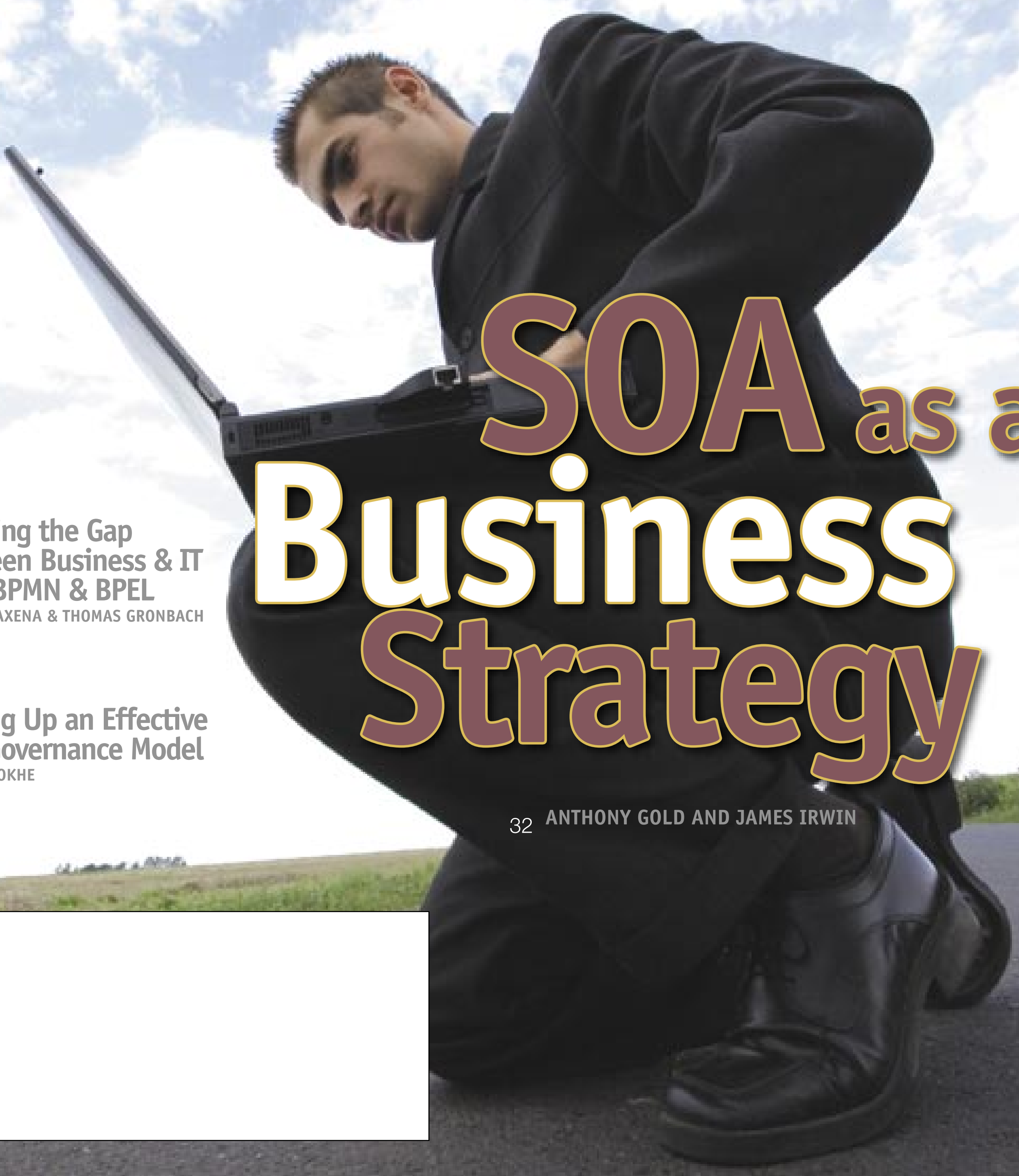
SYS-CON

THE WORLD'S LEADING MAGAZINE DEDICATED TO **WEB SERVICES TECHNOLOGIES**

www.SOA.SYS-CON.com

SOA WORLD MAGAZINE™

JANUARY 2008 / VOLUME: 8 ISSUE 1



SOA as a Business Strategy

SOAWORLD MAGAZINE
www.SOA.SYS-CON.com

EDITORIAL

Editor-in-Chief
Sean Rhody sean@sys-con.com

XML Editor
Hitesh Seth

Industry Editor
Norbert Mikula norbert@sys-con.com

Product Review Editor
Brian Barbash bbarbash@sys-con.com

.NET Editor
Dave Rader davidr@fusiontech.com

Security Editor
Michael Mosher wsjsecurity@sys-con.com

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

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

International Technical Editor
Ajit Sagar ajitsagar@sys-con.com

Executive Editor
Nancy Valentine nancy@sys-con.com

Associate Online Editor
Lindsay Hock lindsay@sys-con.com

PRODUCTION

LEAD DESIGNER
Abraham Addo abraham@sys-con.com

ASSOCIATE ART DIRECTORS
Louis F. Cuffari louis@sys-con.com
Tami Beatty tami @sys-con.com

EDITORIAL OFFICES
SYS-CON MEDIA
577 CHESTNUT RIDGE ROAD, WOODCLIFF LAKE, NJ 07677
TELEPHONE: 201 802-3000 FAX: 201 782-9637
SOA World Magazine Digital Edition (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

SYS-CON MEDIA
www.WSJ2.com
www.SYS-CON.com

Facing Facts

WRITTEN BY **SEAN RHODY**

Last month I wrote about the future, what might be ahead for SOA and beyond, focusing significantly on user interfaces. This month I'm still thinking about user interfaces and the impact they have on the final judge of any SOA project – the end user.

If you've read my editorials over the past few years, you're aware I'm an absolute opponent of the browser as a means of delivering applications. I think it's terrific at what it's made for – delivering content – and has been extended in ways that make it something its original inventors never dreamed of, but at the end of the day, it falls down horribly for delivering applications.

Now content is a big part of Internet usage, so I know the browser isn't going away, and I really don't want it to. What I think needs to happen are for new paradigms to emerge for application distribution.

The main reason people choose a browser for delivering applications is ease of distribution. You don't have to do an install on 10,000 desktops, you don't have to worry about some paranoid IT department's draconian security policies, and you don't have to send out countless updates when you patch a bug in the application. The advantages of zero footprint applications are clear and undisputed. But so are the disadvantages. Whether it's lag time for screen refreshes after each dialog choice, network latency issues, or the inability to work offline effectively, we all know that the browser is the least common denominator approach to application design.

When you think of all the bridges and obstacles SOA has already surmounted in providing vendor-agnostic, machine-agnostic, ubiquitous computing as a service, you have to ask yourself: "Why can't they do the same thing for my user interface?"

Now that's not a trivial question, I admit. And let's be clear – to be successful, we have to handle not just the desktop, but the palmtop, the laptop, the iPod, the GPS, and even the automobile. We can't focus on Windows anymore and assume that's good enough – that's the same thinking that gave birth to SOA in the first place.

We're already seeing novel approaches to this problem in the form of Web 2.0 technologies and mashups. One of the characteristics of this new user experience needs to be flexibility and individualization. Allowing users to work in a fashion that seems the most logical to them will result in higher productivity and application adoption among end users. Being able to pull disparate sources of information together and correlate them will make great strides in improving the quality of work of the average knowledge worker.

We're no longer a work force that's confined to our desks. This is a key learning that needs to be internalized into the IT world. Our users are eclectic – some are very traditional, but others are new users who grew up with technology and who've never owned a land line and consider text messaging a natural means of expression. We have a more mobile and technology-savvy community of users and it's up to us to stop doing business as usual and adapt to the changing times.

To do this we need to focus our attention and energies on how we deliver applications. We need to reform our very definition of what an application is and concentrate on removing the obstacles that stand in the way of the delivery of an application regardless of platform. As I said last month, I don't really care where the application runs – it can be on my laptop, my iPod, my GPS, or some future technology that doesn't exist right now. The point is I want to be able to use it regardless of how I am connected, what I'm doing, and where I am. When we have that down pat, we will definitely have achieved the last mile in SOA – taming the user interface.■

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*

A Look at Master Data Management as a Key Foundation for a Successful SOA

The issue of data inconsistency in an SOA environment

WRITTEN BY SID SURI

Implementing a Service Oriented Architecture (SOA) has the potential to offer tremendous long-lasting benefits to the enterprise. First, it reduces cost by offering a smarter, better way of developing applications by turning both existing and new functionality into reusable services.

The second SOA promise relates to composite applications that let developers leverage pieces of existing functionality and data that may be locked in independent applications and combine them to rapidly produce new business services or applications. For example, a bank may have data on a specific customer in the retail banking, home mortgage, and student loan branches. With customer information siloed by product, it becomes difficult if not impossible to upsell or cross-sell intelligently.

A composite application solves this issue by cutting across system and geographical boundaries and aggregating fragmented data. Along with these promises, however, SOA can also expose buried data problems that are problematic to its success.

This article will examine the issue of data inconsistency in an SOA environment and explore how master data management (MDM) is the key to reaping the full benefits from SOA.

The Origins of Data Inconsistency

Systems store data according to their own purposes using independent taxonomies and formats. Out of 100 to 200 possible attributes, a given system might only deal with the 40 or 50 relevant to its business. In addition, much of the enterprise data is stored outside of systems altogether in spreadsheets and paper documents. Each system has its own set of rules for entering new information, many of which involve manual entry and editing, a process that's prone to errors. This fragmented system-centric view of information, combined with ad hoc processes for updating information, leads to inaccuracies and inconsistencies, particularly in large distributed organizations. The impact of data inconsistency is felt throughout the organization, resulting in disruptions in critical business processes, a lack of visibility into business activities, inaccurate information being fed to decision makers, and a waste of resources as employees spend time resolving data errors.

CONTINUED ON PAGE 18

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

Advertising Events Associate
Alison Fitzgibbons alison@sys-con.com

SYS-CON EVENTS

Event Manager
Lauren Orsi lauren@sys-con.com

Event Planner
Sharmonique Shade sharmonique@sys-con.com

CUSTOMER RELATIONS

Circulation Service Coordinators
Edna Earle Russell edna@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: $99/yr (12 issues)
(U.S. Banks or Money Orders)

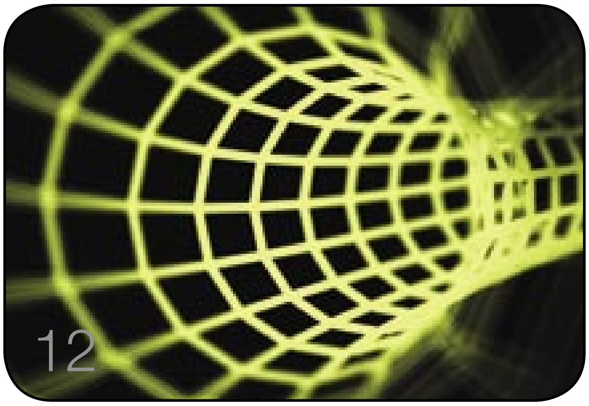
For list rental information:
Kevin Collopy: 845 731-2684, kevin.collopy@edithroman.com;
Frank Cipolla: 845 731-3832, frank.cipolla@epostdirect.com

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

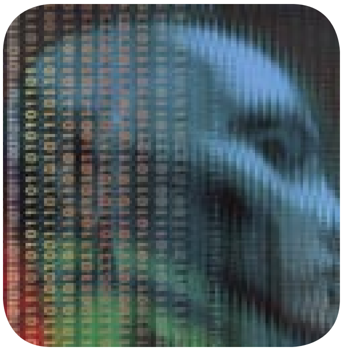
SYS-CON MEDIA www.SYS-CON.com

INSIDE SOAWORLD MAGAZINE

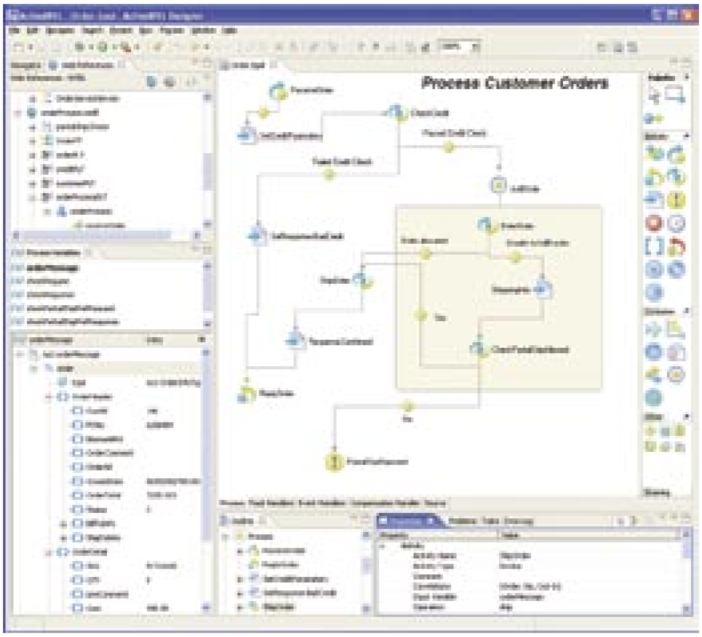
32



12



28


Process Customer Orders

Containers – Is It Time for Another One?

What's happening with OSGi and why you should care

WRITTEN BY KHANDERAO KAND AND DAVE CHAPPELL

The Open Services Gateway Initiative (OSGi) Alliance is working to realize the vision of a "universal middleware" that will address issues such as application packaging, versioning, deployment, publication, and discovery.

In this article we'll examine the need for the kind of container model provided by the OSGi, outline the capabilities it would provide, and discuss its relationship to complementary technologies such as SOA, SCA, and Spring.

Enterprise software is often composed of large amounts of complex interdependent logic that makes it hard to adapt readily to changes in requirements from the business. You can enable this kind of agility by following a Service Oriented Architecture (SOA) pattern that refactors a system into application modules grouped by business functions that expose their public functionality as services (interfaces). For example, a customer relationship management (CRM) solution would have modules such as sales, marketing, call center, and customer service. These modules use services exposed by other modules. To implement such a modular SOA pattern in Java-based enterprise systems there are often issues such as class conflict, unintended use of internal functions, interrupted upgrades, and multiple versions. Problems such as bundling, abstractions, dependencies, dynamic loading, lifecycle management, and side-by-side versioning are programmers' nightmares. Ideally containers should take care of these common issues, freeing application developers to solve the business problems.

Unfortunately many containers, including Enterprise JavaBeans (EJB), don't solve these issues satisfactorily. In this article, we'll explain how an Open Services Gateway initiative (OSGi) container would solve them. We'll begin with an introduction to the OSGi's solution to the problem, concepts, and platform, and then we'll delve into the evolution of the OSGi from its past in the world of embedded devices to its future in enterprise systems. We'll also explain the relationship between the OSGi and other initiatives, containers, and technologies to provide a comprehensive picture of the OSGi from the perspective of software development.

OSGi's Background

In March 1999, the OSGi Alliance was formed to provide a dynamic component model for service gateways to develop networked embedded systems.

The OSGi specifications developed provide a standardized, component-oriented platform for Java-based software. The platform solves the dynamic loading, versioning, and lifecycle management issues for Java-based services and also provides services to develop an ecosystem around it. In the last few years, OSGi has progressed far enough beyond embedded systems that its promoters are positioning it as universal middleware.

In retrospect, the use of "G" for "gateway" in the OSGi acronym is something of a misnomer. Universal middleware has a broader scope than just gateways, which are typically brokers on a network edge that provide interaction between external and internal networks. The use of the OSGi specification has broadened beyond embedded devices. This trend is noticeable in the alliance's supporting organizations too. Initially most of the supporting organizations were from the embedded space. Currently they've expanded to include major J2EE application vendors, ESB and middleware vendors.

Introduction to OSGi's Solution and Its Architecture

OSGi provides a component-oriented, lightweight container framework to host dynamically managed services. A module, called bundles in the OSGi, is a jar containing interfaces, their implementations, other classes, and a manifest. A bundle is a basic module for packaging, deploying, and managing the lifecycle of a group of services.

