**TAKEBACKCONTROL**/FLEXIBLE

Child's Play

WRITTEN BY **SEAN RHODY**

One of my friends has a child who used to bring a blanket with her wherever she went. It didn't matter that it was a hundred degrees outside; she carried it more for comfort than to keep herself warm and safe. Not that it protected her in reality, but it provided the illusion of safety, which is often as important.

When Web services first burst onto the scene, which in my mind was the beginning of the SOA movement, one of the biggest challenges faced by early implementers was the perceived lack of security. Fear and uncertainty abounded, and it was years before the majority of IT organizations became comfortable with the level of security that could be provided.

What I find interesting at this juncture, when SOA is now a fairly well-established architectural paradigm, is that in many ways Security is the security blanket (you had to know this was coming) of SOA.

In looking at implementations of SOA and working with various organizations that are doing the implementations, I've begun to question exactly how vital security is in the overall scheme of things.

Don't get me wrong, I'm a proponent of security and I take it seriously – I'm not suggesting that security is unnecessary, or even that it's just a "nice to have." Far from it. At the same time, I think there's a difference between applying every security concept on the planet in the paranoid hope of keeping data "safe" and the intelligent application of concepts at the appropriate levels.

I've seen systems in the past that were brought to their knees by the improper application of security concepts. SOA implementations are equally vulnerable to poor implementations or improper use of techniques. Indeed, the nature of SOA makes it even easier for a mistake to cripple a service or even the entire architecture. All it takes is a service whose usage grows faster than expected to outstrip the capacity of a poorly planned security scheme and bring an organization to a crashing (and I do mean crashing) halt.

Sense, common or otherwise, must be applied to the design of a service-oriented architecture, especially with respect to security. If you've already authenticated a user, do you need to reauthenticate for every call? Does every field have to have security, especially if the service is for internal use only (or some other use with similarly limited vulnerability)? These and a host of other similar questions must be asked, and re-asked periodically, in order to ensure that the proper application of security will occur.

It's also useful to analyze the composition of a business process for just where security needs to be enforced. A common tendency is to consider every service as the proper level of granularity for enforcement. At the surface, approaching each service as the place to implement security seems reasonable, and many times it is. But some services are never called directly, or some convey information that has no value outside the context of the larger process. It may be sufficient to enforce security while entering the business process, rather than enforce it repeatedly at each step of the process by checking each and every service. A blind adherence to guidelines is very similar to carrying that blanket – it provides the illusion of security, without any real protection.

Security is a vital component of SOA; there should be no question about that. The ability to protect data, transactions, information and even networks from malicious attacks from without or from within is critical to many SOA implementations. But it must be applied intelligently, not blindly, for it to be an effective component and not simply a performance detriment. ■

About the Author

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

SOA WORLD MAGAZINE
www.SOA.SYS-CON.com

EDITORIAL OFFICES
SYS-CON MEDIA
577 CHESTNUT RIDGE ROAD, WOODCLIFF LAKE, NJ 07677
TELEPHONE: 201 802-3000 FAX: 201 782-9637
SOA World Magazine (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:
SOA World Magazine, SYS-CON Publications, Inc.
577 Chestnut Ridge Road, Woodcliff Lake, NJ 07677

- **Immerse yourself in SOA** instead of dealing with tools that tack SOA features onto legacy tooling. With the Toolkit you don't have to deal with the impedance mismatch between your development habitat and the SOA paradigm because none exists.
- **Define and update** Web service interfaces in a quick, collaborative manner. Each time an interface is introduced or revised, a prototype implementation becomes available at the click of a button.
- **Generate XML** documents for requirements or testing by building intelligence into the document definition itself, what we call an "executable" document. No more dealing with blunt instruments like templates or DOM.

NEXT GENERATION WEB SERVICE TOOLING

Announcing The Delve® SOA Fast-Prototyping Toolkit

Web service prototyping, XML document generation, and scripted service testing, all with the world first commercial service-oriented programming language.

Delve® Integrates a code editor, run-time environment, specialized Web server and SOAP stack in a single, simple application.

Supports SOAP/XPath/XML-Schema/XML

CretaceousSoftware™
Evolving SOA

Download the Delve® SOA Fast-Prototyping Toolkit (Beta1.0)

www.cretaceoussoftware.com

Where Have All the SOA Standards Gone?

They've become a lonely lot

WRITTEN BY **DAVID S. LINTHICUM**

To mark a new standard in the SOA space, I create a Google Alert and sift through the pile of links returned to get the scope of its maturation. I'm currently tracking over 60 standards, starting with SOAP and XML (XML happened way before Google was cool).

Lately I've noticed a drop in the number of blogs, links, and articles talking about particular SOA standards. Where I once got dozens of links a week on some standards, I now get only one or two or none. So, I'm thinking that standards, although around, aren't as cool as they once were, and maybe people are a bit confused by the alphabet soup out there.

Standards have changed. At first they were good thoughts about a single way to do something, so everyone could mix and match solution patterns. They are now more marketing hype than anything else. In essence, if you're building a SOA product, make sure to start with a WS-something, and you can prove that you're a standard, and standards are always desirable. Right?

Sure. However, the number of WS-* standards out there today are downright scary. People who are just figuring out what the heck SOA is don't want to walk through a maze of standards. Moreover, vendors have done a less than stellar job of promoting standards. They typically do more of a sell job than an education job...there's a difference.

At first I figured it was just a cyclical thing, but I think it's a real trend. The press, bloggers, and even the SOA companies themselves are getting weary of the number of SOA standards that compete for our hearts and minds, and they are just not paying as much attention to them these days. Again, perhaps it's a matter of bandwidth. There's just too much out there to pay attention to, so things are falling off the truck.

Even at the seven SOA and enterprise architecture conferences I spoke at earlier this year, it was clear that interest in standards isn't driving SOA. Instead, the notions of architecture modernization and agility are really in the forefront these days. While I don't think anyone doesn't consider standards when implementing a SOA, the notion of standards doesn't seem to be driving the decisions as it once did.

To my point above, I think this is silent pushback from the years when the creation of SOA standards was more about marketing than coming together on the implementation of technology. I think users have figured that out. Considering the sheer number of standards out there that the vendors are asking you to follow and support, many find it far too self-serving and confusing. And I'm not sure I can blame them.

Don't get me wrong. Doing things in a standard way, using standards, is very important. Investment in standards could indeed lower some risks in the implementation and the maturation of an SOA. However, while rallying around a few key standards is certainly important, you simply can't figure out how 60 or more, some of them competing, will fit in your SOA. Too much, too early, too confusing.

It's time to normalize the number of standards out there. Remove the redundant standards, and remove the ones that aren't likely to have value anytime soon. They can be reintroduced later, as the need arises. Those who create standards need to be very clear about how they fit into SOA problem domains, and get out of the "We do it all" mentality. Each technology and standard is a part of a SOA, and fitting things together is the job of the architect. ■

About the Author

David S. Linthicum is an internationally known application integration and Service Oriented Architecture expert. In his career Dave has assisted in the formation of many of the ideas behind modern distributed computing including Enterprise Application Integration, B2B Application Integration, and Service Oriented Architecture, approaches and technologies in wide use today. Currently, he is CEO of the Linthicum Group, LLC, (www.linthicumgroup.com) a consulting organization dedicated to excellence in Service Oriented Architecture planning, implementation, and strategy.

david@linthicumgroup.com

SOA WORLD MAGAZINE
www.SOA.SYS-CON.com

CORPORATE

President and CEO
Fuat Kircaali fuat@sys-con.com

Senior VP, Editorial & Events
Jeremy Geelan jeremy@sys-con.com

ADVERTISING

Senior VP, Sales & Marketing
Carmen Gonzalez carmen@sys-con.com

Advertising Sales Director
Megan Mussa megan@sys-con.com

Associate Sales Manager
Corinna Melcon corinna@sys-con.com

SYS-CON EVENTS

Event Manager
Lauren Orsi lauren@sys-con.com

Event Associate
Sharmonique Shade sharmonique@sys-con.com

CUSTOMER RELATIONS

Circulation Service Coordinators
Edna Earle Russell edna@sys-con.com
Alicia Nolan alicia@sys-con.com

SYS-CON.COM

VP information systems
Bruno Y. Decaudin bruno@sys-con.com

Consultant 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

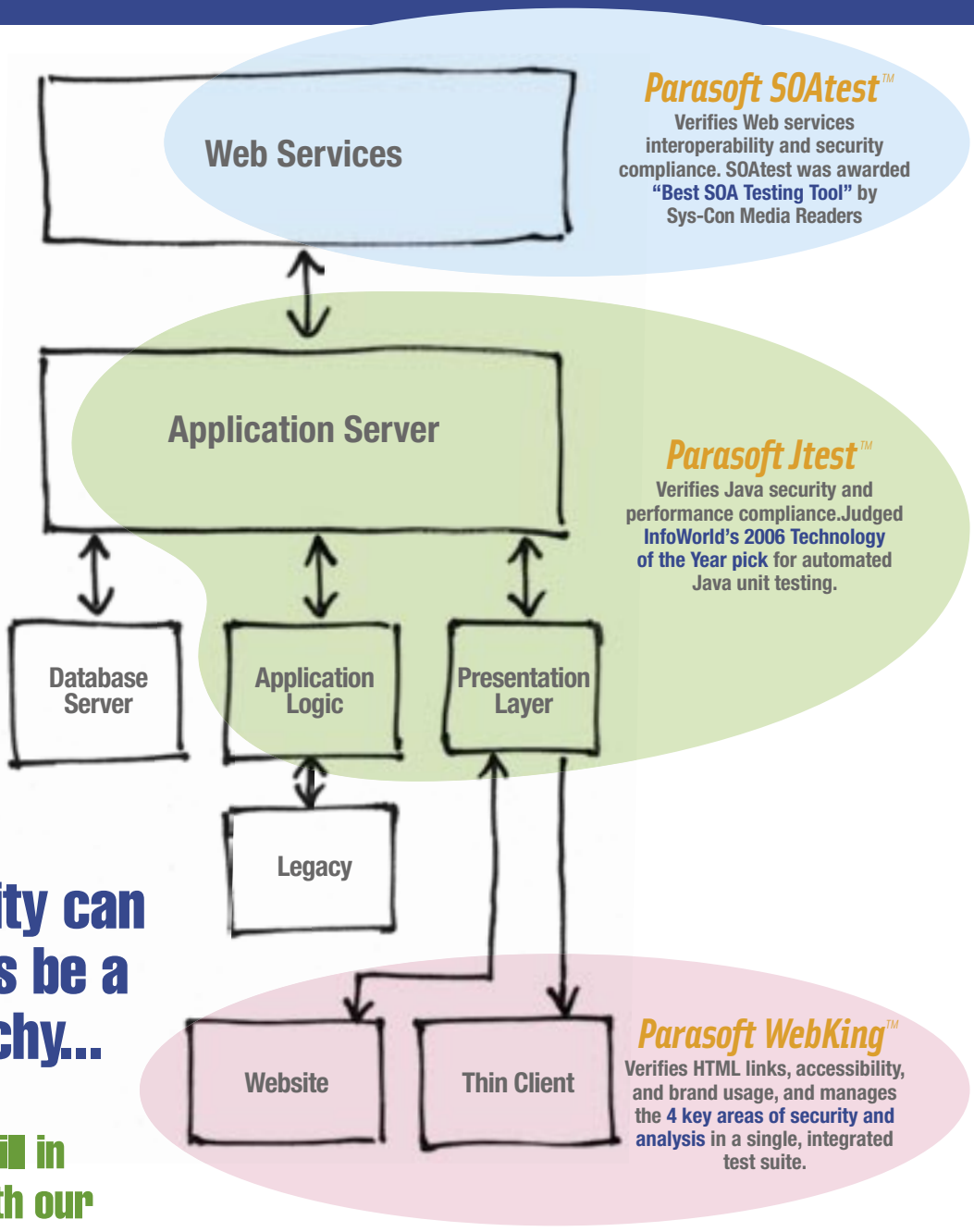
Accounts Payable
Betty White betty@sys-con.com

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



Improving productivity can sometimes be a little sketchy...

Let Parasoft fill in the blanks with our Web productivity suite.

Parasoft products have been helping software developers improve productivity for over 20 years.

Jtest, WebKing, and SOAtest work together to give you a comprehensive look at the code you've written, so you can be sure you're building to spec, the new code doesn't break working product, and any problems can be fixed immediately. Which means you'll be writing better code faster.

So make Parasoft part of how you work today. And draw on our expertise.

PARASOFT®
We make software work.™

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

Service Provisioning via SPML in SOA

Simplifying identity and resource management for distributed services

WRITTEN BY **MANIVANNAN GOPALAN**

Provisioning is the automation of all the steps required to manage user accounts or system access facilities or data relative to electronically published services.

The Provisioning Services Technical Committee (PSTC) at OASIS, the premier standards body for SOA-related standards, defined an XML-based framework named Service Provisioning Markup Language (SPML) for exchanging user information, resource information, and service provisioning information in systems. In this article, we'll explore the role of SPML in managing identity and resource information in SOA environments.

What Is SPML?

SPML is an XML-based request response protocol that is used to integrate and interoperate service provisioning requests. The use of SPML is to enable organizations to set up interfaces for Web Services and applications quickly and securely. This is done by letting portals, application servers, and service centers generate provisioning requests in and across organizations.

If you take a typical SOA security stack, SPML satisfies a complementary requirement for authentication, authorization and fine-grained access control. SPML is used for service provisioning whereas the authentication and authorization of data is done through SAML. Fine-grained XML access control is done through XACML.

Identity Management and SPML's Role

Nowadays user credentials play an important role, be it a network-oriented system or a specific application. Managing user identity is challenging in today's environment given the increasing diversity and complexity of systems. Identity management refers to the management of the entire lifecycle of one or more identities, from creation to destruction, and managing privileges.

