

**COMPUTERWORLD**

---

# Executive Bulletin

---

## Sold On SOA

It's the hot technology for most large companies, but business, technical and cultural issues must be addressed for a successful SOA implementation.

Sponsored by



<b>OVERVIEW</b>	
Executive Summary .....	<b>2</b>
<b>WHAT'S AN SOA?</b>	
QuickStudy: SOA .....	<b>4</b>
<b>THE BIG PICTURE</b>	
The Year of the SOA? .....	<b>6</b>
The SOA Network Effect .....	<b>8</b>
SOAs Eliminate Integration Headaches .....	<b>10</b>
<b>IMPLEMENTATION</b>	
Web Services: Managing the Building Blocks .....	<b>12</b>
Web Services Security: Trouble in Transit .....	<b>15</b>
Message Received? .....	<b>17</b>
<b>FIELD REPORT</b>	
Hammering Out Web Services Links .....	<b>19</b>

# Executive Summary

**S**ERVICE-ORIENTED ARCHITECTURE (SOA) is one of the hottest areas of IT today. SOA has grabbed CIO attention because of its potential to reduce IT costs, improve business and IT agility, increase software asset reuse, orchestrate business processes based on reusable “services,” and ultimately replace the proprietary IT integration model that proliferated in the 1990s with a lower-cost, standards-based SOA approach to enterprise integration.

However, SOA is an IT discipline that requires proper strategy and planning, a new governance model that involves the IT and business organization, a new architecture model and development process, and new technologies to be able to realize these SOA benefits. SOA is both organizationally and technologically complex, and organizations are just beginning to understand the challenges that are part of an SOA initiative. That said, the benefits are too compelling for IT executives to ignore, and thus SOA has become a key initiative in most organizations.

SOA can be defined as follows: a business computing architecture that enables and delivers business functionality to its users (consumers) in the form of shared, reusable services. The concept of SOA has been around since the rise of distributed computing and object technology. Early attempts at SOA achieved limited success due to the platform-specific nature of implementations such as CORBA, DCE and COM/DCOM.

The new incarnation of SOA is enlivened by the industrywide

agreement on a body of standards for Web services — messaging, interface descriptions, security, service registries and many more — that allow the deployment of shared, reusable services in a cross-plat-

form implementation of SOA. This is the big “Aha!” for SOA: cross-platform services running across the enterprise using industry standards. Many organizations now realize that Web services didn’t solve the real business problem. The bigger challenge is the organization, governance, architectural model and infrastructure needed to enable reusable and interoperable services in an SOA. That realization pushed SOA to the fore in mid-2004, and it has been a hot topic ever since.

SOA is fundamentally a more flexible IT architecture than previous paradigms. By offering IT functionality as cross-platform shared services in an SOA, a number of benefits arise. First, there is a clear ROI associated with the reuse of services in an SOA. Once a portfolio of Web services is available to be leveraged in an SOA, these reuse benefits multiply in an “SOA network effect,” where SOA value increases with the number of available services and consumers of those services. This benefit compounds over time as the SOA is leveraged internally and externally by developers, business analysts, and external consumers and trading partners.

Besides cheaper and faster application development and the other IT benefits mentioned above, SOA provides higher-quality applications via pretested components and services, and overall faster response to the business for system enhancements and modifications. SOA benefits the business side through greater flexibility, faster time to market for business initiatives,

**SOA Adoption**  
 Large organizations are more actively adopting SOA techniques and technologies, according to a recent Forrester Research Inc. survey of 116 North American decision-makers familiar with programming technologies, application software architecture and application platforms.