OSGi Bundles Provide an Isolation Model

Each OSGi bundle encapsulates packaged services and controls their visibility through metadata in a manifest file. As shown in Figure 1, a package or service in a module can be made available to other modules by explicitly exporting it. Other modules can't access any internal, non-exported interfaces or their implementations or other internal classes. In Figure 1, notice the import statements. Classes from other bundles can use these exported services by explicitly declaring imports.

OSGi Container Manages Lifecycle & Supports Dynamic Loading

Modules go through lifecycle events such as deploy (install), start, stop, redeploy, and uninstall. When a module (bundle) is being installed, the container checks all the dependencies (imports). If all the dependencies are satisfied, the module is ready to start. Figure 2 shows the lifecycle management of OSGi bundles. The OSGi runtime container provides a lightweight microkernel in which modules can be dynamically added or removed.

OSGi Natively Supports Dynamic Versioning of Services

During the long life span of enterprise software, business and technology changes. The software must reflect these changes and upgrade different modules independently of each other. For example, a company might have a customer data model (CDM) for its sales, order capture, and marketing systems. When there's an upgrade of the CDM, the company might choose to upgrade sales and marketing systems. However, upgrading the order capture system could have implications for business process or stability and so might not be upgraded at the same time, resulting in a need to support two versions of the CDM simultaneously.

The OSGi specification natively supports versioning with version attributes in the export and import instructions (see Figure 1) in the manifest files. It also provides additional control via arbitrary export/import attributes. The selection of an imported service or bundle is based on its version and attribute(s). OSGi also provides a unique isolation model in which multiple versions of the same service can co-exist and each service instance is isolated from the potential problems of class conflicts. A service consumer can get a reference to the correct version by specifying a version and other attributes in a filter as shown in Figure 3.

OSGi Provides a Service-Oriented Programming Model

OSGi also provides a programming model based on SOA. It's important to note that OSGi natively provides service-oriented programming in a dynamic environment. Unlike the RMI service registry, the OSGi service registry is native to the container – the services are automatically registered during the load without having to register them programmatically. A publisher can register a service using the registerService method in its BundleContext. Similarly, a service requester can discover a service using the getServiceReference method from its BundleContext. Unlike EJB or RMI, the services don't need to implement any remotable interface. It's all POJO!

OSGi Can Handle the Dynamic Lifecycle of Services

Again, in enterprise software you can add or remove modules. Since the container automatically manages the registration and deregistration, application programmers have to understand that services come and go at any time. OSGi provides a framework to track and react to lifecycle changes in the service, sending notifications of the registrations and deregistrations to a service layer that also provides a service tracker utility to help client programming in this dynamic environment (Figure 3). There's also a declarative model to handle lifecycle changes. Other containers like EJB/RMI don't handle such changes in lifecycle.

OSGi and Spring: Providing a Stable Proxy in a Dynamic Environment

The OSGi/Spring combination further enriches the solution to the dynamic lifecycle problem. Interface21 is already working to develop OSGi/Spring integration, and in March 2007 provided a first milestone drop for download. The current specification is version 0.7 and is still being evolved. The specification's stated goal is to provide an "easier path to build Spring applications that can be deployed in an OSGi execution environment and that can take advantage of the services offered by the OSGi framework."

The goal of the OSGi/Spring effort is to combine the best features of both OSGi and the Spring framework. This integrated framework inherits isolations and modularity, side-by-side versioning, dynamic deployment, and updates and dynamic discovery from OSGi. Spring provides an easy-to-use framework based on dependency injection (DI) and aspect-oriented programming (AOP), powered by various services.

Spring provides a simple, familiar programming model that

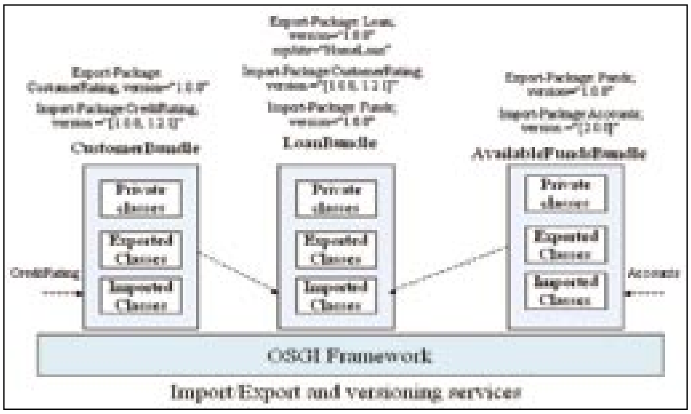

Export-Package: Loan; version="1.0.0" uplift="Fixed.rate"
Import-Package:CustomerRating; version="[1.0.0, 1.2.0]"
Import-Package: Funds; version="1.0.0"
Export-Package: CustomerRating; version="1.0.0"
Import-Package:CreditRating; version="[1.0.0, 1.2.0]"
Export-Package: Funds; version="1.0.0"
Import-Package: Accounts; version="[2.0.0]"
CustomerBundle
LoanBundle
AvailableFundsBundle
Private classes
Exported Classes
Imported Classes
CreditRating
Accounts
OSGI Framework
Import/Export and versioning services

Figure 1: OSGi bundles providing encapsulations, dependencies, and versioning

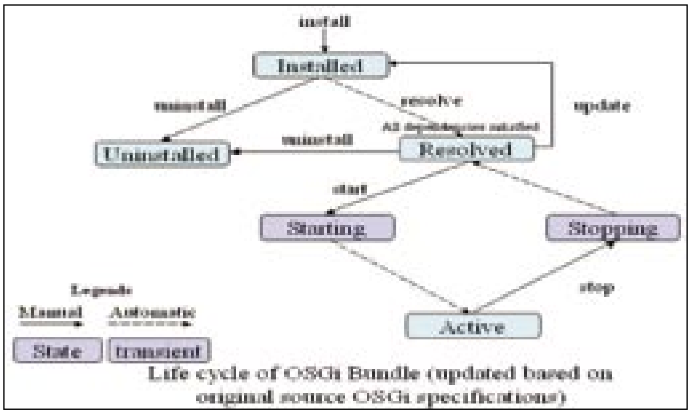


Figure 2: OSGi lifecycle management

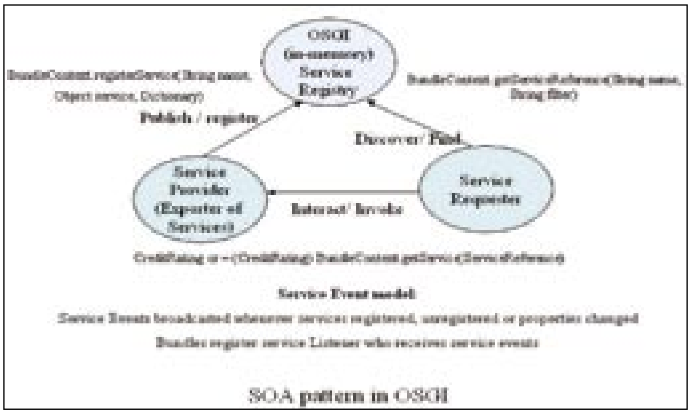

OSGI (in-memory) Service Registry
BundleContext.registerService(String name, Object service, Dictionary)
BundleContext.getServiceReference(String name, String filter)
Publish / registry
Discover/ Find
Service Provider (Exporter of Services)
Service Requester
Interact/ Invoke
CreditRating cr = (CreditRating) BundleContext.getService(ServiceReference)
Service Event model
Service Events broadcasted whenever services registered, unregistered or properties changed
Bundles register service Listener who receives service events
SOA pattern in OSGI

Figure 3: SOA pattern and the dynamic lifecycle of services

enterprise developers can use to exploit the features of the OSGi platform. The OSGi/Spring combination also manages the switching of services in the background. Spring provides a stable proxy for services that can dynamically come and go. If a service in use needs a behind-the-scenes upgrade, the Spring-based proxy lets you to bring down the service and replace it without a glitch in the stateless environment. In stateful Web Services, clients must track the lifecycle of the target service. The OSGi/Spring combination provides a service binder framework to track and react to the binding and unbinding of the target services. The Spring framework provides a stable proxy in a dynamic environment, and the OSGi platform provides dynamic lifecycle and versioning.

OSGi Architecture & Services

Now that you understand OSGi's capabilities, it would make sense to understand its architecture. The OSGi platform provides core framework and platform services. The core framework provides a runtime foundation to run and manage the lifecycle of various applications in a secured and modularized environment. The framework has four layers: security, modularization, lifecycle, and service.

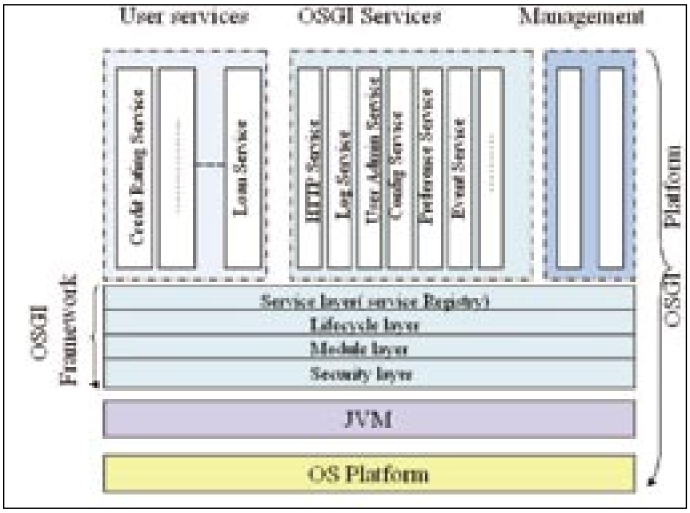


Figure 4:The OSGi Framework and Platform

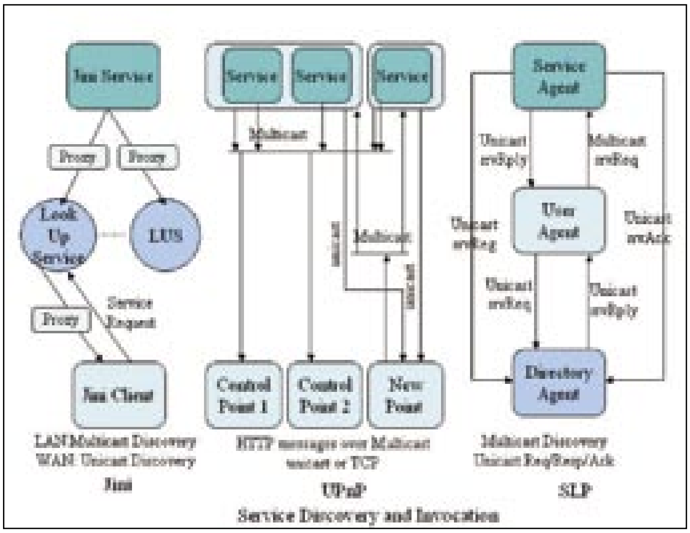


Figure 5:The Service Discovery Protocol options in support of distributed OSGi

The security layer provides additions to Java security that restrict the public and private exposure of packages and services and also provides permissions to import/export packages and register/access services from the service registry.
The modularization layer handles class loading, versioning, and the import/export of packages. It also manages the dependency resolutions of bundles.

The lifecycle layer manages the lifecycle of bundles and provides a generic API abstraction to enable remote management in a variety of management models.

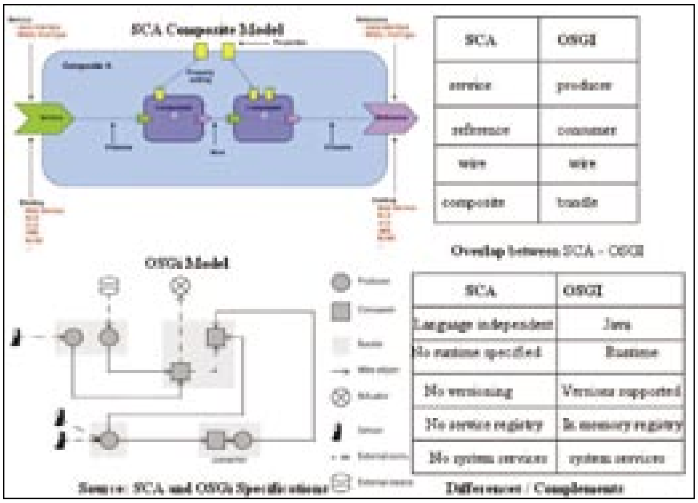
The service layer adds a dynamic behavior to the OSGi platform in which bundles can come and go. The heart of this layer is a service registry in which services are registered and discovered. The framework handles automatic registration and deregistration and triggers lifecycle events. OSGi further provides a declarative model to express service registrations and dependencies in an XML declaration. This declarative framework supports lazy (delayed) loading of resources by loading them only when they're actually needed.

The OSGi ecosystem platform provides various service interfaces that can be implemented by vendors, depending on the nature of their applications. The OSGi framework may provide a permission admin service, conditional permission admin service, a package admin service, and URL handler service. The OSGi Alliance specifies various system services such as a log service, and administrative services for managing configuration, event administration, users, devices, and applications. The Alliance has also defined an HTTP service so that bundles can provide servlets that can be made available over HTTP. Besides declaration and event services, in release 4, the Alliance defined a wire admin service to manage configuration-based connectivity between a service producer and consumers, enabling data objects to be exchanged over a wire.

Recent Momentum

Those who are using the Eclipse 3.0 framework are already using OSGi indirectly. When Eclipse-based applications became more complex, Eclipse needed an appropriate framework that could provide a dynamic, modular, and flexible runtime. The Eclipse group found that the OSGi framework satisfied most of their requirements. The Eclipse 3.0+ plug-in framework is based on OSGi. Eclipse's use of OSGi is now being evolved as the Equinox project.

Eclipse's use of OSGi caught the attention of the developer



SCA	OSGI
service	producer
reference	consumer
wire	wire
composite	bundle

Overlap between SCA - OSGI

SCA	OSGI
Language independent	Java
No runtime specified	Runtime
No versioning	Versions supported
No service registry	In memory registry
No system services	system services

Differences / Complements

Source: SCA and OSGi Specifications

Figure 6: Comparison between OSGi and SCA

community. Besides Eclipse, there are currently many other open source projects such as Apache's Felix, Newton, Knopflerfish, and OXSA. These open source implementations provide a meaningful way for developers and researchers to adapt the platform. Extensive adoption made 2007 a good year for OSGi. IBM and Cisco announced plans to develop a unified communications and collaboration platform called UC2 based on OSGi. And early last year Spring made its first milestone drop available to the public. Most of the leading application server vendors such as Oracle, IBM, and BEA either have an OSGi container in their stacks already or are working to add it.

With Eclipse, application servers, and other open source projects, OSGi has entered the mainstream to solve the modularization problems (dynamic modular deployment, versioning, and so on) of Java applications. The next step in its evolution is to provide interconnectivity for OSGi servers hosting services that can be moved from one server to another.

Future Evolution of OSGi Distributed OSGi: A Journey Towards Universal Middleware

The OSGi specification has, by and large, assumed an environment constrained to a single Java virtual machine (JVM) in that its service repository is referenced as local in-memory objects and the bundles are loaded and accessed via direct references in the same JVM. To be universal middleware, OSGi has to tackle the issues related to distributed computing, which it has started to do. There are a few key issues: How does one discover services on the network? And how can one invoke them?

To address the service discovery and remote services issues, OSGi's R3 specification included recommendations to use Jini and universal plug and play (UPnP). Recently, during the Eclipse 2007 conference, a third approach called R-OSGi was introduced. It's based on a discovery protocol called the service location protocol (SLP), a lightweight, decentralized, extensible protocol (see Figure 5).

However, each option has its own set of issues. The issues are mainly incompatibility between the remote service interfaces – Jini attributes don't match with OSGi, and Jini's reference implementation is based on the RMI invocation framework, which is perceived to be too heavy for OSGi devices. UPnP has a separate set of issues. Although UPnP is still recommended in OSGi R4, Jini is thrown out apparently because of issues such as interactions between class loaders and incompatibilities between security models. Although Jini didn't see a lot of momentum and releases, the specification provided a robust platform for some applications, including financial applications. Moreover, there's an open source initiative called Newton, and its commercial product, Infiniflow, which supports OSGi and Jini. Last year, after Sun open sourced its reference implementation, we expected some momentum, but it hasn't materialized.