SPML deals with provisioning these identities in enterprise ecosystems. It brings standardization in preparing system infrastructure to accomplish business activities. A typical SPML use case scenario in organizations is the situation of hiring a new employee, which involves lots of procedures that can be included in a provisioning workflow. Provisioning involves both digital as well as physical activities. A physical activity involves procuring a PC or laptop and a digital activity involves creating a user account in various applications.

SPML in Enterprise Identity Management

The Different Components of an Enterprise Provisioning System

The typical provisioning system contains three essential components: a Requesting Authority (RA), a Provisioning Service Provider (PSP), and a Provisioning Service Target (PST). This is represented in Figure 1.

- **Requesting Authority (RA):** In a typical provisioning system the RA is the client. Well-formed SPML documents are created by the RA

and are sent to the SPML service point, which is basically a Provisioning Service Provider (PSP). These requests describe an operation to be performed at the PSP end. For an RA to issue a request to the PSP, a trust relationship must exist between the RA and PSP. Sometimes the PSP can act as the RA for another PSP

- **Provisioning Service Point (PSP):** This is the component that listens to the request from the RA, processes it, and returns a response to the RA. Any component that listens and processes well-formed SPML documents is called a Provisioning Service Point.
- **Provisioning Service Target (PST):** The Target is basically actual software or an application on which action is taken. For example, it could be a directory that stores all of an organization's user accounts, or it could be an asset allocation system used to log requests for acquiring IT assets like laptops/PCs.

A typical provisioning system using SPML has one Requesting Authority with an PSP in the middle and one or more PSTs. Suppose there are three systems. Without using SPML the user information would have to be keyed into all three systems using the system portal. User information like name, address, contact number, date of birth, and SSN would have to be keyed in repeatedly across the three systems. By introducing a ProvisioningServiceProvider (PSP) layer and using SPML the user information can be keyed into a single Requesting Authority and be reflected across multiple targets. So we avoid keying the same set of information into various systems.

Operations Supported by SPML

SPML 2.0 supports various core, search , batch as well as async operations related to provisioning.

SPML Core Operations

- **list Targets:** to find the list of existing target (PST) systems supported by PSP
- **add:** to add an object to a given PST system
- **modify:** to modify an object in a given PST system
- **delete:** to remove an object from a given PST system
- **lookup:** to obtain an XML representation of an object from a given PST system

SPML Search Operations

- **search:** to get all the objects that match specified selection criteria (query)
- **iterate:** to get the next set of objects from the result set that the provider selects for a search operation (using selection criteria)
- **closeIterator:** to tell the provider that the requestor has no further need of the search result that a specific iterator represents

SPML Batch Operations

- **Batch:** to combine any number of individual requests into a single request

SPML Async Operations

- **Cancel:** To enable a requestor to stop the execution of an asynchronous operation
- **Status (Async capability):** To enable a requestor to determine whether an asynchronous operation has successfully completed or has failed or is still executing.

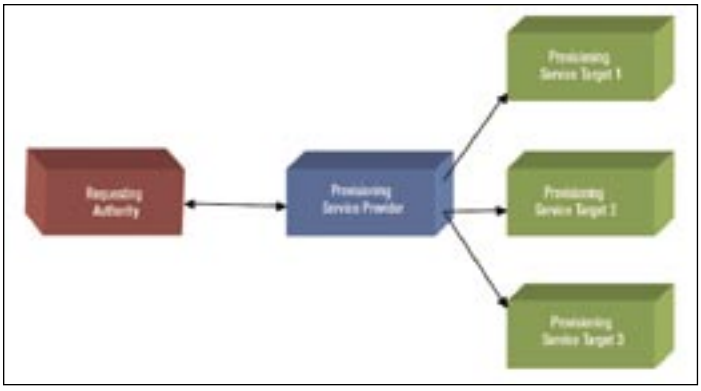


Figure 1 A provisioning scenario with multiple targets

Problems with Provisioning

So a typical provisioning system consists of requesting authorities, a provider, and a target. Before provisioning, the Requesting Authority might use its own portal to update the user information. A typical problem with this kind of system is that it might already be in place and a lot of user information might have been keyed in for a particular target. Now after developing a new provisioning system and putting it in place, the user information might not be there in the audit details of the provisioning system.

Provisioning can be done for different targets at the same time. But doing this makes it difficult to synchronize the data unless you pass the data through the provisioning service provider for the different Requesting Authorities and multiple targets.

Use Cases of SPML

Some typical use cases of SPML will be explored in the sections:

1. A mass federated identity use case, and
2. Partner credential provisioning

Mass Federated Identity

Federated Identity is a process of user authentication across multiple systems. Data lying across multiple identity management systems are joined together by using a user name that is the common identity. In a Federated Identity system, agreements are established among different service providers. The agreements can be based on policy.

Mass Federated Identity describes the technologies and use cases that enable the portability of identity information across autonomous domains. A typical use case on Mass Federated Identity is cross-domain user provisioning.

Suppose there are three different systems, System A, System B, and System C as represented in Figure 2. These three systems are basically service providers. If a user is authenticated by one service provider he'll be authenticated across all the providers by the presence of a virtual identity domain.

The federation of isolated identifier domains gives the client the illusion that there's a single identifier domain. The user can still hold separate identities with each service provider. However, he doesn't necessarily have to know or possess them all. A single identifier and credential is sufficient for him to access all services in the federated domain. An SPML-based approach can streamline the basic task for user identity provisioning in such identity federation systems.

Partner Credential Provisioning

Partner Credential provisioning involves sharing information

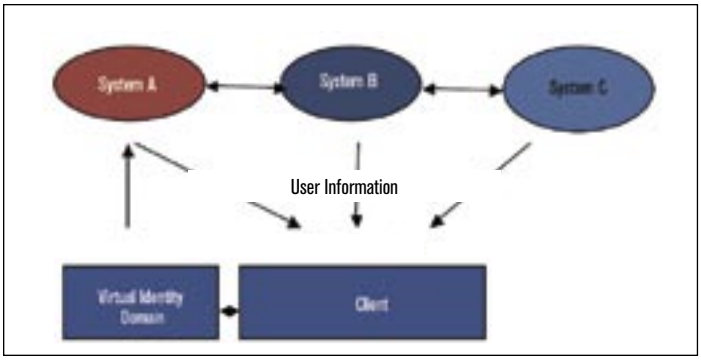


Figure 2 Mass federated identity

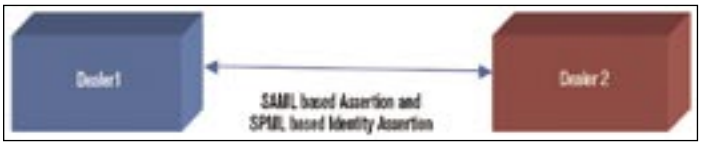


Figure 3 Partner credential provisioning

among business partners. With Auto Industry as a domain, a typical partner credential provisioning scenario is explained below. There are geographically spread-out dealers who basically supply the cars/trucks manufactured. Each dealer will typically use its own credential verification mechanism.

Now suppose there are two dealers, Dealer 1 and Dealer 2 as represented in Figure 3. Dealer 1 will have a system to create, delete, and update user information. Dealer 2 will have a different user identity system. The information in the system of both dealers is secured and confidential. Access rights are restricted. By using SPML we can get whole dealer systems provisioned with user information that gets reflected in both dealer systems. But Dealer 1 might have data that's confidential and doesn't want Dealer 2 to access it. Similarly Dealer 2 will have data that's secured. Both of them can incorporate Security Assertion Markup Language (SAML) along with SPML.

SAML will provide the Authorization decision Assertion and SPML will provide Identity Assertion. Using SAML we can enforce policies so each dealer will have specific access rights.

Related Standards

While SPML is a standard meant to simplify the problem of provisioning user credentials, it's related to similar standards-based approaches from other domains. We'll explore a few of the related approaches here.

WS-Federation Provisioning Component

A WS-Federation Provisioning component is a federated identity management component used to simplify the work of professionals as they seek to cut the cost and complexity of passing identity credentials across organizational boundaries. Various companies, including Microsoft and IBM, have come up with a WS-Federation specification as part of the WS-Security specification.

Federated Identity infrastructure provides cross-company identity sharing, cross-boundary single sign-on, and dynamic user provisioning. WS-Federation provides options to build new Web Services architectures.

The benefit of using a WS-Federation Provisioning component is its improved security features. It has an automated de-provisioning facility for external user access. A valid security token will be issued by the company to authenticate locally before accessing the partner credential.

Liberty ID-WSF Provisioning Component

The Liberty Alliance Provisioning component is to enable networked applications based on open standards where consumers and companies can make online transactions while protecting the privacy and security of identity information. In the present scenario, devices and identities of all kinds are linked by federation and protected by authentication being built today with Liberty's open identity standards, business and deployment guidelines, and best practices for managing privacy.

The Liberty ID-WSF Provisioning Service (ProvS) is the entity that distributes the data and potentially executable code to the client platforms. ProvS also provides a control point for the lifecycle management of provisioned modules (PMs), supporting operations such as update, delete, activate, and deactivate.

Project Concordia

The Concordia Project is aimed at providing interoperability among identity management systems. It's an organizationally independent global initiative consisting of representatives from CardSpace, Liberty Alliance, OpenID, openLiberty.org and the open source, SAML 2.0, and WS-Federation communities.

Conclusion

SPML can be used for provisioning identity and other service-related information in a variety of use cases. User information provisioning, partner credential provisioning, and device provisioning are some of the typical use cases involving SOA that can benefit from using SPML. The role of SPML in SOA is basically to provide a provisioning layer for identity and related information between different SOA participants. The problem of authenticating provisioning requests is resolved through SAML. SPML is the de facto standard for implementing an interoperable provisioning standard in SOA though some related industry efforts are working along similar lines. A concerted effort should be taken to enhance the penetration of SPML and reduce duplicated efforts.

References

1. *SPML FAQ*. http://www.openspml.org/spml_faq.html.
2. *About SPML*. http://www.networkworld.com/details/5623.html.
3. *SPML Overview*. http://en.wikipedia.org/wiki/SPML.
4. *XML-based Provisioning Services*. http://xml.coverpages.org/provisioningServices.html.
5. *Kim Cameron's Identity Web Blog*. http://www.identityblog.com/?p=771.
6. *SPML and its importance in the Security Infrastructure Framework for e-Business*. http://www.omg.org/interop/presentations/2002/Gavenraj_Sodhi.pdf.
7. *Concordia Project*. http://projectconcordia.org/index.php/Main_Page.

About the Author

Manivannan Gopalan is a technical specialist with the Web Services Center of Excellence at SETLabs, Infosys Technologies, Bangalore. He specializes in legacy systems, legacy migration to SOA, and Web Services. He has worked on Web Services primarily in service-enabling legacy system. He has also worked in SPML and published papers at international conferences such as the IEEE International Conference of Web Services and journals.

Manivannan_gopalan@infosys.com

Why is the WSO2 ESB the hottest ESB out there?

... its two products in one!

Full featured ESB

- Proxy services - transport (HTTP/S, JMS, SMTP), interface (WSDL/Schema/Policy), message format (SOAP/POX), QoS (WS-Security/RM) and optimization switching (MTOM/SwA)
- Integrated Registry/Repository, facilitating dynamic updating and reloading of configuration and resources
- Highly flexible, supports XSLT, XPath, Java, Ruby, Javascript

High performance XML Gateway

- Load-balancing/fail-over and throttling support
- Enforce WS-Security through policies, or SSL offloading for SOAP
- Validate, transform and route messages
- Non-blocking http/s transport for ultrafast execution and support for a large number of connections

WSO2 Enterprise Service Bus

Simple to get started and use

- Ships with many end-to-end samples that work out-of-the-box and comes complete with sample client and service code

Web based administration

- Easy to configure even complex standards like WS-ReliableMessaging through simple check boxes

Full XML support

- Built-in support for XML, Namespaces, XPath, XSLT and XOP

Support for non-XML messages

- Supports binary messages, as well as pure text or binary JMS messages etc.

Extensible

- By using Java/Spring classes or BSF scripting languages such as Javascript, Ruby, Groovy etc.

Multi protocol

- Java NIO based non-blocking http/s, JMS, Mail, File Apache VFS files systems such as - S/FTP, Zip, gzip, local, http/s..

WSO2
Oxygenating The Web Service Platform

http://wos2.com

'Delving' into Cretaceous Software's SOA Fast-Prototyping Toolkit

A service-oriented programming language

WRITTEN BY **PAUL T. MAURER**

Dynamic languages like Ruby and Python have been enjoying a burst of popularity in the Web development community and there are a plethora of frameworks for those platforms that allow them to solve a wide variety of problems. There's one company that's building its own dynamic language from the ground up with a single-minded approach to the problem of fast prototyping for Service Oriented Architectures.

That company is Cretaceous Software, Inc., and the programming language is called Tectonic. Cretaceous characterizes Tectonic as a service-oriented programming language because it "directly reflects the concepts and values that embody SOA." What does this really mean? The answer is probably best illustrated by the example in Figure 1.

The top window in the figure shows the seven lines of Tectonic code required to implement a simple Web Service called "GreetingService" with the operation "Greet." The bottom window of the figure shows the two lines of Tectonic code required to exercise the service. Granted the example is contrived but it serves to illustrate the simplicity of delivering a simple service. No annotations are required, no embedded angle brackets, no code generation, and no special widgets of any kind. The notion of a service and all that it entails is embedded in the language thereby providing a low-impedance mechanism for prototyping services.

Delving In...