Percentage of respondents reporting SOA use:

- 70% Large enterprises (20,000+ employees)
- 28% Medium enterprises (5,000 to 19,999 employees)
- 35% Smaller enterprises (1,000 to 4,999 employees)
- 22% SMBs (fewer than 1,000 employees)

SOURCE: FORRESTER RESEARCH INC., 2005

Computerworld editor in chief **Don Tennant** ■ Online projects editor **Ian Lamont** ■ Executive Bulletin editor **David Ramel**  
 ■ Designer **Julie Quinn** ■ Design director **Stephanie Faucher** ■ Managing editor/production **Michele Lee DeFilippo**  
 ■ Copy editors **Bob Rawson, Eugene Demaitre, Mike Parent, Monica Sambataro**

## Overview

---

faster response to business changes, and better alignment of IT services to business needs.

Most SOA adopters will soon realize that it's much more than a technical architecture. SOA is not a packaged software application. It can't be purchased from a single vendor. In fact, it may not even require additional software or technology at all. SOA is a way of running a business and IT organization to provide new levels of business performance and customer service while driving internal IT efficiencies. SOAs comprise the following six elements:

- SOA vision, business goals, conceptual architecture and standards.
- A general view of services, which will include Web services.
- Current and new enabling SOA technology.
- SOA governance and policies.
- SOA metrics.
- Organizational dynamics, behaviors and culture.

Of this list, only one directly relates to technology. Most of the others are organizational or softer aspects of SOA that are critical to its success. SOA governance has become a major focus for adopters because it defines the organizational, process and behavioral issues that make SOAs successful. SOA gover-

nance ultimately influences SOA behavior.

An SOA strategy must address the organizational issues and behavioral challenges that will either facilitate or inhibit SOA adoption, such as services ownership, business and IT interactions, budgeting practices and many more too numerous to mention. Organizational and process issues thread through several areas of an SOA initiative. How do you organize your architecture organization, processes and disciplines to support an SOA? How do you organize your developers to support your SOA initiative? What is the optimal IT structure for an SOA? How does SOA governance impact the existing IT governance process?

SOA is not a quick fix for all of your business or IT challenges, but it does provide a strategic pathway forward for your organization. Remember, the IT challenges that SOAs address are artifacts of years of corporate decisions and behaviors that occurred over time. It will take time to undo the accumulated technological complexity in your IT architecture.

The good news is that SOA is an incremental architecture, so you can scale your road map to SOA project by project and yet still deliver compelling ROI that accrues as you enable more services within your SOA.

There are numerous challenges ahead of you, but they shouldn't slow your efforts.

The SOA standards are still in flux, especially the second-generation standards. Focus on the mature core standards and monitor the others as they evolve. SOA-enabling technologies are still somewhat immature, so careful evaluations must be performed. Technology vendors want to equate SOA with their software platforms, when in reality SOA is much more than that. The core assets of an SOA are the services, not the technology platforms.

Invest in services identification and design; these are the lasting enduring assets of your SOA. Be sure to always maintain the business context for your SOA initiative. Explicitly identify the business challenges or imperatives that your SOA initiative addresses.

And be sure to invest in the development of an SOA governance model and policies that can be enforced both during services design and runtime. Governance can make or break your SOA initiative. Defining and enforcing SOA governance is hard work, but it is essential for success.

---

*Written by AgilePath Corp. CEO Eric Marks, who also contributed several articles to this briefing.*

# QuickStudy: SOA

**G**IVEN THE BUZZ about service-oriented architecture today, let's make a few important points: The SOA concept isn't new, it's not a technology per se, it isn't just the use of XML and Web services, and it's a good deal more than a development methodology.

SOA is a pretty simple idea: Standardize those generic functions that are widely used by many applications into reusable components (services) that are accessible over a network, and code more specific logic needs into the application itself. Indeed, every operating system is a prime example of an SOA in action, if not in name.

IT organizations were successfully building and deploying SOA applications years before XML and Web services existed. They just talked about the process using terms like modularity, reusable components, object-oriented programming or application programming interfaces. Although none of those concepts is identical to SOA, they all embody aspects of it.

SOA is just the latest shorthand for a method of designing, developing, deploying and managing discrete pieces of computer logic (read "services") within a computing network. It's a way of structuring applications, organizing IT infrastructure and standardizing business functionality. Although SOA is often associated with the use of XML and Web services, these latter two are merely the latest implementation of the SOA principle.