The R-OSGi prototype uses SLP as a discovery protocol and provides a framework for service invocation based on dynamically generated proxies. R-OSGi also provides a way to define abstract proxies. During an R-OSGi demo, we observed that it has a very small footprint, but it needs some effort to make it robust. Moreover, the R-OSGi's service invocation protocol is optimized but proprietary. A standards-based approach would have been preferable. It seems that R-OSGi doesn't use RMI for performance reasons and issues related to RMI's distributed garbage collection when working with the OSGi lifecycle.

Service discovery and remote invocation doesn't provide complete distributed functionalities to OSGi. A new enterprise expert group was formed inside OSGi to look into the issues of taking OSGi to the enterprise level. The group has proposed works in areas of distributed registry, interaction with external systems, enterprise Web applications, and OSGi with J2EE, SCA, Spring, and so on. However, instead of charting yet-another new course, we believe the OSGi should reuse the appropriately relevant work already done in the J2EE and WS arenas.

OSGi & SCA: A Possible Alliance

Service Component Architecture (SCA) provides an assembly model for building composite applications using services implemented in a variety of technologies. SCA already has support from the major application servers and enterprise service bus (ESB) vendors. The SCA specification has been submitted to OASIS for standardization. Now work can be initiated for an integration plan for SCA and OSGi.

Conceptually both SCA and OSGi provide a composite model for assembling a services-based composite application that can expose some services to the external world as well as invoke external services. In OSGi R4, declarative services define a model to declare a component in XML, capturing its implementation and references. Besides SCA-like component-level information, the OSGi model captures additional information to control runtime behavior. For example, R4 provides bind/unbind methods to track the lifecycle or manage target services dynamically. SCA metadata defines wires between components or from a component to a reference in its composite model. However, SCA doesn't dictate a runtime implementation. OSGi defines wires in a composite model but lets the administrator add and delete wires using wire admin services (see Figure 6).

For OSGi-SCA integration, there are three possible scenarios:

- OSGi is a container for the SCA runtime
- OSGi and SCA are peers that can import and export each other's services
- OSGi as yet-another implementation type of SCA service

Where Do We Go from Here?

We'd like to see OSGi evolve, specifically in the enterprise market. It has definite uses as a microkernel in application servers. Other uses may evolve – service platforms are leveraging powerful combinations of two or more frameworks, such as OSGi, SCA, and Spring. OSGi would be useful for building SOA and enterprise applications, expanding OSGi's applications beyond embedded and network devices and microkernel and application servers. Some ESB vendors would find OSGi's dynamic services capability useful too. Different technologies working together with the OSGi could make a powerful platform to build large, complex enterprise applications.

About the Authors

Khanderao Kand is an architect of Oracle Fusion Middleware at Oracle. He is involved in the development of the Integration and Business Process Management Platform that includes BPEL and ESB. And he is on various technical committees for standardizing BPEL and SCA at OASIS. Previously he was an enterprise architect of PeopleTools and an architect in CRM.

Dave Chappell is vice-president and chief technologist for SOA at Oracle. He has more than 20 years of experience in the software industry. He is known worldwide for his writings and public lectures on the subjects of Service Oriented Architecture, the enterprise service bus, message-oriented middleware, and enterprise integration, and he is co-author of many advanced Web Services standards. As the author of Enterprise Service Bus (O'Reilly, 2004), he has had a tremendous impact on redefining the shape and definition of SOA infrastructure.

Bridging the Gap between Business & IT with BPMN & BPEL

Ensuring that IT will build what the business defined

WRITTEN **BY VISHAL SAXENA AND THOMAS GRONBACH**

Because the role of IT organizations is to enable business managers to run their businesses better, there has been a constant need for aligning IT closer to business. We often hear business managers complain that a software solution isn't what the business needed.

We have personal experience with this problem – in a previous life one of us worked for a consulting company that built turnkey applications for large enterprises. In such projects, a typical lifecycle starts with a set of consultants who visit the customer and capture requirements in a Word document. Better-run projects include screenshots of the future application in the requirements documents. Several months later, the software is delivered, followed by a lengthy maintenance phase. The budget allocation for the maintenance and enhancements of the application is typically several times higher than the initial budget for developing the application. And behold, more than 75% of post-deployment enhancements are requirements that weren't captured correctly in the analysis phase or originated after the analysis phase was over. To address these issues, it's critical for the gap between business and IT to close earlier in the cycle and that a model of the continuous business process evolution be supported to respond to the constant change requests made throughout the process development cycle.

An Outdated Business-IT Communication Model

Today's communication model between business and IT is limited and the omnipresent "waterfall model" with non-overlapping project phases contributes to these limitations. Besides the issue of capturing requirements that develop after the analysis phase concludes, it is also difficult to communicate originally identified requirements using the current set of tools – often Microsoft Word, Visio, PowerPoint, etc. A decade-old culture of static text documents and flow charts still provides the primary communication channel between business and IT. We believe one reason this ancient approach has been so pervasive is because any solution to this problem is complex. But also, the tools and standards that have evolved so far have really only worked well for one side in the process – either the business users or IT. Any robust solution must also address the UI, process logic, human workflow, and rules.

In this article we focus on a prospective solution to this problem based on industry standards to address the process part of the puzzle.

Shortcomings of Current Approaches

Most current approaches to this problem fall into one of the following categories:

- **Monolithic, Pure Play Tools –** These solutions provide a single environment, which is intended to be a design-time for business, a coding environment for IT, and a runtime to execute business processes. They promise a little bit of everything but often fall short when scalability, SOA standards, and integration requirements are taken into account. These solutions are also typically proprietary and entail vendor lock-in and a cost liability that can strain budgets because only product specialists can contribute to a project. Each tool often ends up as a functional silo in the specific organization that championed it and the proliferation of many such tools in a large enterprise slows their adoption. Also, the many small vendors with proprietary solutions in this category raise viability questions for many enterprises.
- **One-Way Road to BPM** – A second approach focuses mainly on business process modeling and typically does a pretty good job of that. Business managers like these tools since they offer all the bells and whistles one could imagine. These tools might generate Java or BPEL code, however, and once the process code is generated it acquires a lifecycle of its own and renders the modeling tools no longer useful. This leaves a gap because there's no guarantee that the final process implementation will look anything like its initial design. It also works only with a pure waterfall development model, which, as described above, has been shown to be inflexible and brittle and doesn't allow for continuous collaboration between business and IT for closed-loop process optimization.

BPM Standards

Standards organizations have been contributing to the lack of synchronization in the process development environment. The Object Management Group (OMG) defined the BPMN specification, which standardizes a visual notation for business process models – defining a system of graphic symbols for expressing process models. The Organization for the Advancement of Structured Information Standards (OASIS), another standards organization, defined the Business Process Execution Language (BPEL) standard, which describes how business processes can be executed. BPEL, with further enhancements in BPEL4 people, facilitates both system- and human-centric processes. This split between OMG and OASIS business process standards just encourages the gap between tools for business and IT – or the development of proprietary implementations. We should note that OMG saw fit to equip the BPMN standard with rules that detail how each BPMN activity maps to a BPEL activity – a start toward closing this model to execution gap. However, since BPMN allows arbitrary directed graphs, while BPEL is a more structured flow language, mapping between the two requires more than just mapping activities. It's also possible to create process maps in BPMN that are very difficult to implement in BPEL. Any solutions in this area must take these factors into account.

Fundamentally, we believe a key for contemporary, standards-based, continuous business process management is to bridge this gap between modeling (BPMN) and execution (BPEL, etc.) more effectively.

Process Blueprints

The rapidly maturing SOA standards have opened a new road to BPM. The solution we describe here provides for BPMN models tightly coupled with BPEL for process execution through a BPEL-based shared metadata format that we call a "process blueprint," enabling closed-loop business optimization. This approach allows for capturing process definition using best-of-breed tools supporting the widely accepted BPMN industry standard. The metadata is shared with IT as a process blueprint that basically defines an abstract BPEL process. This model may be thought of as the contract between the business and IT; it's the lowest level of modeling for business analysts and a living specification for the IT developer. The process blueprint describes an explicit business process layer, abstracting process logic from existing applications – enabling a more agile enterprise. It facilitates both business and IT having a common understanding of the process defini-

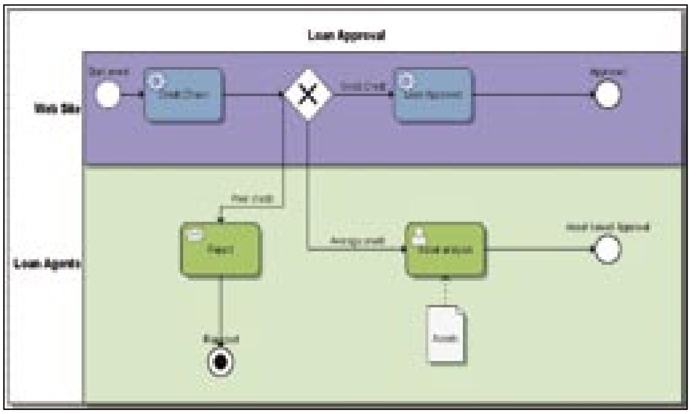


Figure 1:

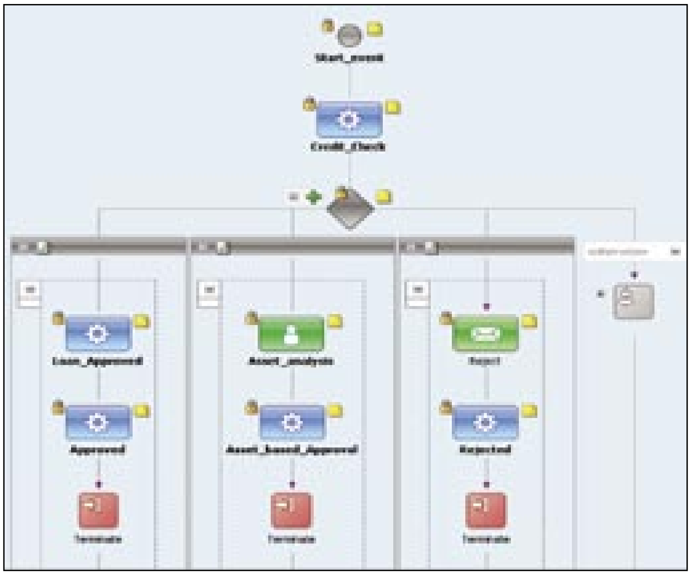


Figure 2:

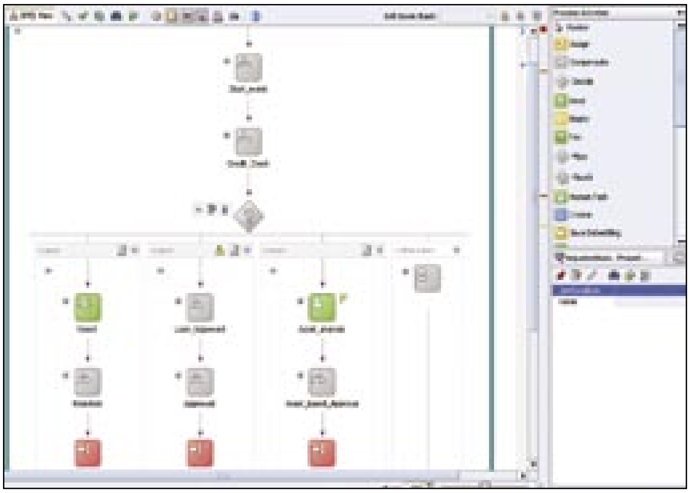


Figure 3:

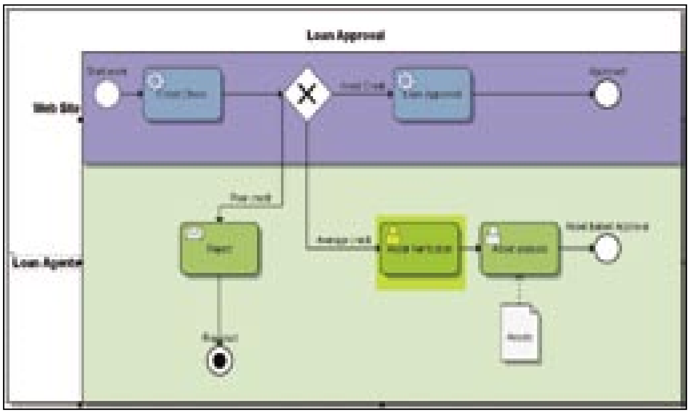


Figure 4:

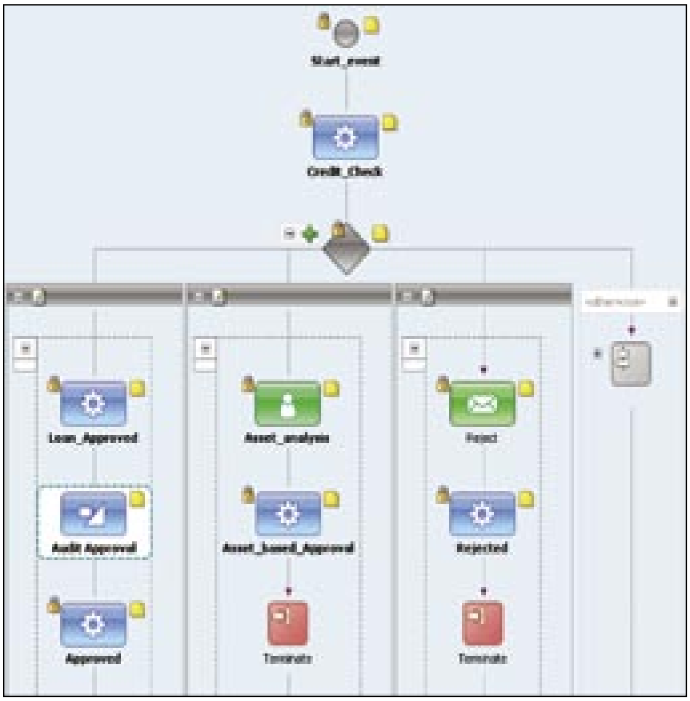


Figure 5:

tion. Additionally, closed-loop business optimization supports a model-driven iterative development approach and integrates the design and execution phase in real-time for synchronous process development between business and IT. Therefore it represents a better approach for continuous collaborative development, and helps overcome the discontinuous nature of the waterfall model.

Benefits of Closed-Loop Business Optimization

With that proposed solution, business users can document their business processes in a common, understandable notation across the enterprise, guaranteed by the standard modeling notation BPMN. These process models can be maintained in a distributed fashion, published to a wide audience and continuously monitored and optimized. These business process assets not only promote common understanding but also represent strategic differentiation for many companies. Most importantly, those business process models represent input for implementing and executing business processes, the underlying foundation for iterative closed-loop business process optimization, which represents an innovative tools approach for business process development.

With closed-loop business optimization, business processes originated by business users in the form of models are immediately available for IT to implement. Requirements definitions are captured through the model's standard metadata and are transparently viewable and editable by IT. Organizations can then tailor their applications as services, which can be deployed, discovered, and invoked, and potentially shared across many executing processes. With SOA standards like XML, WSDL, UDDI, and BPEL, deployment of services and their discovery and orchestration has become mainstream. This lets IT use tools that are right for the job and retain their investment in existing platforms, hardware and software, and people skills. A BPM infrastructure on top of service-oriented system interfaces unlocks the potential of IT assets, and enables the processes that orchestrate them to be changed more easily.

Business User's View

As an example, look at the business user's view of a simple loan approval process. A business analyst defines the process as:
1) Perform a credit check
1.1) Recommend approval of the loan for good scores,
1.2) Do an asset analysis and recommend approval of the loan for average scores,
1.3) Reject the loan for poor scores

The business user can create the process definitions using BPMN notation as shown below in Figure 1.

Business analysts draw business processes using standard BPMN artifacts. They identify process boundaries using pools, specify participants and assign them lanes, and draw the process diagram spanning multiple lanes. These processes may also communicate with other processes using message flows. The models must also be published to end users simulated and optimized.