You may ask, "How do you run this code?" or "How is it deployed?" The answer is in the Delve Development Habitat, which is essentially an integrated development environment and runtime specifically designed for the Tectonic language. Delve provides an editor, language interpreter, embedded Web server, and complete SOAP stack and is designed around the same SOA fast-prototyping concepts as Tectonic.

To run the example in Figure 1, I typed the code into two separate pages (more on this later) and executed the project. Delve automatically exposed the Web Service on an embedded default server then ran the test client and displayed the results.

The WSDL for the service is available via the Web Service URL with a ?WSDL parameter appended. To test interoperability, I pointed my NetBeans IDE at the Delve hosted service and generated a JAX-WS client that ran and returned data from the service without a hitch.

To begin working with Tectonic you create a project then create a new "page" and put Tectonic code in that page. Tectonic has no concept of a file, and the Delve Habitat doesn't explicitly support mapping of pages or projects to files. You may think of each page as a file but there's nothing in the environment that guarantees this or gives you direct access to a file.

There's no restriction as to what may be put on each page but I'm told it's considered good form to distribute code across pages in a logical manner. For example, a service definition, with its supporting types and schema, should reside on a single page. Each page is assigned a name by the user at creation, and that name should reflect the purpose of the page. Pages are ordered in a project in the order they should be executed.

When I installed the Delve Habitat, it automatically configured

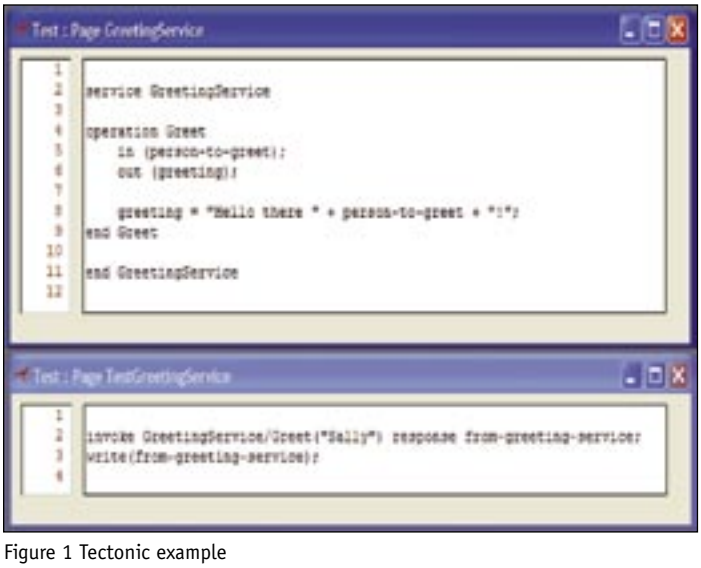


Figure 1 Tectonic example

a server on the standard loopback address, 127.0.0.1. This server is referred to as the "default server." Additional servers can be configured to run in the Delve Habitat on any IP addresses and ports required. Each server is named and can be configured to start automatically when Delve is launched. Servers are visually displayed in a Server Panel in the Habitat and each service deployed is displayed under the server. By default the URL suffix for each service is the project name followed by the service name, but Delve lets you customize it to whatever you want.

Schema

The Tectonic language allows for the definition of XML Schemas. This is natively supported and acts as nice shorthand for constructing schemas without additional tools. In fact, all type definitions in the Tectonic language are associated either explicitly or implicitly with a schema.

A nice feature of the Delve habitat is that it directly supports XML namespaces. The habitat lets users define a common set of namespace mappings that are shared across all code executing in the habitat. This reduces the boilerplate namespace declarations when working across a number of projects.

XPath

So now that you've imported or constructed a schema in Tectonic, how do you navigate the document hierarchy? You might expect a typical dot (.) notation for member access as in other languages, but Tectonic uses XPath notation natively for its member access mechanism. The forward slash "/" character is used as an "accessor" operator. As you would expect with XPath, when the forward slash is placed to the right of an element instance, it produces all child elements with names matching the operand appearing to the right of the accessor.

Resources

Since Delve doesn't have a notion of file, it refers to external entities such as XML documents and schema files as "resources." The resource construct lets a Tectonic developer ignore the physical aspects of the resource. The developer only has to treat the resource abstractly as a document and not be concerned with whether or not the resource is in the local file system or somewhere out there on the Internet. Essentially, this lets Tectonic view a document as a discrete entity rather than the result of fetching data from the system and massaging it into a particular form.

Resources are configured through the Delve Habitat using a resource wizard that lets the developer select the document to associate with a resource. Delve supports mapping a WSDL document to a "Foreign Service Interface" resource. This allows for simple access to external Web Services as shown in the lower window in Figure 1. The ease with which external Web Service calls can be mocked up in the Delve environment lends itself well to testing. I could envision training testers in using Tectonic and Delve and have them build a suite of test cases for all services developed in a project or across an enterprise. Tectonic can also be used to mock up a service to facilitate client development before the "real" service is delivered.

> **"In a world where sexy BPEL orchestration tools are prevalent, a simpler language-oriented model can be a welcome alternative"**

Licensing & Support

Cretaceous Software has kept the licensing simple and inexpensive. Delve Development Habitat 1.0 is $75 per developer seat. Cretaceous offers unlimited online support and up to four phone support incidents for license holders for up to 12 months after purchase.

Conclusion

In a world where sexy BPEL orchestration tools are prevalent, a simpler language-oriented model can be a welcome alternative. I must admit that when I was asked to review Cretaceous' product, I was skeptical. But as I tested the tool and dropped into other environments to test interoperability, Tectonic and Delve always seemed simpler and quicker to use.

The Tectonic language and Delve Development Habitat remove the impedance mismatch between code and service access typical in many languages. The language and tools are simple and straightforward to use, but can support complex implementations
Any readers out there who are looking for tools that do fast prototyping in an SOA environment should definitely consider Tectonic and the Delve Programming Habitat.

About the Author

Paul T. Maurer is principal consultant for PTM Consulting Services, LLC. He consults in the area of enterprise architecture and software.

paul@paulmaurer.net

Cretaceous Software
7229 West Franklin Blvd
Boise, ID 83709

Web: http://www.cretaceoussoftware.com/
E-mail: inquiries@cretaceoussoftware.com

Licensing Information
Delve SOA Fast-Prototyping Toolkit (Single User): $75

Testing Environment
OS: Windows XP Professional (Service Pack 2)
Hardware: 2GHz Intel Core 2 CPU M, 2GB RAM

A Close Look at BPEL 2.0

Drilling down to the next level of detail

WRITTEN BY **MANOJ DAS, KHANDERAO KAND, AND ALEX YIU**

As most readers are probably aware, the Web Services-Business Process Execution Language (WS-BPEL) provides a broadly adopted process orchestration standard supported by many vendors today and used to define business processes that orchestrate services, systems, and people into end-to-end business processes and composite applications. However, in many ways BPEL's adoption has gotten ahead of the formal standardization process.

The BPEL4WS 1.1 specification was submitted to OASIS back in 2004 and after three years of work by one of the largest technical committees at OASIS, WS-BPEL 2.0 finally became an OASIS standard on April 12, 2007. While adoption of the BPEL language has not been gated by the 2.0 standard or the OASIS stamp of approval – there are thousands of successful BPEL projects and deployments today – the formal publication of the standard is an important milestone and will further accelerate BPEL's adoption and vendor support.

A lot has been already written about the new features in WS-BPEL 2.0 on various blogs, Web sites, and magazine articles, including the article "BPEL Grows Up" (http://soa.sys-con.com/read/346372.htm). In this article, we'll drill down into the next level of detail regarding the new features in WS-BPEL 2.0 using concrete examples wherever possible. Throughout this article, we abbreviate BPEL4WS 1.1 as BPEL 1.1 and WS-BPEL 2.0 as BPEL 2.0.

BPEL 2.0 Overview

At a high level BPEL is an XML language that provides a rich set of activities to describe an executable business process. The processes and activities can be synchronous or asynchronous, short-lived or long-running; BPEL provides a sophisticated language for defining the process flow, system interactions, data manipulation, exception handling, compensation rules, etc. First, we will briefly summarize the important features of the BPEL standard, explicitly calling out what is new or changed in BPEL 2.0:

- **Service interaction activities:** A BPEL process is automatically a Web Service and receives inputs via <receive> or <pick> activities. A process can send back a synchronous Web Service response using <reply>. The <invoke> activity is available to invoke an external service, as described below, but also to respond asynchronously to a client.
- **Event handling constructs:** A process can get input requests at non-deterministic points during process execution with <eventHandlers> using <onEvent> for messages (new in BPEL – replac-

es <onMessage> from BPEL 1.1) or <onAlarm> for time-triggered events. <wait> can wait for a specified time or until a deadline is reached and <receive> can wait for events at pre-determined points in the process.

- **Back-end system interactions:** Interactions with external services are represented as <partnerLinks>. Asynchronous conversational interactions can be correlated using <correlationSet> or the WS-Addressing standard. A process maintains its state using <variables> that can be defined at global or local scope. BPEL 2.0 makes it easier to map process variables to WSDL message variables. It also provides the new <messageExchange> activity to distinguish instances of similar conversations (request/response pairs).
- **Data manipulation activities:** BPEL 2.0 adds a new simplified XPath notation ($variableName) replacing the getVariableData() function. Besides the existing <assign> activity to map data between variables, BPEL 2.0 provides a doXSLTransform() function to natively support XSL Transformations. A <validate> activity has been added for schema validations. These additions have already been time tested, having been implemented as extensions in vendor implementations of BPEL for quite some time now.
- **Process structural flow related activities:** BPEL includes basic structural activities similar to other workflow or programming languages for sequencing, iteration, and branching. BPEL 1.1 supported <sequence> for sequential execution, <flow> for parallel branches, and <while> for looping. BPEL 2.0 adds <if> / <else>, <repeatUntil> and <forEach> for richer flow control syntax. In particular, the new <forEach> construct now supports dynamic parallelism (executing N activities in parallel, when the value N is not known until execution time). This was not supported in BPEL 1.1 except through vendor extensions.
- **Exception handling and recovery constructs:** Exceptions, represented as faults, are propagated using <throw>, and BPEL 2.0 adds <rethrow> to provide more explicit control over exception management patterns. In <faultHandlers>, faults can be detected using <catch> and <catchAll>. A process can undo completed work through <compensationHandlers> and the <compensate> activity; BPEL 2.0 adds <compensateScope> to clarify the syntax of BPEL 1.1's overloading of the <compensate> activity. BPEL 2.0 also adds <terminationHandlers> to enable processes to specify the logic to be invoked just prior to the termination of a process.
- **Extensibility:** BPEL 2.0 adds <extensionAssignOperation> to extend the standard <assign> activity; it also provides <extensionActivity> to add new activity types. This is another area where the 2.0 standard now explicitly covers things that vendor implementations were already doing. BPEL 2.0 also now supports <import> and <documentation>.

Kitchens Online Use Case

To illustrate some of the BPEL features, we will use the example of Kitchens Online, a fictitious Internet-based kitchen-remodeling solution. Kitchens Online provides a Web site where customers can select appliances and cabinets and schedule delivery and installation. Kitchens Online doesn't carry any stock and sources the different components from various vendors; however it still wants to keep inventory information as up-to-date as possible. Complicating matters, the vendors Kitchens Online's suppliers don't always have automated inventory systems – in many instances, a person needs to check manually whether a given item is available. When a customer places an order, Kitchens Online tries to reserve all the needed components from its vendors. If it can't reserve a piece, the customer is notified and can then change or cancel the order.

An overview of Kitchens Online's business process is shown in Figure 2.

This example highlights the following advanced requirements:

- **Dynamic Parallelism** – handling of different components (line items) of an Order in parallel
- **Change handling (out-of-band events)** – a delivery schedule may change while an Order is in process
- **Compensation** – reversing credit card charges and releasing reserved inventory when a component is out-of-stock

Since this article is focused on the BPEL 2.0 standard, and BPEL is an XML language, we'll examine the BPEL "source code" in XML format. Of course most developers won't hand-code BPEL in the XML format – they'll use visual tools built on top of the standard. Several vendors and open source implementations of such tools exist, including some that enable developers to switch back and forth between a visual model of a BPEL process and the underlying BPEL source. Here we'll examine the underlying XML source, so get ready!