SOA requires developers to design applications as collections of services, even if there's no immediately apparent benefit to doing so. SOA requires developers to think beyond their current application, to consider reusing existing services and to examine how other developers might reuse the services they

are creating. SOA encourages developers to use alternative technologies and approaches, such as messaging, and to build applications by linking services together rather than by writing new code. This type of application structure allows a company to react quickly to changing market conditions; instead of having to develop new application code, they can simply modify the messaging.

By focusing on business processes and using standard interfaces, SOA can help hide the underlying technical complexity of the IT environment. Analyzing the interaction between services in an SOA lets companies understand when and why specific business logic is being executed, which makes it easier to optimize business processes.

### Loose vs. Tight Coupling

One key feature of SOA is the use of loosely coupled connections. Traditionally, connections between applications or between applications and services have been tightly coupled, as with CORBA. The difference is important.

Eric Van der Vlist, author of *XML Schema: The W3C's Object-Oriented*

*Descriptions for XML* (O'Reilly, 2002), describes the differences between the two types of coupling with this analogy: In a tightly coupled system, you have direct control over the operation. For example, flipping a wall switch to turn on a light is a tightly coupled system. However, making a telephone call to order a book is a loosely coupled system. It could be tightly coupled only if you had access to the button controlling the printer that will print the book you order.

Tightly coupled systems are usually fast and safe, and the risk of transmission errors is very low. Loosely coupled systems, on the other hand, are usually more error-prone but also more flexible. The clerk you talk to on the phone may misunderstand the ISBN number of the book you want to order or make an error while entering it. But if you don't remember the ISBN number, you can still tell the clerk that you want the latest book on the World Wide Web Consortium's XML schema by a guy with a Dutch name from a publisher that puts pictures of animals on its book covers, and when you do that, you've got a good chance of being understood.

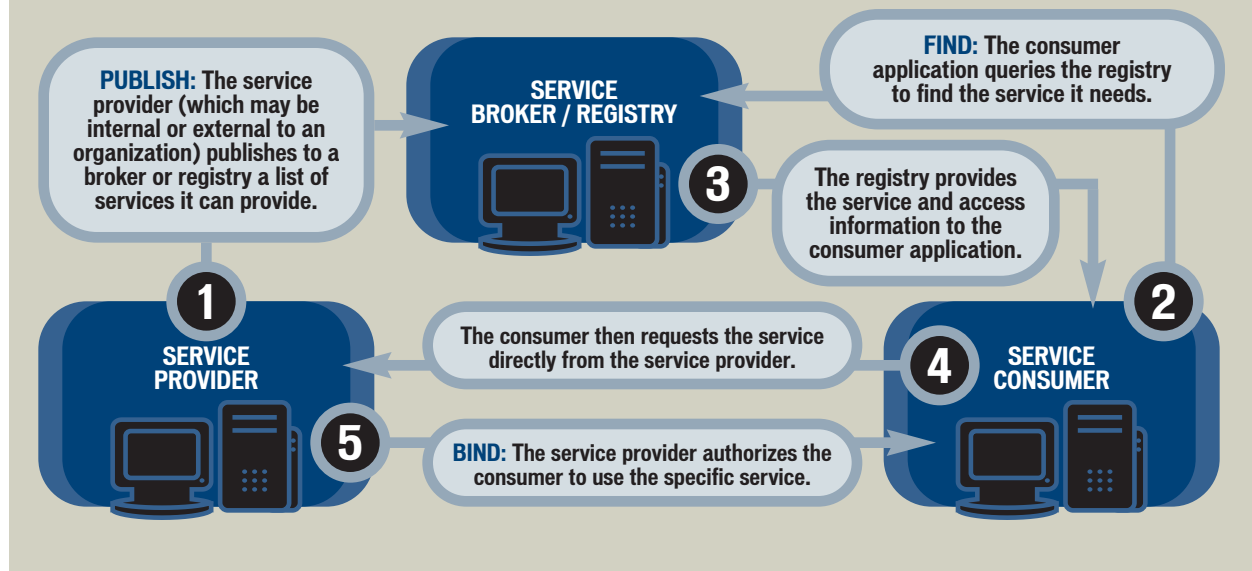
Tight coupling tends to make component maintenance and reuse much more difficult, because a change in one component automatically means changes in others. Similarly, tight coupling makes extra work when an application has to adapt to changing business requirements, because each modification to one application may force developers to make changes in other connected applications.