Share Process Blueprint with IT

When this process is shared with IT for implementation, a process blueprint is published. A process blueprint for this process could be visualized for IT as shown in Figure 2. You can see that a start event maps to a start event, an automated activity is mapped to an automated activity bound by a scope, a human task is mapped to a human task, and an end event has been mapped to

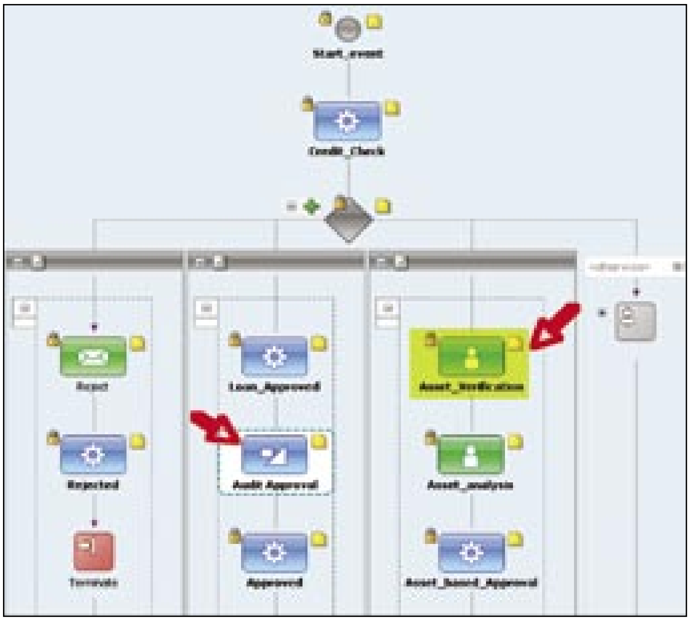


Figure 6:

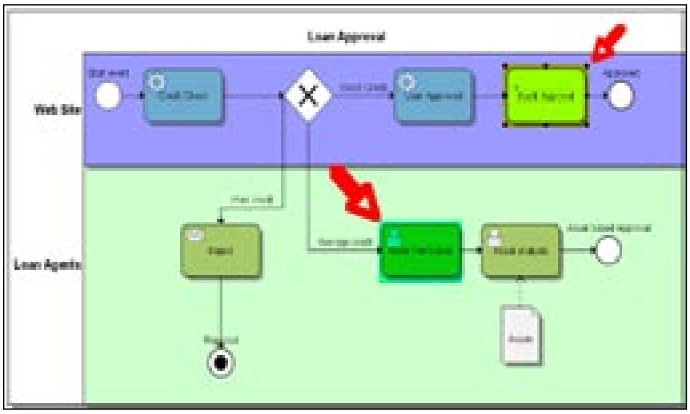


Figure 7:

an end event. For branches with an abrupt end, there's a "terminate."

IT User Implementation View

The IT user gets the full freedom and power of a BPEL designer to add implementation details to the process. Scopes can be expanded and all necessary IT details such as data mapping using BPEL assign or XSLT transforms can be done. Service invocations can be bound to concrete WSDLs and endpoints, variables can be defined and, if the service is asynchronous, users can define correlations and add receive statements for messages, faults, and service callbacks.

IT implementations typically have more steps and detail than the business view of a process. For example, the implementation of the credit check step could entail:

1.1. Map data to the interface expected by the credit check service
1.2. Invoke the service itself
1.3. Handle faults explicitly thrown by the service
1.4. Handle system exceptions such as service timeout
1.5. Map response data between formats (for example, fixed length file formats to XML)

The implementation process will usually uncover additional questions that business and IT must collectively answer. What happens if the service is unavailable for an extended period of time? Should the implementation continue to retry, log an exception, queue up a human task, or do something else? Is compensation logic required to "undo" the service invocation if the process is cancelled? Another important question to consider is how hard it would be to switch the implementation to use a different credit agency, if required by the business. A service-oriented implementation approach with well-defined interfaces should make this quite straightforward.

Business Processes - Evolved by Both Business and IT

It needs to be understood that business processes are living assets. Business and IT might update a process simultaneously according to their requirements. For example, a business user might update a process while an IT developer is adding implementation details. IT developer refinements involve providing physical bindings and transformations, but can also include adding additional implementation-specific processing steps within, or contiguous to, what the business analyst considers a single step. An approach of IT "refinement" respects the logical model as a constraint, enabling the logical model and the physical model to evolve in parallel without breaking the connection between the two.

As such, a BPMN-BPEL bridge has to allow for concurrent updates by business and IT, constantly granting real-time insight into the new state of the business process definition. The following section outlines such scenarios.

Business Changes

Business may add a step called "Asset Verification" before "Asset Analysis." This new step needs to be communicated to, and then implemented by, IT.

IT Also Changes

At the same time, IT may have realized that, as per its compliance initiative, every loan approval has to have an audit log. So the IT user adds a step called "Audit Approvals" after "Approval" as highlighted in Figure 5.

Merging Business and IT Changes

Now we meet the classic problem of the process definition having been modified by both business and IT at the same time. IT changes that are encapsulated in a limited scope can be seamlessly incorporated into the updated blueprint model. A smart merge algorithm can also be designed to identify blueprint changes made by IT users, if allowed. These changes can be tagged to the activities before and after. The merge process will thus take the process blueprint from a business user, tag the IT changes in their correct location, and further populate the blueprint with all the implementation-specific details. A merged process blueprint would look like Figure 6.

In Figure 6, the "Audit Approval" step is added to the branch after the "Approve" step as proposed by IT. And the "Asset Verification" step has been added before the "Asset Analysis" step as specified by business. The key to doing this as an automated merge is to limit IT to process refinements and share the common blueprint model format on both sides.

CONTINUED ON PAGE 19

A Look at Master Data Management as a Key Foundation for a Successful SOA

CONTINUED FROM PAGE 4

The Consequences of Data Inconsistency in an SOA

The problem of data inconsistency isn't new, but in an SOA environment it's much more prominent and potentially debilitating. In a closed environment, it may not have been a priority to the organization that its SAP data was inconsistent with Oracle or Siebel system data. With composite applications, however, inconsistencies become more problematic. In the retail banking example above, it's necessary that a customer's name be represented consistently across systems or it will be difficult to identify that the accounts belong to the same person. Similarly, if a company is ordering the same product from an SAP system and an Oracle system and it's not recognized as the same product, they might miss out on strategic sourcing benefits. Until data is reconciled syntactically and semantically, organizations can't reap the full benefits of SOA.

Why EAI Is Not Enough

Syntactic reconciliation, which refers to bridging the gap between the different data formats that individual systems use, is done by providing transformation maps from the data format of one system to another. The technology to address this is quite widespread and well understood. Enterprise application integration (EAI) technologies, however, don't solve the semantic integration challenges. EAI moves data where it needs to go, but it can't determine if the data is accurate or consistent or resolve conflicts in the data. EAI is charged with keeping systems in sync, but once they're out of sync it has no way of understanding or resolving the problem. EAI provides essential functionality, but inherently presumes that semantic consistency is already in place, which is often not the case.

Reconciling Semantic Data Inconsistency with MDM

Master data management (MDM) software enables organizations to create a single source for master data assets (products, customers, vendors, etc.) across multiple systems, departments, and trading partners. It also enables organizations to support the necessary processes, policies, and procedures to ensure that information remains accurate and consistent as new information is added. Furthermore, it ensures that clean master data is disseminated back to the transactional systems and decision makers to ensure that the data in an organization is consistent across all systems and geographies.

In addition to driving consistency, an MDM solution also serves SOA projects by making master data available on-demand to any consuming application, including composite applications, via standards such as Web Services. It lets organizations create a master information services platform – a set of services that can be invoked in a standard way by consuming applications and processes. These services can be granular data services, such as 'Query Record' and 'Delete Record' or they can include business logic such as 'Add Customer' and 'Update Address.' This enables developers of business services and composite applications in an SOA to focus on delivering high-value business functionality rather than worrying about how to access the underlying data. As developers working on one part of the organization start accessing data from another service, they'll require assurances in the form of service level agreements (SLAs) that the underlying master data will be accurate, consistent, and available in the appropriate format and in a secure and reliable manner. This level of trust and the ability to provide internal SLAs is critical in achieving the type of benefits promised by SOA.

To build these master data services, two things become increasingly important: data governance and metadata management.

Data Governance

Just as there's a need for SOA governance to ensure proper levels of security and performance for services, information governance is also necessary to resolve issues such as who can view or edit the data. If there's a conflict between customer data in Siebel and SAP, which system wins? What is the process for adding a new customer or new account? What are the valid values for each attribute? All of these rules – and many more – comprise data governance. Organizations can clean up their data, remove inconsistencies, and spend a fortune on consultants, but without effective governance, they won't achieve lasting data accuracy. Given the vast scope of master data, fixing it can't be an IT mandate. Data governance must be incorporated into day-to-day business processes and become the responsibility of the natural business users and owners of that information. For example, a person handling logistics is most concerned with product dimension data; IT should therefore provide the tools to effectively maintain that information to the logistics contact. Besides increasing data accuracy, efficient data governance ensures that companies are staying compliant with regulatory mandates pertaining to securing and archiving information.

ERP Metadata Management

Managing metadata is an integral part of both SOA and MDM initiatives, since, in essence, it forms the DNA of information architecture by describing the data model. Metadata stores crucial information about other aspects of the data such as mappings, validation rules, and business processes. It not only defines the structure of the data, but also serves as a "cookbook" for processing that information as it moves across the organization. So it's important to have a systematic way of creating, updating, and sharing metadata.

Enabling Data Services

Metadata also plays a crucial role in enabling data services. Master data services that provide standards-based access to underlying master data should be built on top of metadata or be "metadata-driven" rather than hard-coded so services can adapt to changes in the underlying data structure. This will also let developers of business-level services focus on building the value-added functionality their service provides.

For example, a developer building a cross-sell composite application might need to build a service called "Get Customer Credit." Ideally that developer would have access to master data services such as 'Get Customer Name' and 'Get Customer ID' as building blocks, greatly facilitating productivity. It would significantly slow application development and defeat the purpose of moving to an SOA in the first place if that developer had to reinvent the wheel to understand the structure, validation rules, and policies that govern the underlying customer data every time a new service was rolled out. Equally important, data services also protect higher-level services such as 'Get Customer Credit' from breaking when the underlying data structure changes, similar to the abstraction that an adaptor provides when interfacing with an application.

SOA + MDM – The Integrated Enterprise

For many organizations, the ultimate goal is to create an integrated or real-time enterprise. SOA connects people, processes, and information by integrating systems and providing a platform to develop new functionality while getting the most out of existing investments. MDM provides the information management component – ensuring that critical information assets are aligned internally and across the value chain. The combination of SOA and MDM allows organizations to be connected in real-time at nearly every level, from processing day-to-day activities to making strategic decisions. It gives IT organizations the tools to boost productivity and cut application development costs. More importantly, it gives them the agility to rapidly roll out new solutions that take advantage of accurate, up-to-the-minute data about their products and customers. ■

About the Author

Sid Suri is product marketing manager, MDM for TIBCO Software, Inc. He is responsible for taking TIBCO's MDM products to market, from developing overall go-to-market strategy to offering customer and sales support. Before joining TIBCO, Sid was an iSolutions marketing manager at C-bridge, the technology consultant that pioneered the delivery of trusted information-sharing solutions. Prior to C-bridge, he was director of technology at Cambridge Executive Enterprises, a leading provider of executive education programs. Sid has an MBA from Haas Business School, University of California, Berkeley as well as a BA in economics and Italian from Middlebury College.

Bridging the Gap between Business & IT with BPMN & BPEL

– CONTINUED FROM PAGE 16

The same algorithm is used to circle the IT changes back into the BPMN diagram. If business users don't want IT to change the process definition directly, the changes could be added to the BPMN diagram as suggestions and the business users can accept or reject them. If IT is allowed to submit process changes directly to the BPMN diagram to ensure the consistency of implemented models, they are communicated to the business users as soon as the changes could be reflected without approval. This can be implemented as a configurable option for users to pick what kind of governance they want. An updated BPMN diagram would look like Figure 7.

How Is This Done?

The process blueprint is owned by business, but IT is still allowed to make vetted changes.

Every business activity or task is encapsulated as a BPEL scope. This gives the IT user the freedom to add implementation details to that specific activity. The IT user can add BPEL constructs like assign statements, call out to concrete WSDLs, and specify retry parameters and intervals, compensation logic, and all the other details required to execute a process reliably. The business user's inputs, outputs, and service selection, if any, are available to IT users, enabling the business analysts to take the modeling process as far as they choose to go.

Implementation details are provided and owned by IT allowing them to be preserved across evolutions of a business process blueprint. However, business user changes to the process blueprint are cardinal to the sanctity of the process. If the business user adds, deletes, modifies, or moves an activity, that change is deemed to be correct and the process blueprint follows that change. All IT specific refinements move along with the business user's mandate.

Summary

BPMN and BPEL together can provide a fully standards-compliant solution that supports both business and IT. There are, however, differences between these standards, making validation tools and other constraints important on both sides. It's also important to note that the business analyst models are layered – and not all layers are meant to be executed. However, with a BPEL-based process blueprint approach, detailed BPMN diagrams can be shared with IT for implementation and execution. When the two sides share common metadata, this approach provides high-fidelity preservation of business intent for process definition. And the killer feature for this approach can be a dramatic increase in probability that IT will build what the business defined.■

About the Authors

Vishal Saxena is a product development manager for Oracle Fusion Middleware. He currently leads the development of the Oracle Business Process Analysis (BPA) Suite. Vishal has more than 12 years of extensive experience in enterprise software development, integration and BPM. In addition to leading development teams in multiple geographies, he is an evangelist for Oracle's BPA Suite.

Thomas Gronbach joined Oracle as a principal product director responsible for product strategy and global marketing for Oracle Fusion Middleware. Prior to Oracle, Thomas worked for Fujitsu Computer Systems and was responsible for developing go-to-market activities and marketing strategies for Fujitsu's SOA products. During a previous tenure at Oracle, he managed product marketing for the Oracle E-Business Suite. Before his stint at Oracle's headquarters in California, he managed technical sales for its global accounts in Germany. Before joining Oracle in 1993, Thomas worked for Hugo Boss in Germany as an IT project manager. He has a bachelor's degree in business administration and computer science from the Berufsakademie in Stuttgart, Germany.

The Evolution of SOA Hinges on an Open Marketplace for Services

The law of far-reaching effects

WRITTEN BY CHRIS HARDING

We can't foresee the full potential of Service Oriented Architecture any more than Henry Ford could have foreseen the scale of today's automotive industry when the first Model T rolled off the production line nearly a hundred years ago. But there are signs that it will be equally far-reaching. If so, this will because of the same force that made Ford rich: the open market.

Service Oriented Architecture (SOA) is an architectural style in which the software applications are organized as loosely coupled services. It's being adopted by many enterprises today because it results in agile IT systems that are easier to adapt to change. But it has another capability that could be more important. It lets an enterprise use externally provided services and choose between similar services provided by different suppliers. It enables market competition.

This can bring major benefits and will do so for many of the competitive services that SOA opens up. But it can also bring major changes; large-scale adoption of SOA may bring about a revolution in the way enterprises are structured. This is something that corporate planners should start thinking about now.

The Market Effect

Markets result in improved products and larger-scale operations.

Product improvements come from competition. You can get away with "Any color so long as it's black" only if there's no competitor offering a choice of color. If the customers want something and the suppliers can provide it at reasonable cost, competition will force them to do so. The Model T died because competition upped a car's performance, reliability, comfort, and safety.

This is not to deny the value of good design. But there's no way that anyone in 1908 could have sat down and designed an automobile of 2008. The factors underlying modern design emerged slowly and by natural selection.

Standards enable markets. For the automobile, enabling standards define the fuel used and the way the controls are laid out. Some standards, like fuel, may be precise and formal, perhaps legally enforced. Others, like the brake being to the left of the accelerator, may simply be unwritten conventions that everyone follows; but are nonetheless important. These standards let users switch brands and provide the basis of market competition.