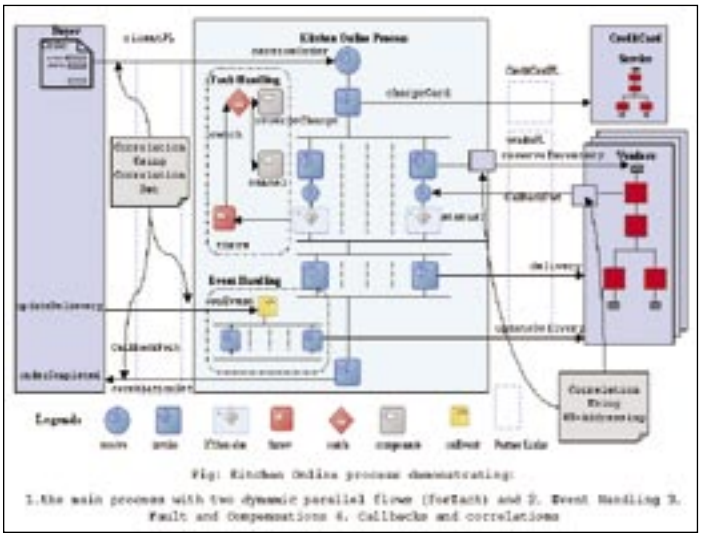


Figure 1 Summary of BPEL features

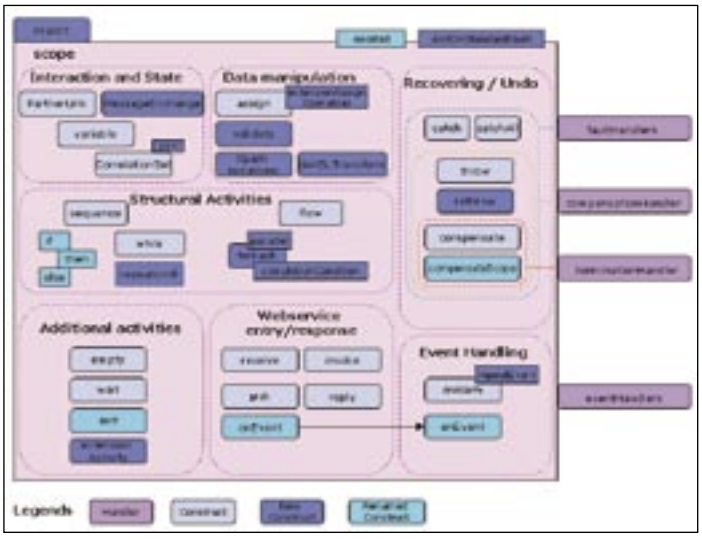


Figure 2 Overview of Kitchens Online's business process

Dynamic Parallelism

The Kitchens Online requirement to process the multiple items of an Order in parallel is a very common pattern. The <flow> activity introduced in BPEL 1.1 supported parallelism, but only when the number of parallel branches was known at design time. Here, an Order may have a variable number of items and therefore the number of parallel branches won't be known until runtime. What's needed is the equivalent of a typical programming language "for" loop where all the iterations of the loop are executed in parallel.

To address this pattern, BPEL 2.0 introduces the <forEach> activity. As the name indicates, this activity causes the enclosed scope to iterate between the <startCounterValue> and <finalCounterValue> inclusive (i.e., N+1 times, where N is the difference between the two). The <forEach> activity has an attribute parallel that when set to yes causes all the branches to be executed in parallel and a counterName attribute that specifies an integer variable that will hold the counter value for each particular iteration (1 for the first iteration in the example below, 2 for the second, etc.).

Listing 1 illustrates how Kitchens Online can use the parallel <forEach> activity to process all items in the Order in parallel. Notice the use of a local <partnerLink>, vendorPL, in the <scope> reserveInventoryItem; BPEL 2.0 introduced this notion of a local partnerLink to create a different service reference dynamically for each branch. Also notice how this <partnerLink> is being initialized – sref:service-ref is another concept introduced by BPEL 2.0. It defines a way to make a service reference in a process. Typically, local variables are used within <scope> to hold request and response data, in this case, reservationResult for each parallel iteration. The <forEach> activity also supports the specification of a <completionCondition> to complete the loop when the specified condition is met without waiting for all branches to complete. This enables the "N out of M" join pattern where a process would dynamically poll a number of services but wants to continue without waiting for all responses to be received. We have often seen this in cases where a minimum of two price quotes are required or where a minimum of two human approvals are needed.

Change Management with Out-of-Band Events

Kitchens Online takes pride in its world-class customer service and a key business principle is to be as flexible as possible in every customer interaction. What this implies for our ordering process is that customers should be allowed to change their delivery schedule at any time before Kitchens Online actually initiates shipping. This is a very common requirement, not only in order processing but also in any business process where the state of the business environment can change at any time.

Kitchens Online uses <eventHandlers> to implement this requirement. Listing 2 illustrates the use of <eventHandlers> in the Kitchens Online scenario.

BPEL lets <eventHandlers> be specified for different scopes, allowing an event to be handled differently based on the state of the business process when it is received. For example, Kitchens Online's process includes the steps after shipping is initiated – shipping, delivery, and installation – in a separate scope from the rest of the process; the event handler is specified only in the scope containing the process prior to shipping (i.e., the <scope> main in Listing 2). Therefore, once the process reaches the shipping step, the event handler will be no longer listen to the modifyDeliverySchedule request. In this use case, a different event handler would handle delivery change requests that are received after shipping has been initiated – perhaps by tasking a customer service rep to call the customer and inform them that it's too late to change the delivery schedule without incurring some cost.

Besides out-of-band event handling as illustrated in the above example, another common scenario enabled by <eventHandlers> is where a process exposes a query or cancel operation that may be received at any time during the execution of a process. BPEL also supports waiting for events at specific points in a process using <receive> and <pick>.

In this example, when the BPEL engine receives the modifyDeliverySchedule event, many instances of the process may be active; how does it know which in-flight process instance the event should be sent to? This is where the <correlations> feature of BPEL comes in, which specifies the attributes of the event that are used to correlate it to a specific process instance. In the example above, we used the orderId and customerId attributes for correlation. In addition to event handling, <correlations> can be used for other asynchronous conversational patterns where the correlation of messages needs to be done based on the content of the message. Besides this concept of payload-based correlation, WS-Addressing has emerged as a standard for correlation in the WS stack; it's complementary to the BPEL standard and commonly supported by BPEL and other Web Services engine implementations.

Readers familiar with BPEL 1.1 may have noticed the use of <onEvent> in place of <onMessage>. This is a syntactic difference in BPEL 2.0 – use of <onMessage> is limited in BPEL 2.0 to <pick>. BPEL 2.0 also includes some clarifications on how variables, partnerLinks, and correlation sets are resolved for <onEvent>.

Exception Handling & Compensation

Although Kitchens Online tries to keep inventory information from its suppliers up-to-date, it can run into situations where a vendor is unable to supply an item in time to meet a committed delivery window. It's Kitchens Online practice to notify the customer of its inability to fulfill an Order within a 24-hour window.

Kitchens Online's business process is first to reserve inventory from all suppliers. If one or more suppliers come back with an out-of-stock notification or fails to confirm the reservation within 24 hours, Kitchens Online must release the inventory reserved from other suppliers and notify the customer.

Kitchens Online uses BPEL's <faultHandlers> and <compensationHandler> feature in conjunction with <onAlarm> to achieve the desired behavior as shown in Listing 3.

Some of the interesting aspects of the example above are:

- **Asynchronous fault handling** – Typically, we think of exceptions as faults and catch them as such. However, in this example, as in many other asynchronous interaction scenarios, the suppliers come back with their responses asynchronously any time within the 24-hour period. However, compensation in BPEL can only be invoked in certain activities such as <faultHandlers>. Therefore, after getting an asynchronous response, if needed, we do a <throw> to generate a fault. Timeout is handled similarly by doing a <throw> from within the <onAlarm>.
- **Fault propagation** – Notice that the cannotReserveInventory fault is thrown from several scopes but we catch it in the main scope. BPEL automatically propagates uncaught faults up to enclosing scopes, similar to how try-catch works in Java. BPEL 2.0 also features <rethrow> to propagate caught faults. The example above uses <rethrow> to enable other parts of the process to be terminated gracefully. (Note: the <exit> activity, previously known

Web 2.0 and Rich Internet Apps

CALL FOR PAPERS NOW OPEN!

September 24-26, 2007

Santa Clara Convention Center
Hyatt Regency Silicon Valley
Santa Clara, CA

To Register
Call 201-802-3020 or
Visit www.AJAXWorld.com

Hurry! Limited Seating
This Conference Will Sell-Out!

Providing 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.

On September 24-26, 2007 over 1,000 developers, architects, IT managers, and software professionals of every stripe will be converging in Santa Clara, CA to attend the West coast AJAXWorld Conference & Expo -- the most comprehensive meeting on the most significant technology subjects of recent times: AJAX, Rich Internet Apps & Web 2.0.


Hyatt Regency Silicon Valley
Santa Clara, CA

Join Over 3,600 Early AJAX Adopters Who Attended the #1 Web 2.0 Event in the World!

Experience AJAX, RIA, and Web 2.0 Knowledge and Best Practices at AJAXWorld Conference and Expo 2007

Delegates will hear first-hand from the creators, innovators, leaders and early adopters of AJAX, and a slew of leading vendors will provide sneak-peeks of the very latest frameworks, tools, products and applications during the conference, which will have over 100 sessions and presentations given by 125 different speakers - the largest AJAX-focused speaker faculty ever assembled in one place at one time.

AJAXWorld Conference & Expo 2007 West will be jump-started with a full one-day "AJAX University Boot Camp," followed by the Main Conference and Expo over the following 2 days.

Highlights Include:

- ***100+ conference speakers*** *selected from among the world's leading AJAX technologists and practitioners, including high-level IT executives, industry thought leaders, VCs and analysts*
- ***Topical, Interactive "Power Panels"*** *simulcast on SYS-CON.TV*
- ***Demos from 30+ vendors*** *both in General Session and in the Exhibit Hall/AJAX Showcase*
- ***Pre-Conference "AJAXWorld University Bootcamp"*** *for those wanting an all-day, hands-on encounter with AJAX*
- ***Leading-edge sessions*** *on issues like Offline AJAX, Mobile AJAX, Enterprise AJAX, AJAX Security, Desktop AJAX, Semantic Mash-Ups, Next-Generation SaaS, JavaScript & XML, Google Web Toolkit, Adobe Apollo, Adobe Flex, AJAX RIA GUIs, and Appropriate vs Inappropriate Use of AJAX.*

SYS-CON EVENTS
For more great events visit www.EVENTS.SYS-CON.com

VISIT WWW.AJAXWORLD.COM FOR THE MOST COMPLETE UP-TO-DATE INFORMATION

as <terminate> in BPEL 1.1, is typically not a good way to stop a process since it "kills" the process immediately without letting it do any recovery work or release any allocated resources.)
- **Scope Snapshot** - When a scope (such as processPayment) completes its work, all the local variables (such as chargeResponse) of the scope are saved in case of the need for compensation. Whenever the compensation of a scope happens, the saved state of local variables will be made available to the compensation logic (such as the reverseCharge service invocation).

If one supplier responds with an out-of-stock message, we release inventory from all suppliers without checking whether they had actually reserved it or not. Some readers may wonder if this would lead to incorrect inventory accounting on the vendor side. Such a check isn't needed because of the way compensation works in BPEL – a scope in BPEL is compensated only if it has successfully completed; therefore, scopes waiting to for inventoryReserved won't be compensated.

What's Next

The aforementioned article "BPEL's Growing Up" highlighted some of the interesting developments in the BPEL world, specifically around human workflow and process model round-tripping. By the time this article gets into print, we expect there to have been some significant announcements around human workflow. Another interesting development is the Service Component Architecture (SCA) standard. While most WS-* specifications have been focusing on the protocol and interoperability aspects of Service Oriented Architecture (SOA), SCA standardizes the deployment and assembly of SOA components, including BPEL. This will further help enable the portability of BPEL processes, and the composite applications that leverage them, across different vendor implementations.

Summary

Even prior to the publication of the BPEL 2.0 standard, BPEL had become the de facto standard for process orchestration. Now with the formal standardization process complete, we can only expect the adoption to accelerate.

The efforts of the BPEL committee over a three-year period have enhanced BPEL 2.0 with important features and clarified the specification in several areas. In this article we covered some of the most interesting new features in depth, however not every aspect of BPEL 2.0 could be covered. Readers interested in learning more about BPEL and BPEL 2.0 should refer to the WS-BPEL 2.0 primer (http://www.oasis-open.org/apps/org/workgroup/wsbpel/download.php/23974/wsbpel-v2.0-primer.pdf) and the WS-BPEL 2.0 specification (http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.pdf). The primer is a good introduction to BPEL concepts, while the specification lets advanced readers reference detailed and precise semantics of the language. ■

About the Authors

Manoj Das is senior manager in the product management group for Oracle Fusion Middleware. His focus is on BPEL and business rules. Manoj joined Oracle with the Siebel acquisition where he was responsible for driving the next-generation process-centric application platform.

Khanderao Kand is an architect for Oracle Fusion Middleware at Oracle. He is involved in the development of various integration and BPM technologies including ESB. He is an editor of WS-BPEL's Primer in WS-BPEL's technical committee at OASIS. Earlier he played a role as an enterprise architect of Peopletools and an architect in CRM.