In general terms, a Web service is a type of SOA in which interfaces are based on standardized Internet protocols. In addition, except for binary data attachment, Web service

## What's an SOA?

### How SOA Works

A Web services SOA involves three primary entities that communicate with one another: a producer or provider of services, a consumer or requester of services, and a directory, registry or broker of available services.



messages must be in XML. Generally speaking, a Web service is little more than an SOA that uses Simple Object Access Protocol and the Web Services Description Language.

However, an SOA doesn't require the use of Web services as we understand them, and some types of Web services can be deployed without an SOA.

# The Year of the SOA?

**T**HIS WILL BE THE YEAR OF THE SOA. Mark it down. According to The Yankee Group, 75% of firms plan to invest in the technology and staffing to enable an SOA. This is a serious movement away from IT solutions of the past and toward standards-based architectures with promise for the future.

This SOA trend is symptomatic of the fatigue experienced by both business and IT end users. They are tired of vendor lock. They've had it with applications that won't interoperate with other enterprise applications. They are fed up with IT architectures that are rigid, expensive to maintain and operate, and inhibit business and IT agility. They are tired of spending ever-tightening IT budgets on internal integration issues that just won't go away. SOA holds the promise to address all of these issues, and as The Yankee Group's survey suggests, many end-user firms agree.

Here are some ways that these SOA adopters can avoid some pitfalls.

- SOA is a business-driven issue. Companies should focus on their business imperatives first. In other words, what business issues must be fixed or improved or else the company is endangered? The "or else" motivator is what creates the compelling event for moving to an SOA. There will usually be a few business imperatives that are undeniably critical to an organization's future survival. These should be used to help drive SOA initiatives.

- SOA doesn't require big-bang investments in technology with nebulous ROI and prolonged payback periods. SOA is achievable in small increments, and the tangible benefits of software and services reuse immediately deliver bottom-line re-

sults. In fact, there is often a clear ROI at the project level while an organization builds toward the strategic vision of an SOA.

- This phenomenon is one we're unaccustomed to, and often business executives are skeptical. However, the proof is in the pudding. Scope a development project, compare traditional development processes and time to market with a Web services approach, then factor in the enabling infrastructure to operate Web services, and compare.

What you will find is a software solution that costs less, leverages existing IT infrastructure and is reusable. Of course, scaling the SOA to accommodate many Web services requires additional SOA-enabling solutions, such as a Universal Description, Discovery and Integration (UDDI) registry, Web services management solutions and a robust security strategy.

- Perhaps a messaging solution or enterprise service bus (ESB) is needed as well. But the price points for these solutions are significantly

### Best Practice

According to both quantitative studies and in-depth interviews with IT executives, there is a consensus that creating systems based on SOA is the best practice to which to aspire.

SOURCE: IDC, 2004

different from those of enterprise software of the 1990s. In other words, there is a clear cost advantage associated with Web services and SOA across the board.

- SOA requires cultural and organizational discipline as much as enabling technology. Organizations should be prepared to educate and train their business and IT executives on SOA. They should plan to create incentives encouraging and enforcing reuse of services in their SOA. They should also design the SOA governance model, governance processes and policies to be enforced in their SOA — such as reuse policies, security policies and standards policies.

- Many of these challenges are organizational and process issues. Their enforcement, however, is both process- and technology-driven. SOA governance must be backed by SOA solutions that can enforce a Web services security policy or an architectural compliance policy both at design time as well as run-time.