When products improve, more people buy them. This leads to economies in scale in production and delivery, so products become cheaper – which means more people buy them. This is the virtuous circle familiar to manufacturing and applies to software too.

It was this virtuous circle that led to the massive growth of the automotive industry and made cheap personal transport available to most Americans. It has changed the quality of life and been a major agent of social change.

The application of the same market forces through SOA could lead to equally significant improvements in the way that organizations interact and change how they are structured.

Markets in Online Services

Conditions are now ripe for the market for business services delivered via the Internet to grow.

Amazon and eBay are leading examples of this trend. Much has been made of the revolution they created in delivering services to end users, but they've possibly achieved a greater revolution in delivering services to businesses. They provide sales and marketing services that other companies can use to reach end customers. They even, through PayPal and Amazon Payments, provide payment services.

Software as a Service (SaaS) is a part of this trend too. It means that a provider hosts and operates a software application for use by its customers over the Internet. Analysts expect that SaaS adoption will grow. For example, Michael Maoz, the analyst, believes that while SaaS currently accounts for just a small percentage of software sales in eight or 10 years it might reach 30%.

But SaaS isn't the whole story. While many business services rely on software to some extent – and that extent is increasing due to techniques such as intelligent agents that enable software to do many tasks traditionally assigned to people – few of them are implemented entirely in software. And services of many kinds – however implemented – can be delivered by one business to another via the Internet.

The markets in online services are developing. The number of business services delivered over the Internet is growing, and we are beginning to see some competition. The launch of Google Checkout in June 2006 was trumpeted as a rival to PayPal. Such rivalry can only improve online payment services to retailers and their customers.

And the fundamental market-enabling standards are now falling into place. These include formal standards such as the Internet Protocol (IP) and the Simple Object Access Protocol (SOAP). They also include the principles of service orientation. This is not a formal standard – how would you frame a formal definition of loose coupling? Nevertheless, service orientation is becoming a convention that everyone follows.

The Role of SOA

SOA is being adopted by many enterprises today because it delivers agility. It does this by virtue of the ease with which basic software services can be recombined to deliver new business capabilities and because there are powerful tools that take advantage of its clear structure to speed software development.

Many SOA services will be Web Services that conform to SOAP and perhaps other Web Services standards such as WS-Messaging and WS-Security. It's through such Web Services that an enterprise can expose its business capabilities in a standard manner. But SOA doesn't just mean using these formal interface standards; it also means adopting the principles of loosely coupled services that relate to business activities. It's difficult to interact with an enterprise that uses Web Services standards but doesn't follow the principles of service orientation, just as it would be hard to drive a car with a brake pedal in an unusual place.

SOA does more than give an enterprise internal agility. It means that an enterprise can easily switch between different providers of the same service. Because of this, the widespread adoption of SOA by organizations that use each other's services encourages competition. This works not only for pure software services, but also for business services that have interfaces that are exposed as Web Services.

Take a credit rating service, for example. In the past, the interface to such a service would have been by letter post, or perhaps by telephone or even e-mail. Now the interface can be an exchange of SOAP messages. The service may be provided entirely by software or may still involve people, but the interface is exposed as a Web Service enabling it to be integrated into a service-oriented software architecture. This makes it easy for a lending company to outsource credit ratings, rather than doing them in-house, which in turn gives the lending company a choice of suppliers and lets it concentrate on a core business area, perhaps credit-card administration or marketing.

Competition in services is the start of the virtuous circle of better features and increased adoption, delivering prosperity to service providers and satisfaction to their customers.

The Impact on the Enterprise

The ease with which enterprises can obtain business services over the Internet will have a big impact on the way they're organized. Sales and marketing were traditionally regarded as integral to the business, but Amazon and eBay have shown how easily they can be provided externally. There are growing possibilities for outsourcing other parts of the business, such as logistics and customer support, and there's no reason why they shouldn't extend to accounting and human resources.

This means that companies can concentrate on what they do best. For example, a widget maker may make excellent widgets but have poor sales because of poor order processing. It can outsource order processing to a specialist company and concentrate on making better widgets.

It will become natural for start-ups to develop this way, as small organizations that focus on their key business ideas and use external providers for the support services they need, only bringing those services in-house when there's a compelling reason to do so.

But what of existing organizations that started when companies provided most of these services for themselves and have grown into major corporations with hosts of interdependent departments? They must adapt to compete.

Can they do so?

An enterprise's IT systems are often a major barrier to change. IT enables an operation to be done in the same way over and over again very cheaply. Enterprises have built their IT systems with the idea of reducing the cost of their operations, little thinking those operations may have to evolve. Cost reduction through IT has been a big success story, but the result has been lack of agility. There have been well-publicized cases where an agreed merger of business organizations had to be abandoned because their IT systems were incompatible. Equally, the dependence of a core business IT system on the systems used by a supporting department can make it difficult to outsource the services provided by that department.

The adoption of SOA can help an organization overcome this problem. It can give it greater agility, not only in changing what it does but in changing its structure. A CEO who decides that a new enterprise architecture is needed to deal with new competition will find it easier to make that change if the IT architecture follows SOA principles.

Time to Plan for the Future

An open market can lead to undreamed of improvements in products and increases in the scale on which they're produced, delivering major benefits to society – provided conditions are right. We now have the right conditions for the application of this principle to business services. This will lead to major benefits for society through the improved operation of business enterprises – but it will produce major changes in the way that those enterprises are structured.

The virtuous circle of larger scale resulting in improved products is already appearing in SaaS. Liz Herbert, a senior analyst with Forrester Research, describes how SaaS enables community-driven improvement by incorporating one customer's successful customization into the product base and using web forums for customers to share best practices.

With widespread acceptance of Web Services standards and the principles of service orientation, we could see explosive market growth in many business services delivered over the Internet. New enterprises can then adopt lean business architectures and focus on their key business. Existing corporations will have to restructure to compete.

Although the trends are recent, they're beginning to take root. Large corporations take a long time to restructure. They should start planning now.

About the Author

Dr. Chris Harding leads the SOA Working Group at The Open Group - an open forum of customers and suppliers of IT products and services. In addition, he is a director of the UDEF Forum and manages The Open Group?s work on semantic interoperability. He has been with The Open Group for over 10 years. Dr. Harding began his career in communications software research and development. He then spent nine years as a consultant, specializing in voice and data communications, before moving to his current role. Recognizing the importance of giving enterprises quality information at the point of use, he sees information interoperability as the next major challenge, and frequently speaks or writes on this topic. Dr. Harding has a PhD in mathematical logic, and is a member of the British Computer Society (BCS) and of the Institute of Electrical and Electronics Engineers (IEEE).

c.harding@opengroup.org

Handling Attachment Payloads in SOA

Gearing up SOA for data-intensive operations

WRITTEN BY DEEPTI PARACHURI AND UJVAL MYSORE

SOA (Service Oriented Architecture) is an architectural paradigm that aims to achieve loose coupling and reuse among software components. It's emerging as the main integration and architectural style in today's complex software infrastructure.

Web Services aim to provide interoperability in machine-to-machine interaction over the network. XML-based open standards like Web Services Description Language (WSDL), Simple Object Access Protocol (SOAP), and Universal Description, Discovery, and Integration (UDDI) are used to achieve interoperability for defining, publishing, and invoking Web Services. Two applications, regardless of operating system or programming language, communicate using XML messages over open Internet transport protocols such as HTTP, HTTPS, SMTP, and FTP

Web Services existed before the principles of SOA were defined but they came into the limelight after it was realized that they were ideal for implementing services and hence form the building blocks of SOA.

SOAP and the Need for Attachments

SOAP is a specification that details how to encode the information to be exchanged in a Web Service. It's become a standard messaging protocol for Web Services. Since it's an XML-based, it's lightweight and stateless. SOAP can be used over various protocols like HTTP, HTTPS, SMTP, and FTP, but it's mostly used over HTTP. In the early days of Web Services and SOAP, because of the XML nature of SOAP, it wasn't possible to exchange non-XML data in a standard and efficient way. As the use of SOAP messages increased in organizations for exchanging information, the need to send non-text information (binary data) has also become important.

Use Cases

Let's take a look at a couple of use cases in which attachments are useful. In the banking domain, banks and financial institutions exchange paper-based financial instruments for clearing. In a few countries, to eliminate the overhead involved in exchanging physical checks, the banks have been mandated to move to an image-based solution, where the financial instruments must be digitized. In this new solution, banks have to transfer images of the checks along with the supplementary data. The supplementary data being small can go as XML content and the images of checks can be sent as attachments to the SOAP messages.

It's also a common requirement for sending binary data in the insurance domain. A Web Service application processing insurance claims requires images of the vehicle that met with an accident along with the claim document. In this case too images of the vehicle can be sent as attachments to the SOAP message, while the claim document itself can be sent as XML content in the SOAP message.

Since XML is a text-based format, it's difficult to include binary data in it. One solution is to embed the binary data as text using base 64 encoding. But the drawback to this method is that the performance degrades as the size of the attachment increases. The base 64 adds an extra 30% to the size of the original binary format because of the 4(characters):3(binary bytes) ratio, incurring a greater bandwidth over the network. And there's also overhead in encoding and decoding. However, it's certainly a good option for smaller attachments and has excellent interoperability but it's not an option for larger attachments. So a need arose for a better way to handle binary data.

Technologies Used for Attachments in SOAP

MIME (Multi-purpose/Multimedia Internet Mail Extensions) One solution that emerged as a standard to send attachments in SOAP is MIME, in which binary information is sent as an attachment outside the SOAP message. MIME was originally devised for sending attachments in e-mail. MIME is streaming-friendly and so can handle bulky payloads smoothly.

Limitation: The limitation in MIME is due to the use of text strings in a MIME package to delineate boundaries between parts. To find a boundary, the whole message has to be scanned for that string. If a part, by coincidence, contains that delimiter string, boundaries of the parts can't be detected correctly.

DIME (Direct Internet Message Encapsulation)

To overcome the limitation in MIME, DIME was proposed as an alternative standard, which uses offset values as a delimiter instead of a string. Though less flexible than MIME, DIME describes a simpler message format, which in turn provides for faster and more efficient message processing.

XOP (XML-binary Optimized Packaging)

XOP defines how to serialize SOAP messages with binary contents, preserving the XML Infoset. It's a generalized concept and also used for SOAP.

Standards for Attachments in SOAP

SwA (SOAP Messages with Attachments)

SwA was the first standard that tried to address the problem of sending attachments with SOAP. SwA is a W3C standard for attaching binary data along with SOAP envelope using SOAP 1.1 and MIME. SwA uses the HREF attribute and Content-ID MIME header to relate attachments to SOAP message parts. B2B standards such as ebXML, RosettaNet, AS2, and the BizTalk Framework use SwA as the messaging layer for its flexibility.

Limitations: SwA doesn't well define how to clearly describe an attachment in WSDL. An attachment by way of SwA meant two data models in one message and these two data models don't operate with existing XML technology, i.e., fail to provide a common logical view. And all the limitations of MIME hold true for SwA as well.

WS-Attachments

WS-Attachments use DIME for SOAP attachments. But, since most enterprises had already started using SwA and since the shortcomings in SwA, because of its use of MIME, weren't that severe, WS-Attachments with DIME didn't get adopted as a standard. It didn't gain much momentum due to the lukewarm response from industry practitioners. Refer to Listing 1 for a sample SOAP message for WS-Attachments.

Limitations: All the limitations of SwA also hold true for WS-Attachment except for the limitation of MIME, which was overcome by using DIME.

Attachment Profile 1.0

Though SwA was widely used, there were still two main concerns with SwA. First, there is no mechanism to identify from WSDL 1.1 that binary data is attached in the SOAP message. And second, there is no way to map MIME parts to corresponding WSDL message parts. SwARef introduced by Attachment Profile 1.0 solved these problems. Refer to Listing 2 for a sample SOAP message for SwARef.

PASwA (Proposed Infoset Addendum to SOAP Messages with Attachments)

PASwA emerged to address some of the shortcomings of SwA. PASwA aims at providing a common logical view to SOAP messages with attachments by using XML infoset. It defines a set of constructs to create a relationship between the SOAP message and its attachments. The principles of PASwA got grouped into four categories in W3C: (i) XML-binary Optimized Packaging (XOP); (ii) SOAP Message Transmission Optimization Mechanism (MTOM); (iii) SOAP Resource Representation Header; and (iv) Assigning Media Types to Binary Data in XML.

MTOM (SOAP Message Transmission Optimization Mechanism)

The MTOM specification is defined in three parts:

1. The first part describes an abstract feature for optimizing the transmission and/or wire format of SOAP by selectively encoding portions of the message, while still presenting an XML Infoset to the SOAP application.
2. The second part describes an optimized MIME multipart/related serialization of SOAP messages implementing the abstract SOAP transmission optimization feature in a binding independent way. This implementation relies on the XML-binary Optimized Packaging format.
3. The third part (HTTP SOAP transmission optimization feature) uses optimized MIME multipart/related serialization of SOAP messages for describing an implementation of the abstract transmission optimization feature for the SOAP 1.2 HTTP binding.

According to the specification:

- If the binary data is encoded as base64 then the original SOAP message is taken and the base64-encoded binary content is processed. Base64-encoded binary data is extracted and re-encoded (i.e., the data is decoded from base64) and put into the XOP package. This is overhead and MTOM-encoded messages might be larger than messages that use Base64 encoding for binary data.
- If the data is available as binary then the application can directly copy that data into an XOP package, at the same time preparing suitable linking elements for use in the root part; when parsing a XOP package, the binary data can be made available directly to applications.

Many products like JAXWS-RI use MTOM in an improvised manner. If the size of the attachment is less than a specific threshold (ideally 1KB) then the binary data is transmitted as base64-encoded due to the overhead mentioned above. But if the attachment is above the threshold size then it's transmitted as defined in the MTOM specification. Refer to Lising 3 for a sample SOAP message for MTOM.

Fast Infoset

Fast Infoset is a standard for specifying a representation of an instance of the XML Information Set using ASN.1 binary encodings. Fast Infoset specifies a binary format for XML documents and Fast Infoset documents are faster to serialize and parse and smaller in size than the equivalent XML documents.

Fast Infoset can also be used to handle attachments efficiently. As said above, as Fast Infoset describes binary encoding of the XML Information Set it allows for the direct embedding of binary data. In a way, Fast Infoset does what MTOM/XOP can do and a lot more. The performance of a Web Service with Fast Infoset is better than MTOM. Though Fast Infoset is being support by various platforms and frameworks, it's not popular yet; it's still emerging. This is definitely a promising technology and should be explored more.

Limitations: Not human-readable, not self-describing and tightly coupled since both the sender and receiver need to have the Fast Infoset processor. Once encoded as Fast Infoset, the XML can't be validated against the schema.

Securing Attachments

With the ever-increasing adoption of SOAP as the messaging protocol in organizations and the need to send sensitive data as attachments to the SOAP message has called for matured mechanisms for securing attachment payloads. The traditional WS-* specifications like plain WS-Security can work only for securing the XML content in the SOAP message but not the binary attachments that traverse outside the SOAP message. Hence, the binary data sent as base64-

encoded can be secured using WS-Security, but as already seen, this comes with a heavy price tag in the form of overhead if the data is huge. One way of securing attachments can be the use of transport-level security like HTTPS. But HTTPS can't provide features like encrypting part of the SOAP message. Also, HTTPS is unsuitable for a multiple-hops scenario.

There's been some amount of work done in this area. OASIS came out with a specification, Web Services Security SOAP Messages with Attachments (SwA) Profile 1.1, an extension to WS-Security that describes how SOAP attachments can be secured for attachment integrity, confidentiality, and origin authentication, and how a receiver can process such a message.

Conclusion

With an ever-growing number of enterprises embracing Web Service technologies and with demand for sending binary data on the rise, many techniques for handling attachments came to the fore. This paper tried to identify various such techniques in use and evaluate them by detailing the pros and cons.