Alex Yiu represents Oracle as an editor in WS-BPEL 2.0 specification, the WS-BPEL 2.0 primer, and the SCA (Service Component Architecture) BPEL specification. He was also a part of joint task force that designed Oracle's cross-tier binary XML format (patent pending). Prior to that, Alex represented Oracle in Java Server Pages JCP Expert Groups and was a speaker at a JavaOne technical session on XML data integration.

khanderao.kand@authors.sys-con.com
manoj.das@authors.sys-con.com
alex.yiu@authors.sys-con.com

Listing 1 Sample excerpt showing dynamic parallelism with <forEach>

```
<forEach parallel="yes" counterName="n">
   <startCounterValue>1</startCounterValue>
   <finalCounterValue>count($order/lineItem)</finalCounterValue>

   <scope name="reserveInventoryItem">
      <!-- variables, partnerLinks defined here are local to each itera-
tion -->

      <partnerLinks>
         <partnerLink name="vendorPL" .../>
      </partnerLinks>

      <variables>          ...      </variables>

      <sequence>
         <assign>
            <copy>
               <from>$order/lineItem[$n]/vendor/sref:service-ref</from>
               <to partnerLink="vendorPL" />
            </copy>
            ...
         </assign>

         <invoke name="reserveInventory" partnerLink="vendorPL" .../>
         <receive name="inventoryReserved" partnerLink="vendorPL" .../>

    ...

         <assign>
            <copy>
               <from>$reservationResult</from>
               <to>$order/lineItem[$n]/reservationResult</to>
            </copy>
         </assign>
      </sequence>
   </scope>
</forEach>
```

Listing 2 Example excerpt of change handling

```
   <eventHandlers>
      <onEvent operation="modifyDeliverySchedule"
         messageType="ns:DeliveryPreferences" ...>
         <correlations>
            <correlation set="orderId" ... />
            <correlation set="customerId" ... />
         </correlations>
         <scope>
               ...
               <invoke name="updateDelivery" partnerLink="deliveryPL"
...>
               ...
         </scope>
      </onEvent>
   </eventHandlers>
```

Listing 3 Excerpt showing fault handling and compensation

```
<scope name="main">

   <!-- If we fail to reserve all inventory, undo the inventory we
        already reserved and back out the payment processing -->

   <faultHandlers>
      <catch faultName="ns:cannotReserveFault" ...>
          <sequence>
             <compensateScope target="reserveInventory" />
             <compensateScope target="processPayment"/>
             ...
             <invoke partnerLink="clientPL" operation="orderCancelled"
...>
             <rethrow/>
           </sequence>
      </catch>
   </faultHandlers>

   ...

   <sequence>
      ...
      <scope name="processPayment">

        <!-- This scope processes credit card payments. The compensation
             handler for it undoes its work - here reversing the charge.
             Note the BPEL engine will maintain variables' values for
the
             purpose of compensation -->

        <variables>
           <variable name="chargeRequest" .../>
           <variable name="chargeResponse" .../>
         </variables>

         <compensationHandler>
             <invoke name="reverseCharge"
                inputVariable="chargeResponse" .../>
         </compensationHandler>
         <sequence>
             ...
          <invoke name="chargeCard"
                inputVariable="chargeRequest"
                outputVariable="chargeResponse"
                ... />
         </sequence>
      </scope>

      <!-- Here we reserve the inventory. If it doesn't succeed within
24
           hours, the fault we throw will be caught above and cause the
           compensation logic to release the inventory and reverse
charges -->

      <scope name="reserveInventory">
         <eventHandlers>
            <onAlarm>
                <for>"P24H"</for>   <!-- P24H == "24 hour period" -->
                ...
                <throw faultName="ns:cannotReserveFault" .../>
                ...
            </onEvent>
         </eventHandlers>

         <forEach ...>
            ...
            <scope name="reserveInventoryItem">
                ...
                <compensationHandler>
                    <invoke name="releaseInventory" ... />
                </compensationHandler>
                <sequence>
                  ...
                  <invoke name="reserveInventory" ... />
                  <receive name="inventoryReserved" ... />
                  <if>
                    <condition>
                      not($reservationResult/status='success')
                    </condition>
                    <throw faultName="ns:cannotReserveFault" .../>
                  </if>
                  ...
                </sequence>
      </scope>
         </forEach>

      </scope>
   </sequence>
</scope>
```

On-Demand Integration

Connecting SaaS providers with their customers

WRITTEN **BY JAMES PASLEY**

Software is now increasingly provided as a service; in other words, it is now offered as a hosted application that users access through Web browsers. Many companies see this as an effective way of outsourcing some of their IT requirements. However, they face an increasing number of integration issues as part of this strategy. Many are turning to ESBs for a solution.

As the use of Software as a Service (SaaS) increases, there is a growing realization that companies making use of SaaS applications need to integrate them within their overall IT infrastructure. This means that data needs to flow between the SaaS applications and their other IT systems. SaaS providers typically provide programmatic interfaces to facilitate this. From the SaaS providers' perspective, there is a danger that the overhead of achieving such integrations could take away from the core values of the SaaS model. The SaaS model offers no installation overhead and is typically priced on a per-user basis. Customers can choose to start with a small number of users and scale up as demand increases. However, the need for integration with other applications can be seen as an upfront cost. It can also delay initial adoption of the software and replace the favorable first impressions of the intuitive browser-based user interfaces with a discussion on reconciling disparate document formats. In addition to providing their core software on demand, SaaS providers are now finding it necessary to provide integration solutions consistent with their on-demand model.

There are a number of different ways in which SaaS providers are addressing these integration tasks. In most cases, the first step is to provide public APIs through which their services can be assessed programmatically. These are typically provided either as Web services (SOAP messages described by WSDL) or through REST style interfaces. In both cases, this means the exchange of XML documents over an HTTP transport. These technologies can be used universally and ensure that regardless of the nature of the customers' IT environment, the technologies will be able to build applications that can make use of the interfaces. In some cases, client-side libraries are also provided for particular environments to further reduce the overhead for customers building integrations. Even so, SaaS providers are finding that customers' tolerance for integration projects is very low, particularly where claims that "software is dead" have been made. In short, expectations are high and customers are getting more demanding. There are, of course, precedents for satisfying such customer expectations. A number of years ago, I was involved in a project with a leading investment bank. The goal was to reduce costs and increase efficiency by allowing their brokers to submit trade instructions online. One of the defining characteristics of this project was the level of assistance and encouragement that brokers needed in moving to the new online system. After all, the major cost savings would benefit the bank and were not of immediate concern to the brokers.

The bank created a new Web-based front end to allow trade instructions to be entered online. This included a feature to allow batches of trade instructions to be uploaded in files. To facilitate rapid client integration, the bank needed to make it as simple as possible for clients to submit their trade instructions. Rather than mandating a single file format, which would require clients to transform their data before submitting it, the bank allowed clients to submit their trade instructions electronically using their existing in-house file formats. A variety of text formats including CSV and Swift were commonly used by the brokers in this scenario.
At the same time, the bank wanted to encourage a number of clients who where still submitting trade instructions via fax to move online. This was very convenient for the clients so they were reluctant to change, but it involved the overhead of manual re-entry for the bank. Manual re-entry can be error prone and the need for interventions to fix errors drove the costs higher. The use of Excel spreadsheets by the client was very popular in this scenario, so a solution that allowed the spreadsheets to be uploaded rather than faxed was an easy transition for the clients.

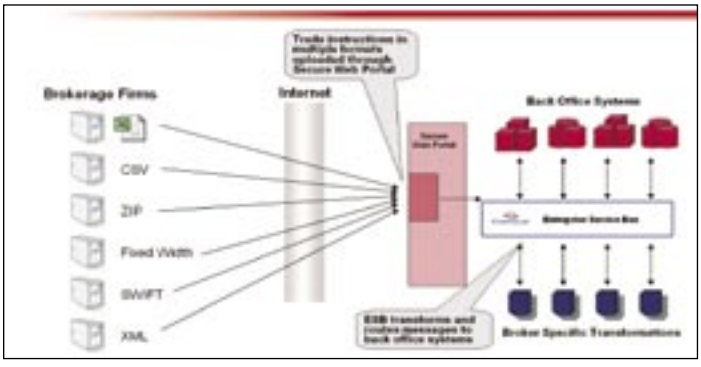


Figure 1 Investment bank hosting integration on behalf of brokers

As shown in Figure 1, Web-based front ends were created for a number of the bank's back-end systems through the bank's secure Web portal. An Enterprise Service Bus (ESB) was used to provide a Web services interface to which data could be submitted. The ESB was also used to host data transformations that converted from each broker's file formats to the bank's canonical format. The ESB would validate the incoming documents, transform them, and automatically route the data to the correct back-end system. This mechanism could also be accessed by uploading files through the Web portal. In most cases, this was the mechanism preferred by users.

Why was the decision taken to host these integrations? In this case, the logic is very clear. The brokerage firms had relatively little IT infrastructure and didn't have IT staff on hand to start developing Web service clients. The savings provided by eliminating the overhead of data re-entry and the increased efficiency by which new brokers could be integrated justified the cost of developing and maintaining the integrations. Hosting the integrations on the ESB provided a very clean way to isolate this task from the core functions, and allowed them to be managed separately. Good old fashioned customer service was also a factor; solving the integration problem as part of the service provided to brokers gave the bank an advantage over the competition.

Now, let's fast forward to 2007 and take a look at the work the industry is doing with SaaS providers. For the investment bank, the decision to host integrations was straightforward. For today's SaaS providers, similar factors can come into play. In these situations, a similar architecture diagram can be drawn (see Figure 2). These days it's common for the core functionality to be exposed using REST interfaces. An AJAX-enhanced user interface exposes this functionality to the user. The Web services interface sits alongside this, providing document interfaces to the functionality. Hosted integrations can be provided through the use of mediation services that sit in front of the Web service interfaces. These mediations focus on the task of data transformation and of supporting the variety of transports that a customer may need to use – HTTP of course, but also e-mail and FTP.

Hosting integrations is not the only option available to SaaS providers. Customers may be willing to host these integrations themselves. However, they expect assistance in the creation of these integrations and may expect an integration solution to be provided. SaaS providers engage with customers to educate them on the use of their document formats and, in some cases, also develop the data transformations. ESBs are finding a new use as a quick and convenient way to host integrations delivered to customers. ESBs provide the connectivity necessary to extract data from customers' existing systems. They can then use Web services technologies to create secure and reliable links to the SaaS provider. The use of an ESB on the client side also serves as an elegant way to support the two-way exchange of asynchronous messages in situations where the SaaS provider needs to push messages toward the client.

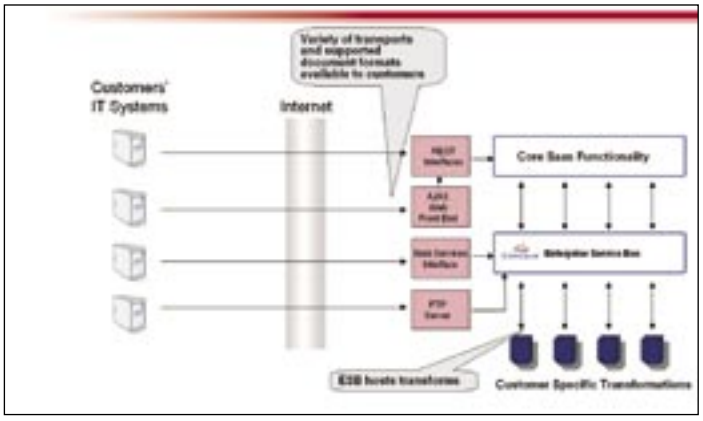


Figure 2 Hosted integrations in a SaaS environment

The Web services technology used to facilitate these integrations makes it possible for them to be hosted by a third party. However, the introduction of a third party into these relationships remains an issue as customers or providers may be reluctant to do this. For this reason, the majority of integrations are either hosted on the client side, or by a service provider.

The SaaS use case represents a change in the traditional way in which ESBs are used. An ESB for SaaS needs additional features over and above those of a vanilla ESB.

- Multi-tenanting support is fundamental to the SaaS model, but good support for it is not available in many ESBs. Hosting integrations on behalf of customers requires that data generated as a result of the messages flowing through the ESB is segmented on a per customer basis. This applies to a whole variety of information stored in log files, activity reporting, and data stored in databases. The segmentation of data in databases also applies to languages such as WS-BPEL, which are heavily dependant on the persistence of data. The SaaS provider may need to make this data available to their customers in order to provide the same level of visibility into their integration solutions as they do for their core services.
- Scalability and performance requirements for SaaS are typically more demanding than within the enterprise environment. The ability to deploy within a clustered environment is now a must.
- Productivity tools are vital. It must be possible to create integration solutions quickly for customers. Within the enterprise, integration tasks might have been viewed as one-off projects. For the SaaS provider, they are now a normal part of providing access to their systems to each new customer.
- The software license model for the ESB needs to be consistent with the on-demand or per-user billing model for the SaaS provider.
- The ability to deploy client-side instances of the ESB is also important as the SaaS provider may not want to host all the integrations. For this scenario, ease of installation and maintenance is vital as is the need to work within a wide variety of IT environments.

In summary, SaaS providers are finding that in order to satisfy their customers' needs and expectations, they must actively engage in solving the issue of integration. Many are turning to ESBs to assist them in achieving this, either as a means to host integrations or to deliver solutions to their customers.

About the Author

James Pasley is chief technology officer at Cape Clear Software. As CTO, James is Cape Clear's lead technologist and is responsible for ensuring that Cape Clear's enterprise service bus (ESB) technology supports the evolving needs of the company's global customer base. Read his blog at http://www.jpasley.com/.

james.pasley@capeclear.com

Introduction to Complex Event Processing & Data Streams

OMG's Data Distribution Service data streams when you need high performance from complex event processing systems

WRITTEN BY **SUPREET OBEROI**

With the evolution of distributed IT systems and the advent of Web Services, applications can now make more informed decisions by using real-time information from third-party sources. For example, today's automated trading applications can make over 30 trades a second by analyzing stock trends, market movements, news, and events that may only be relevant for a fraction of a second. A trading algorithm may infer a negative stock trend line for the next tenth of a second and may short the stock for that bit of time.

Such applications require Complex Event Processing (CEP) engines. These engines can detect patterns of activity from multiple data streams and infer events continuously. Many critical CEP use cases require peak performance from both the event engine and the data streams. In the example above, the trading algorithm can't profit from making buys and sells in a tenth of the second if the event engine and the data stream latency exceed the operable time window.

This article will survey the suitability of OMG's Data Distribution Service data streams in use cases that demand high performance from complex event processing systems.

What Is Complex Event Processing?

Consider the following example: The Securities and Exchange Commission has different margin and reserve requirements for traders classified as "pattern day traders." The term "pattern day trader" means any customer who executes four or more day trades in the same stock in five business days. However, if the number of day trades is 6% or less of his total trades for the five-business-day period, the customer won't be considered a pattern day trader and the special margin and reserve requirements won't apply.

With Complex Event Processing, we can detect a "pattern day trader" in real-time as follows. Each time a trader makes a transaction, the system posts an executedTradeEvent <traderId, stockId> event. The CEP engine maintains a window of five days and searches for cases where the count (count_n) of executedTradeEvents for a given trader and a stock exceeds four. Detecting such a pattern it counts the total number of trades done in the last five days. If count_n is more than 6% of all trades in the last five days then it posts a detectedDayTrader event.