- SOA does threaten entrenched enterprise software application vendors unless they have embraced Web services and SOA as well. Beware of their desperate attempts to market themselves as an SOA or Web services solution. Perform due diligence to make sure. Evaluate their ability to interface to your enterprise systems using Web services. Do a detailed evaluation of how their application functionality is exposed as services to the outside world as well as their ability to consume services.

- If there is no true Web services support, ask for their product road map. Ask them when they will support Web services. Write a performance service-level agreement into your contract with them requiring

## SOA in Large Enterprises

**FORRESTER RESEARCH INC.** recently surveyed 116 North American decision-makers familiar with programming technologies, application software architecture and application platforms. When asked “How would you describe your firm’s approach/status to service-oriented architecture?” those firms identifying themselves as large enterprises gave the following responses:

- 29%** We have an enterprise-level strategy and commitment for SOA
- 19%** We have only department-level strategies for SOA
- 24%** Selected projects are independently using SOA without a clear strategy at the departmental or enterprise level
- 5%** Plan to begin pursuing SOA within 12 months
- 10%** Not pursuing SOA and have no immediate plans to do so
- 14%** Don’t know or not applicable

SOURCE: FORRESTER RESEARCH INC., 2005

Web services support by a specific date. Do not let them off the hook on this, and be sure to follow up.

■ Get to know the various Web services and SOA software vendors. Group them in four fundamental categories of functionality initially,

and then map the rest to adjacent or related categories. The minimal functionality required for an SOA includes the following: Web services management; services registry (UDDI); Web services security; and an ESB or messaging bus.

Some of these categories are clearly defined, while others are a bit fuzzy. Web services security can be accomplished in multiple ways — at the Web services management level, at the enterprise level with single sign-on and access management solutions, or at the XML firewall level with security appliances. In all these cases, the Web service is where security happens.

The “how” of security has many choices. Defining a security strategy and a policy that can be enforced at services design and runtime is essential, and let that drive the technology choices for your SOA team.

This may well be the year of the SOA. If you are seriously contemplating investing in SOA and Web services capabilities, make sure you do so with business intent in mind.

Using the “fix it or else” scenario helps identify clear and compelling business issues that must be solved, as well as potential IT barriers or issues that must be addressed to enable the business outcome.

These are critical business-focused starting points for SOA thinking that will help drive a jointly owned business and IT SOA strategy going forward.

# The SOA Network Effect

**S**OAs ARE IN VOGUE NOW. While many organizations are wrestling with the technical issues of SOAs and Web services, they are overlooking the organizational and cultural aspects of SOAs that will drive business value faster.

This brings up a chicken-or-egg scenario. “Should we build the SOA-enabling technical architecture first and worry about the cultural and behavioral issues later? Or should we focus on education, training and the supporting cultural systems — SOA organizational competencies — that will reinforce the use of the SOA technology?”

In an ideal world, an organization should do both. Building the proper SOA technical and organizational/cultural capabilities will facilitate maximum SOA use by the potential universe of business and IT users and lead to overall SOA success. This is as much driven by technical issues as it is by softer ones — education, training, encouragement, reward and compensation systems, and others. If these two facets of SOA adoption are addressed up-front, SOAs will demonstrate increasing returns — the “SOA network effect.”

The right technical architecture will facilitate faster service creation as well as increased use of the SOA, which will drive increased business value through the combination of more users and services. The SOA network effect will help organizations achieve the business results of an SOA in an accelerated time frame, but only if the organizational, cultural and behavioral processes of SOA are in place.

The SOA network effect is predicated on the tactical and strategic benefits of an SOA combined with processes that drive more services and more users of the available services. Providing IT functionality as

cross-platform shared services in an SOA generates a number of benefits, including asset reuse, reduced integration expenses, greater IT and business productivity, and greater enterprise agility.