References:

- SOAP Attachment State of the Art by Miko Matsumura http://www.infoq.com/news/SOAP-Attachment-State-of-the-Art

- Transferring Data securely using the MTOM Standard Part 1 by Digg, Reddit, Del.icio.us and DZone http://microsoft.apress.com/asptodayarchive/73639/transferring-data-securely-using-the-mtom-standard-part-1

- Evolution of Web Services Attachments Technologies by Canyang Kevin Liu and Sanjay Patil https://www.sdn.sap.com/irj/servlet/prt/portal/prtroot/docs/library/uuid/3b28e590-0201-0010-968f-e10ed5328191

- Boost Web Service Performance in JAX-WS with Fast Infoset by Young Yang http://java.sun.com/developer/technicalArticles/xml/fastinfoset/

- Web Services Security SOAP Messages with Attachments (SwA) Profile 1.1 http://www.oasis-open.org/committees/download.php/16672/wss-v1.1-spec-os-SwAProfile.pdf

- W3C MTOM specification http://www.w3.org/TR/soap12-mtom/ ■

About the Authors

Deepti Parachuri is a junior research associate at the SOA/Web Services Center of Excellence at SETLabs, the research arm of Infosys Technologies, Ltd., and holds a MS degree from IIT-Madras, India. She has an expertise in the field of vision, especially recognition and video tracking, and has been actively involved in various publications and conferences. Today her major research areas includes Semantic Web scaling RDF, OWL, OWL-S, and agent technologies, and, in addition, analyzed XML schema compression methodologies for SOA-based applications.
deepti_parachuri@infosys.com

Ujval Mysore is a member of the SOA/Web Services Center of Excellence at SETLabs, the research arm of Infosys Technologies, Ltd., a global IT consulting firm, and has substantial experience in publishing papers, conference presentations, and defining standards for SOA and Web Services. His fields of interest include SOA-enabling technologies like ESB and Web Services.
ujval_mysore@infosys.com

Listing 1 Sample SOAP message for WS-Attachments/Dime

```
POST /CheckService HTTP/1.0
Content-Type: application/dime
SOAPAction: “ http://example.com/soapaction “
Content-Length: 143616

[][][][][]\uuid:714C6C40-4531-442E-A498-3AC614200295[][][][]http://
schemas.xmlsoap.org/soap/envelope/[][][][][]<?xml version=”1.0”
encoding=”UTF-8”?>
   <soapenv:Envelope xmlns:soapenv=”http://www.w3.org/2003/05/
soap-envelope” xmlns:xsd=”http://www.w3.org/2001/XMLSchema” xmlns:
xsi=”http://www.w3.org/2001/XMLSchema-instance”>
      <soapenv:Body>
         <Detail xmlns=”http://example.org/mtom/data”>
            <CheckDetail>myBank</CheckDetail>
         </Detail>
      </soapenv:Body>
   </soapenv:Envelope>[][][][][]cid:1193909448545@myDomain.
com[][][][][]image/jpeg[][][]abcdefghijklmonpqrstuvwxyz[][][][]2006
:12:08 23:02:09
[][][][][]cid:1193909123545@myDomain.com[][][][][]image/jpeg[][][]a
bcdefghijklmonpqrstuvwxyz [][][][]2006:12:08 23:02:09
[][][][][]cid:1193909111545@myDomain.com[][][][][]image/jpeg[][][]a
bcdefghijklmonpqrstuvwxyz
```

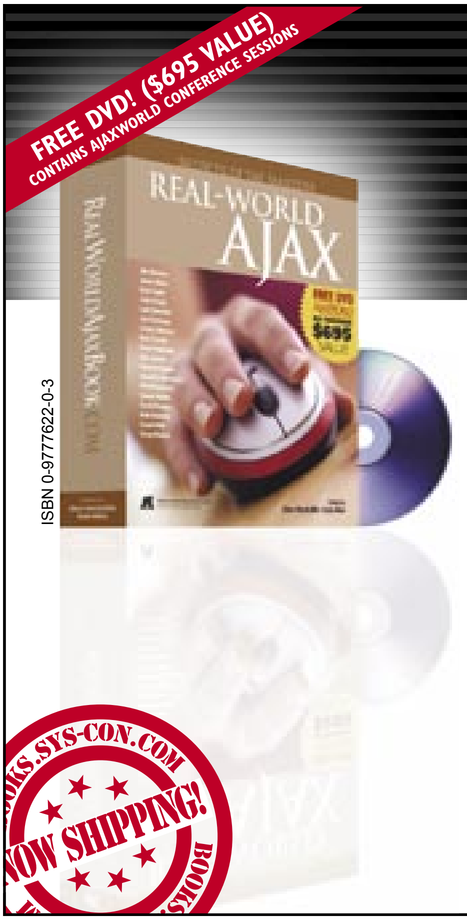
Listing 2 Sample SOAP message for by Attachment Profile 1.0

```
POST /services/CheckService HTTP/1.0
Content-Type: multipart/related; type=”text/xml”; start=”<8A10B726C
F88C59402909EA21095BFB3>”;    boundary=--MIME_boundary
SOAPAction: “http://example.com/soapaction”
Content-Length: 142549

--MIME_boundary
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: binary
Content-Id: <8A10B726CF88C59402909EA21095BFB3>
   <?xml version=”1.0” encoding=”UTF-8”?>
      <soapenv:Envelope xmlns:soapenv=”http://schemas.xmlsoap.org/
soap/envelope/” xmlns:xsd=”http://www.w3.org/2001/XMLSchema” xmlns:
xsi=”http://www.w3.org/2001/XMLSchema-instance”>
         <soapenv:Body>
            <check xmlns=”http://example.com/mytypes”>
               <checkDetail>myBank</checkDetail>
               <frontImage>cid:frontImage=1193908029165@myDomain.
com</frontImage>
               <backImage>cid:backImage=1193908029165@myDomain.
com</backImage>
               <greyImage sref=”cid:greyImage=1193908029165@myDoma
in.com”/>
            </check>
         </soapenv:Body>
      </soapenv:Envelope>

--MIME_boundary
Content-Type: image/jpgContent-Transfer-Encoding: binary
Content-Id: <cid:frontImage=1193908029165@myDomain.com>
....binary image....
--MIME_boundary
Content-Type: image/jpgContent-Transfer-Encoding: binary
Content-Id: <cid:backImage=1193908029165@myDomain.com>
....binary image....
```

```
--MIME_boundary
Content-Type: image/jpgContent-Transfer-Encoding: binary
Content-Id: <cid:greyImage=1193908029165@myDomain.com>
....binary image....
--MIME_boundary
```

Listing 3 Sample SOAP message for MTOM

```
POST /CheckService HTTP/1.0
Content-Type: multipart/related;type="application/xop+xml"; start="<4
DDB770EA58256BC4831FC317AB2A263>"; start-info="text/xml; charset=utf-
8";       boundary=--MIME_boundary
SOAPAction: " http://example.com/soapaction "
Content-Length: 143016

--MIME_boundary
Content-Type: application/xop+xml; charset=utf-8; type="text/xml;
charset=utf-8"
Content-Transfer-Encoding: binary
Content-Id: <4DDB770EA58256BC4831FC317AB2A263>
   <?xml version="1.0" encoding="UTF-8"?>
      <soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/
soap-envelope" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:
xsi="http://www.w3.org/2001/XMLSchema-instance">
         <soapenv:Body>
            <Detail xmlns="http://example.org/mtom/data">
               <CheckDetail>myBank</CheckDetail>
               <frontImage ns0:contentType="image/jpg" xmlns:
ns0="http://www.w3.org/2005/05/xmlmime">
                  <Include href="cid:frontImage=1193908940782@myDomain
.com" xmlns="http://www.w3.org/2004/08/xop/include"/>
               </frontImage>
               <backImage ns0:contentType="image/jpg" xmlns:
ns0="http://www.w3.org/2005/05/xmlmime">
                  <Include href="cid:backImage=1193908940782@myDomain.
com" xmlns="http://www.w3.org/2004/08/xop/include"/>
               </backImage>
               <greyImage ns0:contentType="image/jpg" xmlns:
ns0="http://www.w3.org/2005/05/xmlmime">
                  <Include href="cid:greyImage=1193908940782@myDomain.
com" xmlns="http://www.w3.org/2004/08/xop/include"/>
               </greyImage>
            </Detail>
         </soapenv:Body>
      </soapenv:Envelope>
--MIME_boundary
Content-Type: image/jpgContent-Transfer-Encoding: binary
Content-Id: <cid:frontImage=1193908940782@myDomain.com>
.....binary.....
--MIME_boundary
Content-Type: image/jpgContent-Transfer-Encoding: binary
Content-Id: <cid:backImage=1193908940782@myDomain.com>
.....binary....
--MIME_boundary
Content-Type: image/jpgContent-Transfer-Encoding: binary
Content-Id: <cid:greyImage=1193908940782@myDomain.com>
.....binary.....
--MIME_boundary
```



Why 'Enterprise Architects' Are Ineffective with SOA

WRITTEN BY DAVID S. LINTHICUM

Architectures are like archaeology; in essence, layers upon layers of systems, applications, databases, and connections, typically built or procured to solve a tactical problem.

Many corporations talk a good game and brag about the strategic long-term direction of the enterprise architecture that serves the business. The fact is, tactical needs have trumped strategic direction over the years. Thus, layers upon layers of technology on top of technology are the end result, and an architecture that is inflexible, static, fragile, and thus difficult to change along with the business requirements. This is the norm, not the exception.

In reaction to this dilemma, enterprises created positions called enterprise architects. These are single individuals, or groups, within the organization who have the responsibility to drive the enterprise architecture strategy going forward. While a good idea in theory, the reality is that many of these enterprise architects simply don't have the political or budgetary authority within their companies or government agencies to make much of a difference. In many instances, they have been relegated to those who create reports and presentations that nobody reads, and provide direction and guidance that's easily ignored.

Thus, without good architectural governance and ongoing corporate management pressure to redirect resources to tactical IT projects, the enterprise architectures continue to become more unnecessarily complex, static, and fragile. What was a mere annoyance only a few years ago, is today a clearly limiting factor in the businesses' ability to create shareholder value. The company can't easily shift into new and emerging markets, acquire companies, and adjust major business processes without a great deal of latency. In some cases, they are completely unable to change. In other words, things are bad and getting worse.

SOA is not a miracle cure for bad architectures. However, it is a step in the right direction for those looking to move their existing enterprise architectures into something much more efficient and valuable to the business. Those who embrace SOA as a practical architectural pattern in the context of a long-term strategic architectural plan, and are able to execute architectural rejuvenation without tactical interruption, will find that they are quickly ahead of the game.

SOA provides two primary values. First is the ability to save development dollars through reuse of services. Second is the ability to change the IT infrastructure faster to adapt to the changing needs of the business. Agility is the primary value proposition of SOA, and enterprise architecture for that matter.

Those looking to leverage the notion of SOA are tempted to sign up for the SOA-in-a-box type of solutions...perhaps an ESB, a BPEL engine, or a governance tool, or all of the above. Unfortunately, "buying-and-bolting-on" technology solves very little and could actually make things worse. SOA is something you do, not something you buy. But the buying patterns of those in the planning stages of SOA are still very much influenced by "hype-driven" and "manage-by-magazine" solutions that could cause many to find SOA distasteful as they realize the technology does not live up to the hype. There are no quick fixes, and real work must be done.

Indeed, doing SOA is a complex undertaking, and you'll need to learn a great deal to become efficient with the emerging approaches, techniques, technologies, and methods. Those who are successful at SOA, plan and design long before they develop and implement. The path to a truly strategic and valuable SOA is something that only comes to those who understand the importance of the work leading up to the technology. In addition, they have corporate sponsorship, the appropriate funding, and the proper amount of mentoring and training.

The movement toward a SOA should be something that has key strategic significance within the company or agency, much like a new product or line of business. In fact, a well-planned and implemented SOA will be far more valuable in comparison, considering its long-term ROI. In essence, SOA should have boardroom visibility.

Sometimes it seems that architecture must have a bad PR agency. The value is clear to anyone who analyzes the real cost of the limits that bad enterprise architecture places upon the business. However, the negative effects on the business are still widely accepted and thought of as something that really can't be fixed. Nothing could be further from the truth. ■

About the Authors

David S. Linthicum (Dave) is an internationally known application integration and SOA 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 SOA, approaches and technologies in wide use today. He keynotes at most major SOA and Enterprise Architecture conferences, maintains one of the most read SOA blogs, is the host of the weekly SOA Report Podcast, and is the author of 10 books, three on integration and SOA topics.

linthicum@att.net

Rich Web Technologies for Enterprise Web 2.0 & Social Applications

6TH INTERNATIONAL

AJAXWORLD™ CONFERENCE & EXPO

March 18-19-20, 2008
New York City
www.ajaxworld.com

Join More than 10,000 Early Adopters Who've Already Attended!

Come & Learn How to Achieve the Optimum Balance Between Business Needs, Technology Potential, & Enhanced User Experience!

Let AJAXWorld Provide Your Business Blueprint for Future Web Success. Doing business on the new Web is 99% about innovation and optimizing the user experience, and AJAXWorld Conference & Expo is 100% about the tools that encourage and support a superior user experience - the tools for business survival and Web success in 2008 and beyond.

Reasons to be there...

- You will leave AJAXWorld Conference & Expo 2008 East with resources, ideas and examples that you can apply immediately.
- Whether you're going to be using the technology to enhance an existing Web application or to migrate a client/server application to the Web, AJAXWorld will help you get up to speed more quickly.
- Attendance will improve your company's ability to adapt and prosper in the Web 2.0, user-centric environment.
- Whether you need to work with a designer or do the work yourself, front-end engineering makes more sense after you return from AJAXWorld!

The Real-World Questions That Will Be Answered by This Yearts Faculty!

- AJAX development IDEs: what are the alternatives?
- When it comes to accessibility, how do I know what's required?
- How can I make AJAX applications that easily go offline?
- How do you make a secure AJAX application?
- Are mashups really contributing to the experience of the Web?
- What's the difference between a mashup and a composite application?
- Am I better off using an AJAX framework, a toolkit or just coding?
- Is AJAX commoditized or will it be soon?
- Will AJAX be standardized in the form of widget APIs or declarative markup?
- What AJAX development tools are available for visual development today?
- How should you handle Web analytics and metrics for Rich Web sites?
- Do we really need JavaScript 2.0?
- How do you scale an AJAX-based design to a very large application?
- How can I harness the energy of my customer base online?
- On what timeline will AJAX skills become commoditized?

Technical sessions will cover challengers to AJAX such as new innovations like Silverlight and JavaFX. We will be dealing with performance and browser compatability issues, showcasing viable strategies for performing unit/stress testing on an AJAX application, and how to apply user interface patterns and user experience designs to your AJAX project.

Every aspect of AJAX and rich-Web alternatives will be covered, and the emphasis throughout will be on actionable insights, real-world case studies, and up-to-date technical information and advice.

Who Should Attend...

- CTOs & VPs of Engineering
- Directors of Technology
- Sr. User Interface Architects
- Front-End Engineers
- VCs & Industry Analysts
- Directors of Business Development
- Software Engineers
- Senior Architects
- Application Programmers & Software Developers
- Project Managers
- Web Programmers & Designers
- Companies & Organizations that need to stay in front of the latest Web technology

2008 Call For Papers Now Open!

AJAXWorld offers the undisputed best platform to position your company as a leading vendor in the fast-emerging marketplace for AJAX and Enterprise Web 2.0.

REGISTER TODAY & SAVE

Must-See Speakers

BOLOKER
IBM

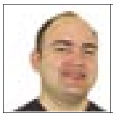
WEI
Nexaweb

FERRAIOLO
OpenAjax

CROCKFORD
Yahoo!

PELED
Gizmox

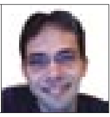
PELLER
IBM

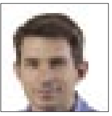

SCHALK
Google

GUPTA
Sun

MIRAGLIA
Yahoo!