Based on this example, we can describe many key characteristics of Complex Event Processing:

- Events are inferred. In the example, the trading application doesn't and can't send a detectedDayTrader event. This event has to be inferred by processing other events like executedTradeEvent, maintaining a count of events satisfying a query condition over a given period of time and integrating with non-streaming content like the number of total trades stored in a relational database.

- The system correlated the data from multiple sources to infer the

event. In the example, it included explicit sources like the database that stored the total number of trades for a customer and implicit sources like time.

- The system wouldn't have sent a detectedDayTrader if there wasn't a set of executedTradeEvents preceding it. The executedTradeEvent plus some other criteria caused the detectedDayTrader event to occur.

CEP engines manage event-driven information systems by employing techniques such as detecting complex patterns, building correlations, and relationships such as causality and timing between many events.

From a black box view, a CEP engine takes as input a set of input streams like RTI, database files, or JMS. Most CEP engines use an SQL-like programming language such as the Continuous Computation Language (CCL) with extensions for event processing.

While discussing the entire semantics of CCL or CEP is beyond the scope of this article, here's a simple example of how application developers can infer a weather event using the CCL programming language.

In this example, the query searches for events from the input data stream WindIn in which the wind speed changes by more than five miles an hour in two seconds. The events matching the pattern are then inserted into an output data stream WindPatternOut:

```
INSERT INTO WindPatternOut (Location, Speed1, Speed2)
SELECT W1.Location, W1.WindSpeed, W2.WindSpeed
FROM WindIn W1, WindIn W2
MATCHING [2 SECONDS: W1 && W2]
ON W1.Location = W2.Location
WHERE (W1.WindSpeed - W2.WindSpeed) >= 5;
```

As you can see from this example, CCL is loosely based on SQL semantics with extensions (such as matching patterns, viewing samples in a given time window) for complex events. The input and the output data streams are logically modeled as database tables, regardless of the underlying messaging protocol.

Why Use Complex Event Processing?

From an oversimplified view, applications have implemented functionality for inferring events from existing data for a very long time. Credit and fraud-risk applications, for example, have existed for decades without an explicit design and implementation for CEP. However, the data avalanche produced by edge devices like Radio Frequency Identification (RFID) readers and sensors is rapidly changing design-detection algorithms and the need for a configurable and flexible way to detect patterns is becoming more vital. Here are some situations in which software architects might consider incorporating CEP into their application technology stack:

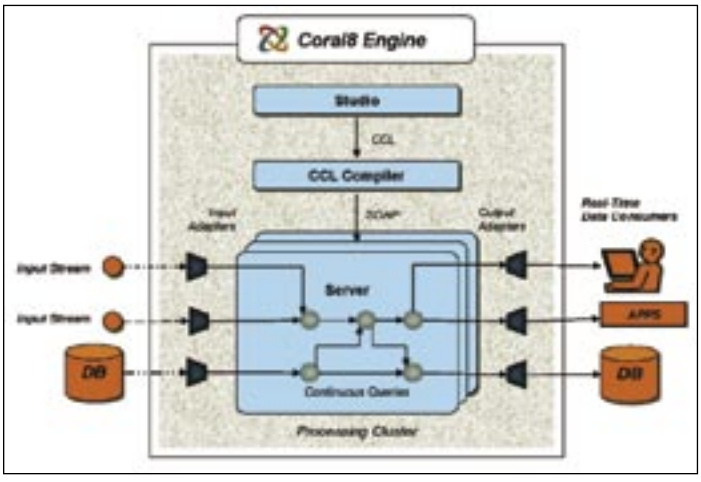


Figure 1 Architectural block diagram for a CEP engine

- **Only the "processed" data is useful:** In applications where edge devices such as sensors or RFID readers connect to the enterprise, all the raw data samples aren't of equal interest to the enterprise business process. The data might need to be cleansed, validated, and enriched before it's useful. In the wind example, the application isn't interested in each sensor read of the wind speed. The application is only interested when the wind speed changes by more than five miles an hour in two seconds, possibly inferring a hurricane or tornado.

- **Software development cycles can't keep up with the changes in algorithms for detecting patterns:** In trading applications, patterns for detecting buy and sell events may only be competitive for a few months, or even a few weeks. In some cases, new patterns are discovered, implemented, and deployed in a day. In such cases, it's necessary to parameterize and abstract out the pattern detection layer of the application. Having the trading algorithm embedded in the application code is not a good design practice.

- **Event processing has to be done in real-time:** Not all event processing requires a CEP engine. Data warehouse applications analyze trends by correlating multiple dimensions of the data in an "offline" manner (for example, send Home Depot discount promotions to all residents who have moved to zip code 95138 in the last two months). However, in many applications, events have to be inferred in real-time and the applications simply can't wait for the raw data to be persisted and analyzed. For example, radar tracking applications must process events in real-time. A trading desk application seeks a competitive advantage by analyzing an event microseconds ahead of its competition. Traditional event-detection applications require that the data is persisted first and then correlated. This methodology is too slow for applications such as trading desks where the data has to be analyzed for event patterns in real-time.

- **Event processing has to scale**: The media is now heralding the arrival of truly ubiquitous computing, where tiny microprocessors in our ambient surroundings communicate to deliver a more intelligent service. For example, in healthcare monitoring, pressure sensors in the shoes of patients at risk of a heart attack can monitor any change in their pattern of walking and alert the healthcare provider, who can immediately schedule a check-up. Enterprise applications that integrate with the edge have to cope with a large number of sensors simultaneously sending high rates of data. Traditional applications won't be able to handle this impedance mismatch of high data rate and volume. Using a CEP engine designed for performance, one can take steps to manage the load. Selecting the right messaging protocol for the data stream is the other step.

How to Select the Right Messaging Protocol for Your CEP Engine

Selecting the right CEP engine is only one part of building your

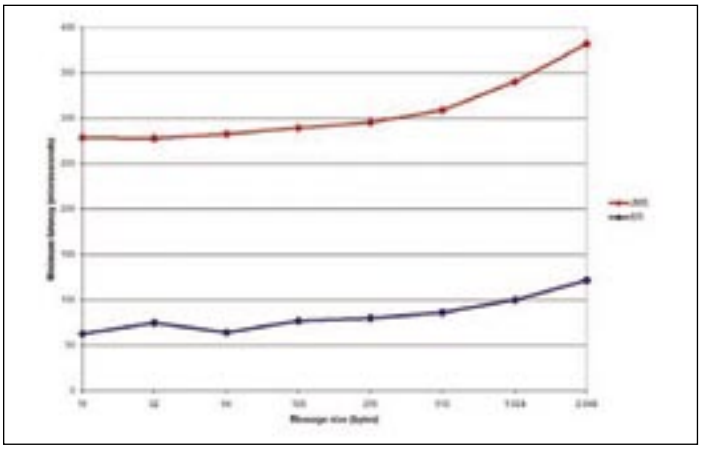


Figure 2 Comparison of latency for JMS and RTI's implementation of the Data Distribution Service

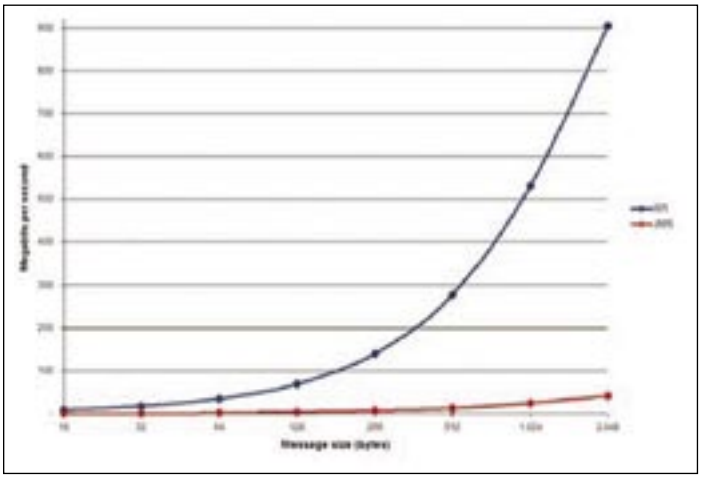


Figure 3 Network throughput comparison between JMS and RTI's implementation of the Data Distribution Service

event processing solution. The other significant part of the architecture is selecting the right messaging bus for your data stream. With the correct selection, you can fully leverage the high-performance CEP engine, scale to a large number of nodes, and do more. Consider the use case of trading desk applications where microseconds matter in identifying a trend for buying or selling a stock. Regardless of how fast the CEP engine may be a typical JMS implementation that usually delivers messages in tens of milliseconds will be a non-starter.

Here are some of the criteria to consider in adopting a messaging bus for your complex event processing needs:

- **Latency:** CEP isn't about speed. It's about correlating data from different data streams. However, chances are that if you're considering using CEP then latency is a significant concern for developing a successful application. In algorithmic trading, command and control, and fraud detection applications for electronic trades low latency of the entire application solution is critical. In trading desk applications, even the physical proximity of the trading floor to the exchange has become a critical issue. The extra nanosecond that a remote trade takes to register could cost the bank a deal. Many applications that rely on high-performance data streams simply can't afford the periodic (JVM) garbage collection that can degrade overall performance. Messaging products like implementations of OMG's Data Distribution Service, which have been designed and deployed for real-time mission-critical applications, can deliver event samples with a latency an order better than traditional JMS applications. In addition, while the overhead of Data Distribution Service depends on the message size and transport used. For reasonably sized messages and standard transports (100 Mbit-1 Gbit Ethernet) the overhead is typically less than 15% above the raw transport.
- **Managing network bandwidth:** Many developers who are considering CEP engines are also concerned with managing network bandwidth efficiently. In trading desk applications, the total volume of stocks traded daily has been following its own Moore's Law, doubling every 18 months since the start of electronic trading. This means the amount of data that must be processed follows the same curve (because the houses have to track all trades, not just their own). In addition, they have internal consumers of the data whose demands are expanding. New trading strategies are being developed that run in parallel with existing ones. Each model requires input and generates output.

Some messaging vendors can manage the network bandwidth more efficiently by packing more event samples into the network pipe and by sending only the relevant data over the network.

For example, with the Data Distribution Service, middleware can apply content-based filtering using SQL-like patterns so that only relevant data is sent over the network.

With pub-sub implementations of the Data Distribution Service, developers can configure the rate at which event samples have to be transmitted on a per data stream basis.

Consider the example of a market-data application that sends all stock ticks to the trading application. However, using CEP, sending a snapshot of the five-second stock high and low may not be required. By configuring a different transmission rate for the stock tick feed, and the "five-second high-low" feed, we can preserve network resources.

In addition, with intelligent network middleware for data streams, we can conserve network bandwidth by determining if the data has to be transmitted at all. For example, with the Data Distribution Service applications can subscribe to topics of interest such as stock news relating to a particular symbol or specific content within the topics of interest. If there are no subscribers for a given topic, RTI won't put the event samples on the network.

- **Quality of Service (QoS) for data streams:** QoS refers to a service contract made between two entities. Each data stream has unique attributes or characteristics. For example, a trading desk application may require resending all lost stock ticker change events (reliable transmission). However, a security surveillance application may only require that the data stream for video transmission use best-effort (rather than reliable) techniques. The Data Distribution Service provides a rich set of QoS contracts that can be enforced out-of-the-box. Some examples of such pre-built contracts include ownership strength (selecting the right data source when multiple sources are generating the same data for failover), history (how many event samples are needed for late-joining consumers of information), reliability, time- and content-based filtering of event samples at the source, and persistence (in case the publisher dies and a late-joiner consumer of the information arrives).

– continued on page 28

Visit the *New* www.SYS-CON.com Website Today!

The World's Leading *i*-Technology News and Information Source

24/7

FREE NEWSLETTERS
Stay ahead of the i-Technology curve with
E-mail updates on what's happening in your industry

SYS-CON.TV
Watch video of breaking news, interviews with industry leaders, and how-to tutorials

BLOG-N-PLAY!
Read web logs from the movers and shakers or create your own blog to be read by millions

WEBCAST
Streaming video on today's i-Technology news, events, and webinars

EDUCATION
The world's leading online i-Technology university

RESEARCH
i-Technology data "and" analysis for business decision-makers

MAGAZINES
View the current issue and past archives of your favorite i-Technology journal

INTERNATIONAL SITES
Get all the news and information happening in other countries worldwide

JUMP TO THE LEADING i-TECHNOLOGY WEBSITES:

IT Solutions Guide
Information Storage+Security Journal
JDJ
Web Services Journal
.NET Developer's Journal
LinuxWorld Magazine
Linux Business News
Eclipse Developer's Journal
MX Developer's Journal
ColdFusion Developer's Journal
XML Journal
Wireless Business & Technology
Symbian Developer's Journal
WebSphere Journal
WLDJ
PowerBuilder Developer's Journal

Improving the Customer Experience with DITA

Personalizing documentation by product, location, and, yes, customer

WRITTEN **BY JERRY SILVER**

We've all experienced the thrill of acquiring a new product only to have it diminished when it's not as easy to use as expected. You rip open the box ready to start playing with your new gizmo and 20 minutes later you're stuck on the phone with tech support because the instruction book was incomprehensible.

Obviously this experience negatively impacts the likelihood of you purchasing from this vendor again or recommending the product to a friend or colleague.

The quality of product information for customers is often an afterthought, yet the importance of any post-sales customer-facing information shouldn't be trivialized. While businesses invest heavily in customer service training and customer relationship management systems to improve customer satisfaction, they often overlook the experience that customers have with product documentation. User manuals, online help, Web self-service, and training provide the first touch points after the sale and the opportunity to make a good impression on a new customer. Informative instructions or trouble-shooting tips are crucial to ensuring the customers' full understanding of, and satisfaction with, the product. Because even if a customer is having a problem with the product, the manual or Web site are the first places people look to get started, where they expect to find answers to their questions quickly. If the customer gets discouraged and doesn't fully adopt the product, he's unlikely to recommend it to others.