Once a portfolio of Web services is available to be leveraged in an SOA, the asset reuse and other SOA benefits multiply and cause the SOA network effect, where the value of the SOA increases as the number of available services and the number of users using those services increases. This benefit compounds over time as the SOA is leveraged internally and externally. SOAs demonstrate increasing returns in this fashion.

The question becomes one of how many services and how many users of those services are required to trigger the accelerated model of increasing returns. And what technical factors are most important to achieving the SOA network effect? And what organization and cultural factors will facilitate the SOA network effect? (Again, what comes first, the chicken or the egg?) Let’s explore some technical and cultural/organizational factors that can facilitate achieving the SOA network effect.

## Four Functionalities

On the technology side, SOAs minimally require four categories of functionality that can be acquired from single vendors or from many in a best-of-breed model: a Web services management platform, a registry (UDDI-compliant), an enterprise service bus or related messaging infrastructure, and Web services security.

A key element of an SOA here is

## Key Lessons

### Four suggestions from leading deployers of SOA:

- Organizational issues become less of a concern as executives begin to see the benefits of a well-managed SOA strategy.
- Leading deployers don’t obsess about how far along a standard may be in a standards body, or that a cat-fight has broken out between competing sectors — they take the specifications and standards as they currently exist to achieve the benefits of Web services, and make modifications as needed.
- Can current IT and network infrastructures process Web services and SOA messages? Companies moving into Web services and SOA need to evaluate existing capabilities and make improvements where they are needed.
- As a Web services or SOA infrastructure grows, there needs to be a mechanism to communicate services availability.

SOURCE: WEBSERVICES.ORG, 2005

## The Big Picture

---

the registry solution. There is debate as to when a UDDI registry should be added to an SOA — after a critical mass of services is already built and running, or upfront, before reaching a critical mass of services such that the SOA merits a registry solution.

A registry solution can facilitate more rapid SOA adoption if it is implemented earlier rather than later in the SOA adoption process. Remember the role of a service registry in an SOA: it facilitates discovery of available services in an SOA and provides advertising or notification for new or updated services that are available to be leveraged.

A registry can be used to encourage developers to publish their services for potential use by a wider community of developers across an organization. A registry can be used by business analysts and business process owners (assuming they can locate services by nontechnical search terms or business-based metadata) to leverage existing services for new or modified business processes.

A registry is the linchpin for achieving reuse of existing services. SOA registries must support extended metadata taxonomies that allow a wider audience of potential users to find and use services based on their roles in the enterprise. The registry may include developer-specific

UDDI technical information, but it should also include business information relevant to the business users — what process a service relates to, high-level descriptions of what a service does and relevant keywords for locating the service.

The potential audience of a service may also be external users in a business-to-business sense, such as trusted trading partners, suppliers and customers. They may have to locate services by the same business or process terminology rather than by technical jargon, and a solid registry approach will accommodate this wider audience of potential SOA users. The registry scenario is an example of a technical feature of an SOA that has great impact on business processes and the various communities, users and roles in an SOA.

### Educate the People

However, don't forget the organizational and cultural issues that attend successful SOA rollout and the achievement of SOA network effects. In order to encourage business audiences to leverage an SOA and the portfolio of available services, they must know about them.

An SOA strategy requires education of both the business and IT communities to realize success. For the business community, education must focus on why an SOA approach is important, how reuse of

Web services can benefit the business, and how to identify business process opportunities that can benefit from SOAs and Web services.

For the IT community, developers may feel constrained by the standards-based model of Web services and SOA. They may feel as if their creativity for solving business and technical problems is being limited: "Why should I use that other division's service when I can write my own better?" These issues can be overcome by a combination of education, encouragement, reward and compensation systems, and personnel and team reviews. Employee reviews especially can be used to drive a culture of IT asset reuse.

### Conclusion

There are many ways to build behavioral and cultural factors into the SOA rollout and adoption process. But keep in mind that SOAs are more than technology. SOAs are a business strategy, a governance model, a technical architecture and a portfolio of services. But SOAs require behavioral and cultural reinforcement to be successful and achieve the SOA network effect.