STAGNER
Microsoft


POLLACK
Ruby Users

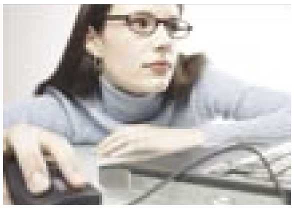

REGAN
Sonoa

AJAXWorld 2008 East - Tracks

- **Track 01:** Enterprise AJAX
- **Track 02:** Rich-Web Case Studies
- **Track 03:** Enterprise Web 2.0 & Social Applications
- **Track 04:** Enterprise Web 2.0 Mashups
- **Track 06:** iPhone Developer Summit
- **Track 06:** Diamond Track

Introducing in 2008

AJAXWORLD™ SECURITY BOOTCAMP



ajaxsecuritybootcamp.sys-con.com

March 18, 2008 New York
October 20, 2008 CA

iPhone Developer Summit

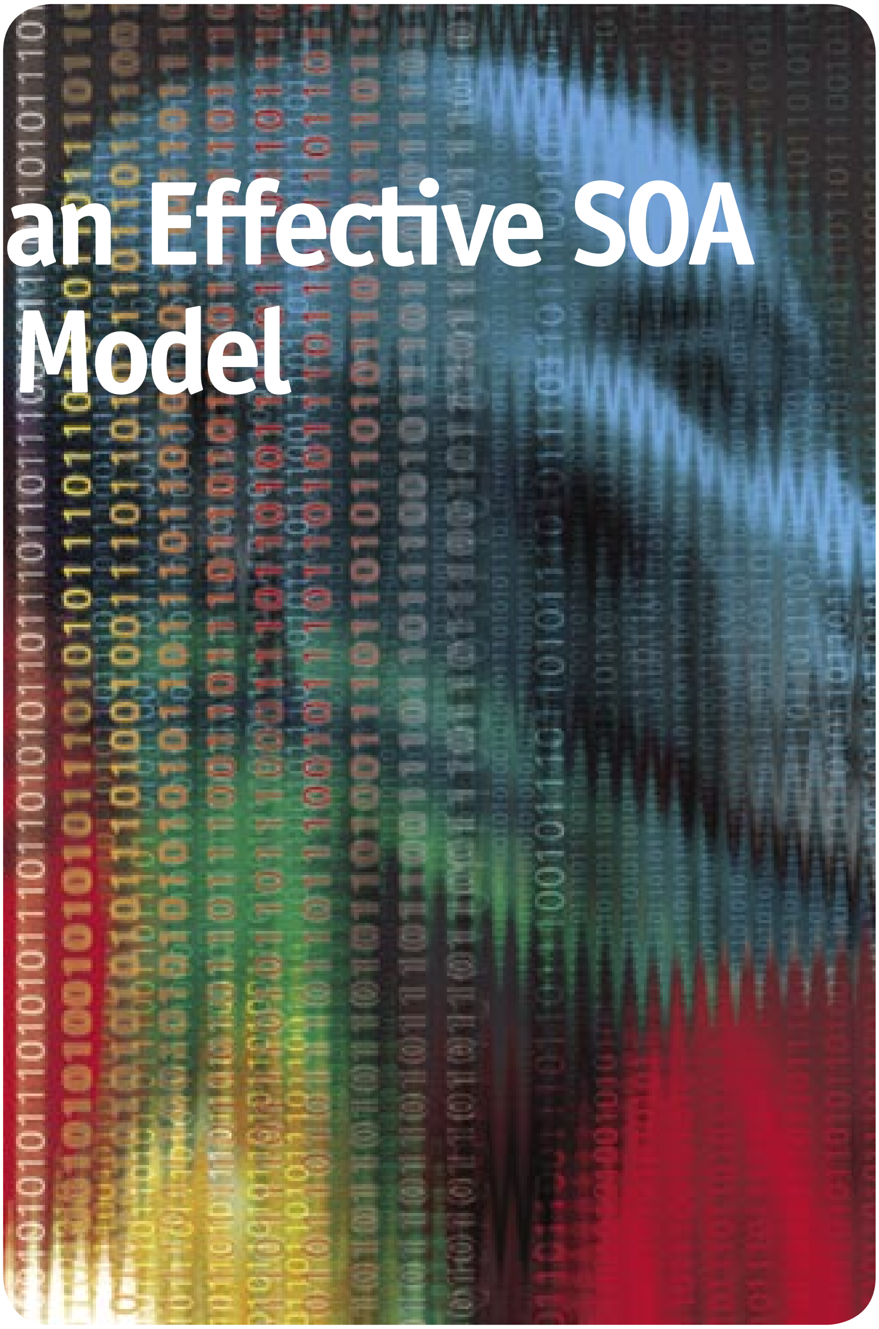
iphonedevsummit.com

March 19-20, 2008 New York
October 20-22, 2008 CA

Call Now for Sponsor & Exhibitor Opportunities
Space Is Limited 201-802-3021 or carmen@sys-con.com

CALL 201-802-3020 OR E-MAIL EVENTS@SYS-CON.COM

Setting Up an Effective SOA Governance Model



Laying the foundation for SOA success

WRITTEN BY **PRAVIN GOKHE**

While organizational business units and departments are in a hurry to embrace Service Oriented Architecture (SOA), without a proper control or governance of SOA adoption, the organizations are at risk of building unmanaged SOA environments that result in chaos and failure. The success of SOA largely depends on a robust governance mechanism. As a first step, organizations must start by defining a Governance Model addressing various challenges surrounding SOA space. In this article, we will discuss the key objectives of SOA governance and the various artifacts required for establishing a sound SOA Governance Model. A well-defined Governance Model will act as a foundation for long-lasting SOA success.

SOA Governance Objectives

Organizations are SOA-enabling their applications using technologies like the orchestration engine (BPEL), Enterprise Service Bus, and Web Services. While these technologies and standards help realize the objectives, they don't guarantee SOA success. Success and ROI are not primarily dependant on enabling technology platforms or vendors. SOA success is challenged by an altogether different set of issues that are related to process discipline, collaboration, organization culture, and decision-making.

Some of the key challenges are:

- The correct identification of a business service with appropriate granularity and reuse potential
- Achieving the objective of semantic interoperability by standardizing on canonical data models for service interface
- Ensuring uniformity in SOA implementations across project groups by enforcing best practices, standards, and guidelines
- Overcoming the mistrust and hesitation of service providers and consumers
- Building a common vocabulary for sharing service information across different business units or departments
- Breaking the barriers of the traditional organizational culture and behavior to ensure ongoing participation by service providers and consumers
- Inducing discipline in decision making

The only way to adopt SOA successfully is to address these challenges first. These challenges impose the need for SOA governance.

Some of the key objectives (see Figure 1) of SOA governance should be to:

- Introduce discipline while carrying out various processes related to the service lifecycle (Service identification-Service operation) by applying appropriate policies
- Create a culture of openness and cooperation by encouraging participants through rewards, incentives, and recognition programs
- Ensure effective collaboration by instilling trust and confidence
- Control decision-making by setting up organizational structures with appropriate responsibilities and accountability

SOA Governance

Until now, governance has been perceived as a luxury during the pilot phases. It is often underestimated as a mechanism to publish and discover business services by means of the registry tool. As SOA adoption moves from the pilot stage to the payoff mode, it's imperative to rethink SOA governance. Governance is required from day one, the cost of establishing governance late in SOA journey is much higher than if you adopt it early on.

The core of any governance is to ensure a desirable outcome in its discipline; the same applies to SOA as well. Business services are the primary artifacts in SOA and need governance to ensure high-quality enterprise-wide reusable/shared assets (business services) are being developed to meet the goal of business agility.

Effective SOA governance is what will separate leaders from dawdlers. Governance is a MUST in the overall SOA strategy and will form the base for adopting SOA in an organized manner. The very first step in establishing effective governance is to define a Governance Model. This process involves identifying and defining various artifacts related to goals, principles, policies, processes, models, metrics, and role responsibilities. Implementing a Governance Model by choosing the right technologies and tools would be the next step.

Governance Model Overview

The key areas of a Governance Model are depicted in Figure 2

Goals and Principles

Setting up the goals is the first step; shorter time-to-market, cost saving, and process flexibility are among the key objectives of a SOA. Aligning with business needs, goals need to be defined in concise, clear, and measurable terms. Then define a strategy such as focusing on specific domains or functional areas with high service reuse potential and ROI. For example, the customer service domain – targeting consolidation of customer-related business services spread across various LOBs. Well-defined goals and accompanying strategies will help to articulate a SOA vision clearly.

Defining the fundamental set of SOA principles is necessary to establish the common consensus as to how SOA should be used while developing enterprise SOA solutions.

Some of the basic SOA principles are:

- A business service should have a coarse-grained interface
- A business service should be exposed using a technology-agnostic interface and protocol
- A business service should adhere to enterprise-wide technology and semantic interoperability standards
- A business service should be autonomous. Any changes to its implementation technology, runtime environment, or location should not impact the service consumer
- The Interaction between a service consumer and a service provider must follow a document-oriented style of communication
- A business service must be discoverable by consumers belonging to other business units or LOBs
- A SOA infrastructure must provision for protection of the sensitive information exchanged between consumer and provider

Policies

Policies are central to governance. Start with identifying the core set of processes to be governed. Though the policies can be applied at various levels such as business, architecture, service, and technology, we will limit our discussion to business services

Some of the key challenges in the SOA space are:

- **Reuse –** How to ensure that business services are being developed for optimal reuse and interoperability?
- **Adoption –** How to ensure that business services have proper visibility, and there is a smooth and productive collaboration among LOBs
- **Sustenance –** How to ensure SOA sustains in the long run

In view of thes challenges, core processes have to be identified

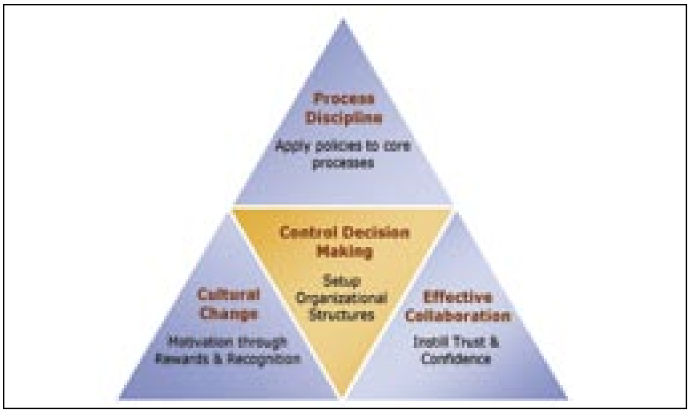

Process
Discipline
Apply policies to core processes
Control Decision Making
Setup Organizational Structures
Cultural Change
Motivation through Rewards & Recognition
Effective Collaboration
Instill Trust & Confidence

Figure 1: SOA governance objectives

SOA
Governance

Figure 2: SOA governance model

in the areas of service lifecycle, collaboration, and sustenance. Governing these processes would require identifying the control points within them and applying appropriate policies to control the outcome. The challenge here is to define policies prudently; they should not be too strict or too lenient. They should be reasonable enough to make their adoption smooth and acceptable.
Figure 3 shows the sample abstract of processes and control points within processes to which policies can be applied.

Service Lifecycle

A service lifecycle comprises activities right from service identification through service operation. Various design-time and runtime policies need to be defined around these lifecycle processes.

The primary objective of design-time policies would be to ensure that truly reusable and interoperable business services are being developed.

Reusability – Identifying which business functionalities are the right candidates for service enablement is a difficult task. Without a proper approach to service identification, project groups will end up identifying services that don't have enough reuse potential. The policy related to the service identification process should enforce the use of a business process-centric, top-down/bottom-up analysis approach to identify genuine reusable functionalities.

Interoperability – Interoperability policies should focus on building technically and semantically interoperable business services. For example, as a part of the Service Interface Spec/Design process, a high-level policy for semantic interoperability can be defined to enforce the adherence of the business service interface to the enterprise-wide canonical format. This can be followed by more specific policies aligned to domains or functional areas. Similarly on the technology interoperability front, policy can be defined to enforce compliance with the WS-I basic profile to achieve Web Service interoperability across platforms, operating systems, and programming languages.

Runtime policies should be defined and enforced to govern the behavior of a service once it's operational. For example, a runtime policy for SLA monitoring and non-compliance reporting should be defined and enforced with the intention of getting visibility into the compliance issues and subsequently taking remedial action either in an automated or manual way (automatically instantiating an additional instance of service in case of performance degradation is an example of automated remedial action).

Examples of runtime governance include:

- Managing security aspects such as access control and data-level security through encryption
- Managing service level agreements (response times, availability, etc.) through compliance monitoring and reporting
- Managing auditing, logging, and exceptions

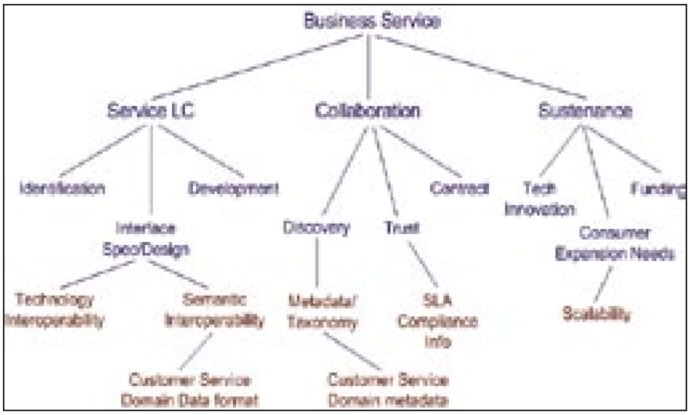


Figure 3: Sample abstract of processes and control points

It's important to govern the runtime concerns or issues related to service behavior as it will help preserve the trust between service providers and service consumers and create a sustainable SOA environment.

Collaboration

Policies around collaboration should be targeted at accelerating the adoption of business services. This would involve defining policies around key areas such as discovery, trust, contract, and so on. The policy around service metadata would enforce use of clear business-technology-operational taxonomies for describing the service. Use of proper metadata and taxonomies will enhance service visibility across consumers. Similarly, defining the strong policies around trust and contract will help overcome the barriers of mistrust and hesitation present among service consumers. Agreeing to service levels and the integrity of service through a formal contract is necessary to gain consumer confidence. Providing historical compliance data on aspects such as SLA, support will elevate the trust further.

Sustenance

SOA is an ongoing phenomenon. Some of the key aspects around sustenance that need governance are:

- **Consumer expansion needs-scalability –** It's important that business services should scale to embrace new consumers on an ongoing basis and support the expansion needs of existing consumers in terms of increased volume

- **Technology innovation –** Migration of applications-business functionalities to a newer standards-based platform is inevitable for reasons such as reduced licensing and maintenance costs. While this is happening at the providers' end, existing consumers should be protected against the risk of changes in SLA in terms of performance, supported volumes and availability.

- **Funding –** There should be continuous funding to support and maintain business services and SOA infrastructure

Unless these sustenance-related challenges are thought through upfront, confidence in SOA's ability to sustain will diminish.

Models and Metrics

It's necessary to define a robust enforcement and compliance model. Some of the key considerations are:

- Deciding how various policies should get enforced – as part of the process or using tools technologies. For example, most development policies will be candidates for automated enforcement through tools
- Defining validation and review processes to ensure compliance
- Certificate of compliance for business services through various stages in the lifecycle to make the process more robust
- Exception management and impact analysis

Behavioral and cultural changes are necessary to get SOA right. The conservative, inward nature of an organization can lead to SOA failure. People have to stay away from the traditional mindset of "develop it myself." Being an enterprise-wide strategy, SOA's success depends primarily on the productive and effective participation of service providers and consumers. In short, to succeed SOA needs a culture of openness and healthy relationships. Creating such a culture requires organizations to promote positive behavior by providing rewards, recognition, privileged funding, and such. At the same time the negative tendency of reinvention-duplication should be discouraged.

Metrics provide the basis for measuring SOA success. Plan to measure success through indicators such as time-to-market for new product-process, cost-savings, and number of processes streamlined. Metrics can also be used to determine the effectiveness of governance. Define metrics to collect statistics on factors such as successful negotiation, reusability, and the interoperability compliance rate. This will help in understanding the loopholes and rigidity in current governance processes-policies and improving it further.

Roles & Responsibilities

Defining organizational structures is mandatory in bringing discipline to the decision-making process. These organizational structures will include various business, domain, and technology groups/boards/committees. The responsibility of identifying roles within these structures and assigning them unambiguous responsibilities and accountability lies with a central SOA governance committee.