Conversely, improving user aptitude increases the chances that they'll find greater functionality in your product — a key component for becoming the next product advocate. Strong documentation can be a key element to enhance customer loyalty. Moreover, well-crafted user guides and online documentation also reduce costs by driving down the number of help desk calls from new customers.

Yet, all too often documentation specialists and other information developers are bogged down by out-of-date systems and processes that hinder their ability to deliver the goods.

What Is a Good Customer Experience Worth?

Most companies recognize that customer experience plays a key role in their market position. 85% of companies surveyed in Forrester's Q4 2006 Customer Experience Peer Research Panel Survey said customer experience had a critical or a very important role in the firm's competitiveness. However, many of these same organizations felt their overall approach to customer experience management is lacking — with 57% reporting that their companies had an undisciplined approach to customer experience management.

Unfortunately, creating excellent product documentation is easier said than done, particularly for companies selling multiple products and/or distributing products globally. The sheer volume of documentation to be created for each product iteration and language can be staggering. Keeping up with changes is a monumental task. Moreover, documentation development can usually only begin once the product is near completion putting authors and documentation teams under immense pressure to deliver content

quickly and not delay time-to-market. Too often content quality suffers.

Overcoming these challenges involves tackling several issues:

- **Ensuring consistency between print, online documentation, help, and other forms of communication**
  With multiple options open to customers for finding information about products, companies must ensure that the information across these channels is consistent. Regardless of whether customers are interacting with a company via phone, Web, help files, manuals, retail outlets, or e-mail; "providing inconsistent content across multiple channels can generate customer frustration or the inability to respond to changing market conditions," according to a recent Forrester research report. However, delivering on these demands is no easy feat in one language, much less several.

- **Keeping information up-to-date in all channels**
  Organizations must have a mechanism in place to ensure timely updates across these multiple channels to ensure information is accurate by the time the customer reads it. This is not only critical in ensuring customer satisfaction but can be a compliance requirement in some industries.

- **Customizing documentation**
  Today's customer won't stand for "one-size-fits-all," yet writing content for each individual audience is too cumbersome. Documentation teams need a quick and efficient way to personalize content by product, location, and often by the role of the person reading it — thereby giving customers just the information they need, nothing more.

Businesses often overlook the impact that documentation has on customer satisfaction and view it as a last-minute item on the product release checklist. So how do organizations ensure the quality and quantity of valuable information isn't compromised in the rush to deliver? And, how do they keep up with the influx of content changes when the materials are produced in numerous versions, formats, and languages?

A New Approach

To face these challenges, leading documentation and information development teams are moving to the Darwin Information Type Architecture (DITA) to handle these tasks in a significantly more efficient manner. Simply put, DITA provides a framework for content creation and management that facilitates reusing and repurposing content across multiple communication channels and languages. This new open standard of XML information architecture is ideal for collaborating on and publishing high volumes of content in documentation and ensuring consistency and accuracy of that information.

Using DITA enables writers to streamline the entire content creation and updating process in four key ways:

- **Topic-level authoring and management**
  Traditionally, documentation teams developed individual books or manuals for each product, support teams wrote FAQs or knowledge base articles for online reading, and trainers put together training materials. This approach meant there was a considerable amount of redundancy of content and effort. One simple product feature change might necessitate changes to the same content in numerous places. Thus, keeping content updated to align with product changes has become a major challenge.

By using DITA, companies can now modularize content, writing and updating information in "chunks" instead of writing an entire document. Authors focus on writing small pieces of standalone information known as topics and then assemble these topics using DITA maps into the end-published media.

Without worrying about the information layout and formatting, authors can use the modular topics as building blocks for information products, support teams can leverage them in knowledge base articles and call center scripts, and trainers can reuse them in classroom materials. Information developers and subject matter experts across the organization can easily find and reuse DITA topics because they also contain metadata, making it easy to search and retrieve appropriate content. By managing content as objects instead of whole documents, organizations gain one updateable source of the truth — automating and accelerating the review, approval, and publishing processes.

Using DITA, modular content topics are easily extended to bring information such as interactive tutorials, support documentation, online help, FAQs, and quick reference guides to all customer service channels, such as phone/chat, e-mail, and self-service Web sites.

- **Repurposing content across deliverables**
  With DITA, organizations can automate publishing into multiple formats for print, online help, or a Web site from a single source. This single source production of multiple output types eliminates the need to create and maintain content for each specific output format. Topics can be reused in various combinations for customized documentation.

  The DITA specification provides several content reuse methods, such as "conref," a simple text inclusion mechanism that lets authors easily reuse content stored elsewhere, and "conditional text," which lets publishers select only the content that relates to a specific audience or product. These features make it simpler to write, assemble, and publish customized documentation.

- **Ease of content updates**
  Most importantly, DITA provides the ability to "update once, change everywhere" — greatly easing the process of content editing and improving the organization's ability to keep all materials accurate and up-to-date. DITA makes it vastly easier to keep up with changes; when a product changes, editors need only update that topic, and wherever that topic appears, regardless of the format or language, the content will be updated. This eliminates low-value administrative tasks such as searching for and updating content in multiple places.

- **Translation efficiency and acceleration**
  Applying the DITA model has proven very effective in reducing translation costs and globalizing your information faster. By translating information in topics rather than documents, organizations can send content to be translated as each topic is completed. This significantly reduces delays because information is being translated in a continuous stream instead of in a last-

minute rush at the end of the authoring process. Additionally, with the proper relationship maintained between languages, no piece of content has to be sent out for translation twice, helping save on translation costs.

Next Steps

If customer satisfaction, loyalty, and retention are a strategic focus for your company, make the business case for a move to DITA for customer-facing documentation and communications. Besides the cost-savings realized in the call center or help desk by empowering customers to help themselves, you can demonstrate DITA's ability to improve the productivity of your current information development staff significantly. With the number of hours saved by using DITA and automated publishing, your staff will be able to spend more time creating additional valuable materials for your customers. For more on the financial impact of XML and DITA, download a white paper on this topic featuring calculations you can use in your business case at http://na.justsystems.com.

Another thing to consider as you make the move to DITA is the change that your documentation teams will go through. The shift from authoring whole documents to authoring standalone topics will require some getting used to, but organizations that have done so are encouraged by the increased productivity and quality. Your company might also consider upgrading your team's skill set to include content modeling and information architecture to leverage DITA and XML best.

When planning a DITA implementation, evaluate how your team will collaborate with others: engineers who may contribute expertise, lawyers who may need to be part of your approval process, and the localization team that is critical to globalizing your content. Other players in your organization will have to understand new processes and systems to optimize your content lifecycle.

Lastly, what systems do you need to work with DITA? The open standard is supported across several authoring, management, translation, and publishing systems that are capable of recognizing and handling XML and DITA content. Doing due diligence on your selection will best enable your company to transition to DITA, which is vital to a successful implementation.

Conclusion

Authoring and editing content using traditional systems is no longer a viable option for global businesses that have to produce high-quality documentation not only to keep up with but exceed customer expectations. The DITA framework offers information development teams a highly effective way to maintain the integrity of information across multiple channels, languages, and audiences while reducing the time and costs associated with information publishing. DITA is helping companies like Sybase, Tellabs, and Sterling Commerce, to ensure content accuracy and consistency in a world of multi-channel communications, enabling these companies to deliver better customer satisfaction through better documentation. ■

About the Author

Jerry Silver has over 25 years of IT experience as a developer, consultant, and product manager specializing in database and application modeling and design, application architectures, Web technologies, content management, and collaboration. He has been a featured speaker on these topics at numerous industry conferences. Jerry spent 15 years at Oracle in a variety of technical roles, most recently as principal product manager of Oracle Application Server Portal. He also served as director of product strategy with content management vendor NCompass Labs, now part of Microsoft. Currently, Jerry is director of product management with JustSystems Inc., responsible for XMetaL content lifecycle solutions and software.

Introduction to Complex Event Processing & Data Streams

– continued from page 24

- **Number of messages a second:** Use cases requiring Complex Event Processing typically demand that messages be transmitted at a rate of 1,000/second to 500,000/sec. This is understandable considering that the CEP application typically integrates with edge devices or, as in trading applications, process a large amount of market data from multiple exchanges to make trend inferences. In such cases high-performance middleware should be capable of intelligently compressing the messages to fit the given system's MTU. Vendors such as 29West and RTI provide performance numbers in this range. For example, with RTI the user can transmit up to 10 million four-byte messages a second.

- **Supporting heterogeneous platforms:** From one point-of-view, CEP engines are about integrating data streams from different sources. While different data streams can use their own messaging protocols, this poses a needless headache for application architects by having multiple technology stacks for their data stream implementations. Messaging protocols like JMS are supported on major enterprise platforms, but aren't prevalent in embedded ecosystems where edge devices reside. Traditionally, the Data Distribution Service implements a vast set of architectures including, but not limited to, enterprise platforms like Linux, Solaris, and Windows and embedded platforms like VxWorks, LynxOS, and Integrity operating systems.

Summary

Remember the mid-90s? Some people still assembled their own PCs by purchasing the right combination of motherboards, processors, and disk. But a non-optimal combination of motherboard and processor ensured that you didn't get the right performance from your system. Similarly, while CEP engines herald the promise of delivering high-performance event detection and generation, you need to ensure that they run with data streams with compatible aims of low latency, high availability, throughput, and flexibility. ■

About the Author

Supreet Oberoi is vice-president of engineering at RTI, with over a decade of experience in building and deploying Web-based enterprise applications. He was a founding member and director of engineering for Trading Dynamics, which was acquired by Ariba in 1999. Later, he led the engineering organization for the Mohr-Davidow (MDV)-funded start-up, oneREV, Inc., that was acquired by Agile Software in 2002. Most recently, Supreet served as director of engineering at Agile Software. He received his BS in computer sciences with highest honors from the University of Texas at Austin and an MS in computer sciences from Stanford University.

Just How Fit Is Your SOA?

Implementing SOA is like going on a diet, trying for the perfect figure

WRITTEN BY **JOHN SENOR**

So you've decided to invest in Service Oriented Architecture (SOA). You've read up on it and heard the experts proclaim its potential to transform the way your business interoperates. You're excited about SOA's prospects and what it will do to improve the fitness and agility of your company. The only thing left to do is take the plunge.

It's like going on a diet. Like a diet the road to hell can be paved with good intentions and high expectations. The quest for the "perfect figure" – in this case a cost-effective, high-ROI Service Oriented Architecture – requires a lot more than will power and well-meaning thoughts. It takes real knowledge of the pitfalls you may encounter and the seemingly tasty goodies offered by some software vendors you will have to resist. It takes understanding the factors that can impede success and turn your best intentions into another fad diet failure.

Like the celebrity diet that promise results with little or no effort software vendors make just as many promises about achieving dramatic results. Unfortunately, like most things, there's no magic elixir. However, simply by understanding the factors that make a SOA more difficult and more costly to implement, you can avoid setbacks and stay on track to achieve lasting and measurable results.

Fitness Tip #1: Beware of hidden "calories," costs, and complexities.

When you start a new diet or fitness plan, it's important to understand all the components and how they contribute to your goal of a South Beach physique. Otherwise, if you're not careful, you could end up losing more muscle than fat, feeling tired, looking haggard, or even seriously damaging your health.

The same applies to starting to implement a SOA – and accounts for why so many of these implementations disappoint. Like the hidden calories in many so-called diet foods, the complexities and costs associated with the technologies underlying your SOA implementation can wreak havoc with your ROI objectives. To put this in context, we need to go back a few years and look at the evolution of enterprise integration initiatives.

During the Enterprise Application Integration (EAI) era, we learned that large-scale monolithic implementations of integration technology ultimately did not best serve the needs of the business enterprise. They were simply too complex, too proprietary, and too costly to implement, manage, and maintain. Plus, each implementation was not reusable for other purposes. No matter whether the integration project was large, small, or in-between, the implementation style and cost never varied. You always had to contend with a large stack of proprietary products that were required to make the integration software work.

This software stack was extremely complex and costly, and difficult to manage and maintain. Consequently, this approach needed bigger machines, and more operational staff, training, and maintenance, simply because of the complex software involved. As a result, typical implementations could run as high as $1 million before any useful integration was realized. At that time, there simply was

no lightweight, low-cost, manageable, and reusable alternative.

SOA, as the next-generation vision of enterprise integration, now promises to improve on its predecessors as follows:

1. **Trim the amount of integration software needed.**
   SOA replaces complex, proprietary, and costly centralized integration software stacks with lightweight, standards-based integration software alternatives that take far less infrastructure software to implement, operate, and maintain.

2. **Utilize code, implemented as services that's not only easy to write and maintain but can be reused across applications and systems**.
   Say goodbye to tightly bound process orchestration code, proprietary tools, and runtime engines with little reuse potential across applications or systems. SOA takes advantage of loosely coupled integration code called services that can be implemented easily and quickly then assembled for use and reuse by any application or process that needs to do a similar function. SOA visionaries recognize that the know-how in implementing and maintaining services resided with the owners of business processes and applications, not with small groups of IT professionals who specialize in integration software tools.

Fitness Tip #2: Let the heavy baggage go.

If your SOA is falling short of expectations, it may be because most companies that supply SOA software base their service implementations on reworked EAI software, heavily dependent on an underlying application server/Message-Oriented Middleware (MOM) foundation. Coarse-grained services are implemented using tools tightly bound to the runtime environment of an integration broker that is, in turn, completely dependent on a proprietary J2EE server, MOM, and/or proprietary JMS.

Such monolithic stacks are expensive and complex to put in place and require sophisticated skills and training for IT resources. Configuration management is difficult too, especially if implemented on a large scale, which can be completely cost-prohibitive for the global business enterprise interested in large-scale, multi-location SOA deployments. That 's simply a much too complicated recipe to create and deploy a service that's reusable.

"Tired of all the hidden costs and complexities of your SOA implementation? Not achieving the hard ROI you were promised?

But what if a business enterprise has major investments in application infrastructure software stacks as a result of years of integration efforts?

Fitness Tip #3: Stay powerful, independent, and flexible.

SOA is based on the concept of distributed computing environments working in collaboration through shared services that aren't bound by the constraints of a specific J2EE container, method of operation, or form of transport protocol. Instead, services are developed and maintained on a distributed basis (where the business process expertise resides) and operate synchronously or asynchronously over any form or combination of transport protocols. There's no deployment or execution dependency on any proprietary J2EE stack, MOM, or specific form of service exposure (such as only deploying services as Web Services). Rather, related sets of fine-grained services aggregate into course-grained composite services at any number of logical points in the distributed architecture. These composite services may be:

- Deployed as (Unbound) Web Services
- Bound to specific synchronous or asynchronous transport protocols
- Bound across multiple forms of mixed protocol transports such as AS2, MQ Series, Tibco Rendezvous, BEA JMS, or even FTP.

It's only when services are deployed in this lightweight and highly flexible way that SOA can scale ubiquitously and an organization can achieve service independence.

Service independence enables a business enterprise to create reusable composite services on any scale easily then deploy them not simply as Web Services, but as service channels using any variety of transport protocols, including Internet protocols, proprietary messaging systems, or legacy transports. This is extremely important since most business enterprises simply can't afford to rip and replace existing investments in networking and integration software. And by fully leveraging the Java SE Platform (J2SE), SOA implementations won't drag additional, impossibly complex, and expensive prerequisite software products into the mix.

Just How Fit Is Your SOA?

Tired of all the hidden costs and complexities of your SOA implementation? Not achieving the hard ROI you were promised? Then consider putting your SOA on a new fitness program – one that's a simpler, more flexible, and more practical approach – that will deliver results for the long-term. Forget the fads and quick fixes. Select a service-independent SOA implementation with an open transport approach and your enterprise architecture will be running like a highly toned athlete in no time.

As for the diet, if it were only that simple!

About the Author

John Senor is the president of iWay Software. iWay provides rapid integration solutions that help companies large and small meet new business challenges without making their vital information investments obsolete. iWay's hallmark is its simplicity, based on the assembly and configuration of off-the-shelf components. These components include over 300 pre-built connections to virtually any backoffice system, including legacy systems and packaged applications. This simplicity enables companies to achieve rapid integration without lengthy custom coding or expensive systems overhaul, and implement mission-critical applications without costly delays.

Production Line Lifeline for Telecommunications Providers

Sizing "the box" and reducing complexity

WRITTEN BY **BRIAN NAUGHTON**

The ability to rapidly assemble new shorter-life services will be the key differentiator for telecommunications service providers (SPs) striving to beat out cable, entertainment, and Internet companies as they encroach on each other's customer base.

SOA and BPM offer potential solutions to this critical issue by bringing levels of business agility and responsiveness hitherto unseen in the IT environs of SPs.

As the lines blurred among telecom, entertainment, retail, and Internet domains, SPs were forced to seriously consider the state of their networks. Now with the network in place, SPs are instead considering how they can increase operational speed and responsiveness to customer demands.

The realization in hotly contested triple- and quad-play markets is that SPs must in fact become customer service providers (CSPs). They must make the transition from me-too services to truly converged, on-demand services that differ from those offered by MSOs and non-traditional competitors.

Services On-Demand

To achieve that end, CSPs will have to work with third-party developers to create scores, if not hundreds, of niche services that leverage their substantial investments in IP networks. After all, they laid the fiber to enable voice, video, and data to come together over the same connection in very short time frames. In theory, that unique ability will enable CSPs to create prodigious catalogs of converged services without disrupting the underlying architecture.

Carriers must shift from the staid and stodgy belief that services must take months, if not years, to roll out to a paradigm that enables services and products to be rolled out in hours, if not minutes. The only way to effect that mindset change is to drive the reuse of common data models, formats, naming conventions, interfaces, and design processes across the organization.

That need for reuse will drive carriers to break back-office silos down into components. Such components will represent operational elements of network and IT systems, as well as product, service, and resource specifications. These components can ultimately be turned into "building blocks" that are loosely coupled so that they can be used interchangeably among different services and products.

The ability to use building blocks interchangeably among applications requires a common framework in which carriers can segment operations and couple services in the spirit of Service Oriented Architecture (SOA). SOA requires an execution environment in which services are decoupled from networks for integration with business processes.

Once this type of integration is achieved, the environment can be used as a true service delivery platform (SDP) from which new functionality can be driven (i.e., SIP capabilities around presence,

location, and more advanced voice mail services that can be used in creative product bundles). By implementing common SIP servers for applications needing connectivity over IP networks, carriers can procure data from disparate sources so that billing authorization and billing detail are consistent across the organization.

As new services are created through increasingly agile SDPs and execution environments, CSPs will have to orchestrate changes simultaneously within OSS/BSS applications. The complexity of orchestration for dynamic services will require full automation of activation, ordering, and billing processes so that fulfillment and assurance processes can seamlessly work for new service rollouts.

Too Many 'Moving Parts'

In today's marketplace, the bottleneck for product management and design is usually the sheer volume of paper documentation and ambiguous hierarchies with which engineers, architects, and marketers have to grapple.

Rather than invest in software, companies tend to throw people at the problem. These people usually fail to resolve issues expediently because they continue to look at problems at the granular level. A granular approach requires intimate knowledge of the "nuts and bolts" of an organization. That is a major impediment to rapid service deployment.

Larry Goldman at OSS Observer believes that CSPs are now willing to "generalize" parameters in exchange for speed to market. For example, CSPs today possess as many as 30 or 40 pieces for their fulfillment system. As a result, dozens of applications have to be strung together to handle and track orders and interface to networks.

More often than not, distinct inventory systems are purchased for each technology type in the carrier environments – such as one instance for copper wires, one instance for DSL, and another for basic switching systems. On top of that, more systems are layered to track and manage services riding over those technologies.

Rather than manage so much complexity, carriers should strive to possess fewer "moving parts." The ability to handle a large range of services in a tight manner is more important now than disrupting the business with custom-made solutions running in isolation.

Thinking "Inside the Box"

To reduce the number of small pieces they must manage, CSPs must first ask, "What size box is the right size?"
Carriers are wary of the smaller boxes that must be strung together with enormous integration projects, but they are also wary of monoliths that take care of all aspects of fulfillment, assurance, billing, and customer care.

Most CSPs don't want to be held hostage, or surrender control to a vendor that can take its time with upgrades to accommodate the desired feature sets. Carriers would rather possess more capabilities for self-care and control of feature sets and services.

While IT and engineering grapple with the hierarchies and dependencies necessary for new products or services, it's the product managers that must ultimately introduce new services into commercial environments.

The main issue facing product managers is that whenever they suggest a new product innovation they have to wait months, or even years, for IT departments to assemble technical committees that try to embrace the business need. They gather details and slide presentations that attempt to detail for marketing the reasons why a new service either is or isn't possible.

Rather than endure such a lengthy process, product managers need tools that abstract the IT complexity for them by creating "objects" representing components of the IT and network domains of which they know little.

Such objects would be a foundation for an ecosystem built around common descriptions about pieces of fulfillment, assurance, and billing.

As consistency in terminology and protocols is established around how services are defined and what components are necessary to make services work, IT could better understand how components should be deployed and discrepancies easier to identify (i.e., multiple elements deployed to do the same function).

Simultaneously, consistent language would enable product managers to focus on their core competencies. For example, a product manager would be able to focus on whether a metro Ethernet service was priced by bit or bandwidth without having to understand the intricacies of the frame relay or ATM components it will replace.

Rather than the people, it's the "box" that should possess the intelligence about how to connect and what pre-requisites are necessary for launching particular services.

A "box," for example, that represents metro Ethernet should have the intelligence to try to connect to a voice mail service. In instances where IP routers or elements aren't specified, it should have the intelligence to ask for pre-requisites for launching that service.

Such intelligence relies on rules-based engines and computer-aided design top-end systems that allow for the definition of components with rules that can be associated to the services (i.e., what racks, cards, or slots are necessary in provisioning and activating certain services.)

If product managers could use such "boxes" to decouple management of product lifecycles from OSS, BSS, and network engineering, the time-to-market could be greatly expedited.

The Services Production Line

Adopting such an approach borrows heavily from the manufac-

"As new services are created through increasingly agile SDPs and execution environments, CSPs will have to orchestrate changes simultaneously within OSS/BSS applications.

turing industry, where computer-aided design (CAD) and computer-aided manufacturing (CAM) concepts helped drive mass production while retaining high levels of adaptability. The core principle was that once a design has been produced with a CAD component, the design itself can control the machines that make the part.

In the telecommunications world, if resource, service, and product components are created in a CAD-like way, each component definition will include full instructions for the automated CAM-like handling of that component. That way these components understand or reference the processes that live in the traditional OSS/BSS stack needed to deliver (Fulfillment) and manage (Assurance) that component.

So a product manager, or customer, could directly assemble an offering from a number of service building blocks without worrying about what the end-to-end process that delivers the offering will look like. The assembly rules intrinsic to each building block would dynamically figure out the process fragments that have to be glued together on-the-fly.

Enter "Active Catalog"

The Product & Service Assembly (PSA) Initiative, a TeleManagement Forum Catalyst project launched in mid-2006, set out to create an IT reference architecture that aligns service design and creation with service execution.

The output is a design-and-assembly environment, the "Active Catalog" that provides a framework in which service components can be defined and configured without needing to write any code.

In true BPM style, this opens the door to product managers who want to decouple the management of product lifecycles from OSS, BSS, and network engineering to expedite product rollouts.

At the heart of this SOA-based approach is a rich library of components and products through which product managers and architects can drive dependencies, prerequisites, exclusions, and visual metaphors about service components.

To ensure that there is an accurate model of the infrastructure, an Active Catalog has been designed to sit on top of most major network resource management systems (inventory) that serves as databases of record for carriers.

The key participants in the initiative, namely Axiom Systems, Atos Origin, BT, Cable & Wireless, Convergys, Huawei, Microsoft, Oracle, QuinetiQ, TeliaSonera, and TIBCO, were all involved in the initial definition of components that can be used interchangeably across services and functions.

With this level of support in the inventory space, as well as other facets of OSS, networking, and database management, Active Catalog is in a good position to influence the building-block approach.

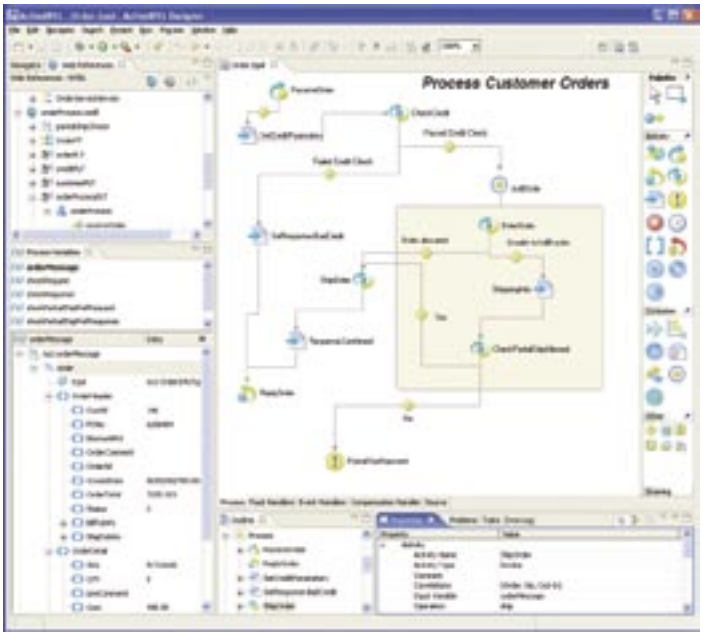
By providing a framework that can serve as the foundation for collaboration among product managers, service and network engineer, as well as operational communities, Active Catalog creates a central point for the standardization of multiple vendors' products that will allow CSPs to move closer to the SOA and BPM principles they strive to embrace. ■

About the Author

Brian Naughton is VP of Architecture and Strategy at Axiom Systems. Over the past 15 years, Brian has held a number of senior roles within leading IT and telecommunications companies focusing on the Information and Communications Technology (ICT) arena, with roles ranging from engineering to business strategy for prominent industry organizations.

bnaughton@axiomsystems.com

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

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

Altova® StyleVision® 2007 – The ultimate visual stylesheet designer for XML and databases.

ALTOVA® stylevision® 2007

Transform with style

Outfit yourself with StyleVision® 2007, and fashion multiple outputs from a single stylesheet design.

Spied in StyleVision 2007 Release 3:

- New Standard Edition specifically for XML to HTML transformations
- New Table of Contents creator with TOC wizard
- Support for reuse of StyleVision design fragments
- Support for Apache FOP 0.93

Altova® StyleVision 2007, the ultimate visual stylesheet designer, lets you transform XML and database content into eye-catching HTML, PDF, and Word/RTF output – all from the same design. Also use it to create intuitive electronic forms for Authentic® 2007, Altova's powerful, FREE XML and database content editor that enables business personnel to view and edit data without being exposed to the underlying technology. Get more out of your designs!

Download StyleVision 2007 and Authentic 2007 today:
www.altova.com

StyleVision is also available as part of the value-packed Altova MissionKit™ software bundle.

ALTOVA®