Examples of the structures and roles include:

Business

Business Process Streamlining Committee – Will be responsible for working closely with different business units to understand process streamlining/automation opportunities in a particular domain. For example, the customer service domain (examples of business units would be retail banking and investment banking for individuals). This committee will deliver a Process Plan outlining the processes to be streamlined and automated.

Cost-Benefit Analysis Committee – Will be responsible for assessing the service enablement/sharing of business functionalities from the perspective of business benefits, cost involved, and ROI. This report will be input to the Process Owners Committee.

Process Owners Committee – This committee (domain-specific) will involve people from different business units. This committee will be responsible for approving/rejecting the plan for service enablement/sharing of business functionalities.

Domain

Customer Service Domain Architect Group – The Process Plan will be input to the Domain Architect group. This group will be responsible for delivering detailed processes blueprint identifying common reusable assets across processes. This processes blueprint will act as an input to the Cost-Benefit Analysis Committee and Process Owners Committee

Customer Service Domain Analyst Group – Will be responsible for identifying or defining enterprise canonical semantic standards to be used by the business services.

Customer Service Interface Committee – Will be accountable for approving the semantic standards and their use by business services.

Technology

SOA Center of Excellence – Will be responsible for carrying out following activities

- Support the project/product teams with design and development of business services, composite applications, etc.
- Mentor the project/product teams on SOA-enabling technologies and standards (BPEL, ESB, Web Services)
- Develop best practices and guidelines
- Proof of concept (PoC) execution
- Build a pool of resources capable of executing SOA projects and programs
- Provide thought leadership and technology direction

SOA Platform Review Board – Will be responsible for validating vendors' technology platforms for standards compliance and interoperability with other components in the SOA ecosystem.

Architecture-Design Standards Board – Will be responsible for identifying emerging trends and standards related to Web Services-SOA.

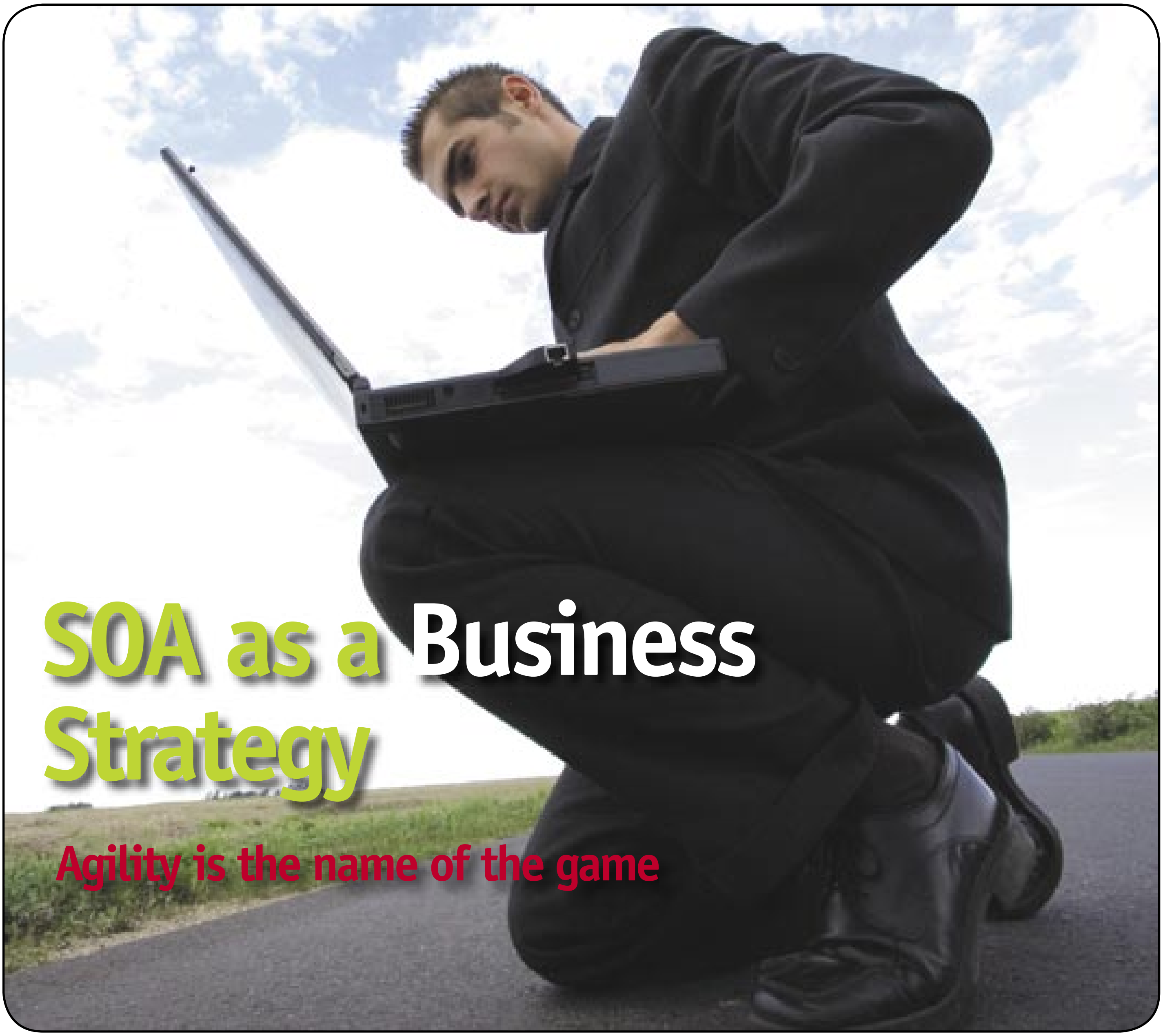
SOA Standards Review Board – Will be responsible for approving/rejecting architecture-design standards.

Once the Governance Model is in place, an organization needs to focus on its implementation. An organization won't find a single vendor or tool capable of implementing the governance solution; it might have to work on custom framework development and the integration of different governance tools to realize the Governance Model.

Conclusion

Organizations are in the initial stages of adopting SOA governance. Without effective governance, the ability of SOA to scale will impede. Therefore it's vital to establish a sound Governance Model addressing the various challenges surrounding SOA discipline. This will help organizations to adopt SOA successfully by establishing process discipline, controlling decision-making, ensuring smooth collaboration, and addressing cultural/behavioral issues. SOA governance is complex. As SOA evolves, an organization needs to evolve its Governance Model to address the new set of challenges. A well-defined Governance Model will lay the foundation for SOA success. ■

> "Governance is a MUST in the overall SOA strategy and will form the base for adopting SOA in an organized manner"



SOA as a Business Strategy

Agility is the name of the game

WRITTEN BY ANTHONY GOLD AND JAMES IRWIN

If you had to pick a single business benefit that service-oriented architecture (SOA) can provide, it is the ability to respond to change. Change occurs continually in a multitude of places that affect the enterprise: the market, the supply chain, strategic processes, regulations, and so forth. SOA can enable the creation of an agile environment that creates stability in the face of change because it restructures automated functions into reusable pieces that can be quickly reconfigured into new or modified processes.

But an architectural approach alone will not create agility. Agility comes from having the right pieces from which agile processes can be assembled. Developing, refactoring, or acquiring the right services requires a deep understanding of your business.

Over the past few decades, many models have evolved to push automation toward agility. Simple mainframe applications gave way to numerous integrated processes running in central mainframe – enabling libraries of functions. Client/server and later multi-layered systems began to allow for more targeted replacements or upgrades – so-called "separation of concerns" – the hallmark of agility.

Those advances in agile technology, however, were often offset in the operations center by the introduction of multiple disparate systems, each somewhat agile in its own way, but not designed to operate together in an "agile enterprise." Business drivers such as mergers and acquisitions often brought together systems whose designers had no vision of such cross-enterprise interoperability, or, worse, whose vendors had no motivation to create an integrated system responsive to changing requirements. While Enterprise Integration suites very successfully addressed many challenges, the basic approach of integrating individual and disparate interfaces is inherently expensive and brittle.

SOA ushers in nearly a perfect storm for agility, enabled by standardized best practices, pervasive communication and data protocols, and a general movement toward "openness" in the industry. Best practices have emerged from enterprise integration efforts and other advances such as object-oriented approaches. The Internet has made the plumbing necessary for integration ubiquitous and well understood. Openness has come both from the demands of customers as well as the "disruptive" open source and open information movements.

Why nearly a perfect storm? Because like all the previous technologies, SOA is only an enabler. Without a deep understanding of today's processes and a general sense of how they are likely to change, the inability to choose the right components prevents the storm of agility from forming completely.

SOA is about how – but a true business plan using SOA really needs to start with what. Simply put, without a clear understanding of the "what," the "how" becomes nearly irrelevant.

Blueprints

What exists to guide the application of these new architectural approaches and underlying technologies? What prevents SOA practitioners from simply rearranging the pieces without really achieving SOA's promised agility and alignment with business objectives? What ensures that the pieces are not rearranged into "an SOA" without producing significant benefits?

There has been a lot of discussion in the SOA literature about the importance of governance, but this too is more about the "how" than the "what." Design-time governance, for example, supports the reuse of components, which is a critical piece of SOA adoption (why pay to make something reusable if no one is able or compelled to reuse it?). But this says little about the process of determining what is being reused. Again, the prerequisite to success seems to be a model that guides the selection or creation of components that truly enable agility – components that can be called upon to support new or changing higher-level business processes.

There is also a lot of discussion about whether SOA should be adopted from the top down or the bottom up. Both approaches have clear advantages and challenges. While top-down can "feel right" because it's based on planning and thoughtful investigation, it can become overly prescriptive and mired in "analysis paralysis" – sometimes to the point where requirements and even technologies are changing out from under the plan.

The bottom-up approach enables a quick start, is often driven by enthusiasm for new technical possibilities, and has the benefits of an "organic" methodology. However, it can also lead to gold-plated components that are never reused or disorganized collections of services that might adhere to service-enabled principles on a small scale, but which collectively fail to provide the right mix to truly enable agility and realignment with business processes.

A "meet in the middle" strategy can provide the best of both worlds. Understanding top-down business objectives and decomposing them and the processes that support them is clearly the most controlled and planned approach. But "the best-laid plans…" wisdom supports the idea of bottoms-up, where getting started and refactoring along the way allows progress to benefit from the lessons learned as well as supporting finer-grained adjustments to account for changing priorities, requirements, and technologies.

The "meet in the middle" approach often has the time benefit of supporting parallel development efforts, because top and bottom often represent different organizations and disciplines. But therein lies the challenge. Two teams digging a tunnel to "meet in the middle" need to understand quite precisely how one side will connect to the other. Left to chance, there is very little possibility that this will happen.

A comprehensive and layered set of models representing current and desired states, sometimes referred to as a blueprint, supports "meet in the middle" quite effectively. That approach requires clear traceability from the top to the bottom, and the blueprint provides it. Development of such a blueprint will iteratively enable "the big picture" to be evolved to its maximum benefit while enabling progress in parallel on obvious infrastructure and process requirements.

It is common advice from SOA practitioners to establish a proof-of-concept that has significant visibility to keep attention and funding, but with simple enough requirements to ensure an initial success. Adoption of such an approach within the framework of an iteratively developed blueprint allows exploration with traceability from business objectives down to the infrastructure that helps realize them. Do enough top-level modeling and definition to establish a foundation and identify key processes for early development, but get started quickly with the development to gather experience to continue to the next steps, well informed and confident.

Because a blueprint provides traceability, it shows what requirements you were satisfying with the current implementation or proof of concept. It becomes a plan that captures changes in each step toward an enterprise based on SOA and containing the right components to maximize agility. A blueprint creates a common language and defined touch-points between the business architect working at the top and the systems architect working from the bottom. A blueprint supports a plan to ensure that they truly "meet in the middle."

In essence, a blueprint is a "paper copy" of some or all of the enterprise – in fact, because it is activated by software tools, it might more properly be called a digital model. In a blueprinting approach, governance becomes the controls and communication structure necessary to turn the "copy" into reality. Planning lays out the schedules and resources to address the gap analysis between "as is" models and "to be" models within the blueprint.

Because of the traceability between the "layers" of the blueprint, prioritized business objectives help identify the business processes that require change, which in turn point to the underlying automation and components involved, which then identify any needed infrastructure changes. It is easy to see that with a blueprint, projects can be prioritized based on scope, ROI, resource availability, or other factors not directly driven by the architecture and technologies. Execution of the plan can occur in parallel with the expectation that all the interdependent pieces are available on schedule. IT aligns with business goals.

Applied Blueprints

A blueprint and the traceability it includes have a direct impact on ensuring that the right reusable pieces are produced. Continual requirements gathering coupled with prioritization facilitates the right discussions around which components really will provide the maximum ROI. Like change, requirements come from all over the enterprise. An inventory of existing components and processes, sometimes referred to as the “as is” model, helps establish the current state of the enterprise. This enables tools such as maturity models to be applied as part of the gap analysis referred to earlier to couple goals with needed changes.

A blueprint will expose business process patterns that also can be reused, making the subsequent iterations of the blueprint and development easier with faster return and less risk. The blueprint business objectives guide your architecture while identifying business processes, and the patterns within them guide the design and priority of components. Existing assets identified in “as is” models can be examined against modeled processes to determine if they need to be refactored to support the “to be” states.

While SOA is an architectural approach, it typically resides in the technical domain. A major positive impact of the SOA movement has been the acknowledgement that what affects business affects IT and, conversely, what affects IT affects business. Many maturity models now include dimensions such as organization and funding. With an emphasis on reuse, there must be consideration of the organization and funding that support development, purchasing, and support of reusable components. This is a different model than siloed organizations using siloed applications.

While some organizations with centralized IT may be well positioned to follow this model, other organizations with more distributed structures face additional challenges. Often public sector organizations are distributed based on the services they provide, e.g., law enforcement and court administration. A blueprint has the capability to identify these potential adoption challenges and integrate the functions. For example, an integrated justice information system based on a blueprint can help police learn about outstanding warrants in different jurisdictions, and that can make it easier to prevent crimes by identifying offenders more readily.

By bridging various horizontal disciplines, a blueprint helps expose the gaps between vertical interests while creating an opening to reduce risks and capitalize on opportunities.

How Do You Measure Success?

It would be easy to look at the architectural diagrams of an IT infrastructure, count the number of Web services that were developed, look at how many are reused, and declare your SOA journey a success. Perhaps from a technical perspective it has been a success. But the real measure of success is the increasing ease of executing the next iteration from your blueprint. That is true agility.

It becomes easier because you are recognizing patterns in your processes. It is easier because you find that, from the previous iteration, you increasingly have the right component services in your arsenal. It is easier because you are positioned to understand how to purchase and integrate services you need. It is easier because you invested in externalizing the business logic for processes you anticipated changing and now those new requirements can be achieved without recoding. If it’s not getting faster and easier, you may have adhered to the technical design principles, but are likely missing the real return promised by the SOA enabler.

A blueprint provides a foundation to get started, objectives on which to structure an iterative plan, and a basis against which to measure success at the business level. It provides a foundation against which all the elements of SOA adoption can be applied: technologies, governance, and iterative improvement.

You probably could evolve your enterprise to SOA without a blueprint, but it would be like attempting to build a house without its architectural equivalent. You may end up with a kind of Winchester Mystery house – many years of construction, lots of nice but unrelated features, staircases that lead nowhere, and lots of features based on superstition due to bad or misunderstood premises and assumptions. ■

About the Authors

Anthony Gold is vice president and general manager, Open Source Business, Unisys Corporation.
Anthony.Gold@unisys.com

James Irwin is an open source software architect, Unisys Corporation.
James.Irwin@unisys.com

Advertiser Index

ADVERTISER	URL	PHONE	PG
Active Endpoints	www.active-endpoints.com/soa		15
AJAXWorld Con & Expo	www.ajaxworld.com	201-802-3021	27
Altova	www.altova.com		2
Attachmate	www.attachmate.com/soa		36
Fiorano	www.fiorano.com/downloads		7
Managed Methods	www.managedmethods.com		17
Mindreef	www.mindreef.com/not-stranded.html	603-465-2204 ext.581	5
SYS-CON Books	www.realworldAJAXBook.com	201-802-3020	25
Web Age	www.webagesolutions.com	877-517-6540	35