The combination of proper enabling technologies and cultural and behavioral systems will help drive the business value of SOAs and the ultimate achievement of the SOA network effect.

# SOAs Eliminate Integration Headaches

**W**HAT WOULD BUSINESS BE LIKE without IT integration? What if your CEO could acquire another firm and integrate its information and operations into the existing business and IT architecture without the integration challenges? What if the acquired firm's business processes could seamlessly integrate with yours with zero latency? What if there was no integration effort required at all? What if the IT systems were "preintegrated," where they could exchange information without any incremental integration expense and effort?

Imagine how business would be without two factors that have become ingrained in our expectations of IT today: the IT integration hurdles that attend every business initiative, and the inevitable time lag between the need for the business initiative and the ability of the IT organization to deliver it on time. This is the "zero integration enterprise," and SOA is the way to get there.

So, what is the zero integration enterprise? It is an organization whose business and IT teams are committed to the precepts of SOA and Web services. These organizations see the business value and strategic advantages of SOA and are migrating their processes, applications and skills to support the concept of services, specifically Web services. But before defining the characteristics of a zero integration enterprise, let's examine why this is such an important concept. It starts with IT complexity, and complexity has created the need for IT integration.

Where did all this IT complexity

come from? Accumulation of legacy assets. Why is most of your IT budget focused backward on maintaining the past rather than looking ahead to supporting the future? This is called rearview-mirror budgeting.

This rearview-mirror budgeting problem is legendary among CIOs and is partly responsible for the lack of strategic IT investment that is possible for CIOs today. How back-

### Integration Issues

Survey of 16 North American decision-makers at large enterprises who expect to be using SOA by the end of 2005:

HOW WILL YOU BE USING SOA?

Internal integration	69%
External integration	50%
Multichannel applications	31%
Strategic business transformation	25%

NOTE: MULTIPLE RESPONSES ALLOWED  
SOURCE: FORRESTER RESEARCH INC., 2005

ward-committed is your IT budget? What percentage of the IT budget is allocated to maintaining your legacy investments rather than focused on forward-facing initiatives that help move the organization ahead? This is a real challenge for both business and IT executives today, and it must be addressed.

At what point does IT complexity become an obstacle to business goals and an impediment to achieving even IT's goals? Complexity becomes intolerable for firms when the following actions have been taken or are under consideration:

**1. The firm hires a chief architect.**

**2. The organization creates a central architecture team.**

**3. The firm acquires or develops an enterprise application integration solution.**

**4. The organization forms an internal integration or middleware team.**

If you have one or more of these things in place, your organization is at the point where your integration burden is consuming IT resources, compounding the existing complexity problem and inhibiting business and IT effectiveness. You most likely have a rearview-mirror IT budget, and you are ready to try a new approach. The approach is SOA and Web services, and the goal is the zero integration enterprise.

What is a zero integration enterprise like? This organization can launch new business initiatives faster than its competitors because less IT integration is required to support various business projects. This time-to-market benefit allows faster response to business conditions, customer requirements, competitive threats and increased innovation.

# The Big Picture

---

This organization has a higher return on assets and greater IT productivity than its peer companies because of the asset-related benefits of SOA, such as software component reuse, Web services reuse and extending the capabilities of existing IT systems and infrastructure.

This organization launches new software applications faster than before with reduced effort due to component and Web services reuse. Building upon proven software and services capabilities, the organization spends less time developing new code and more time focusing on business process issues.

This organization uses the 30% of its IT budget previously used for integration projects to solve strategic business problems such as improving customer satisfaction, reducing time to market for new products and increasing sales through various IT initiatives.

This organization implements concepts of agility and flexibility through its SOA initiatives. Agility is

reality and is measured by unambiguous metrics. A preintegrated enterprise has the following characteristics:

- An agile business model that can quickly respond to business challenges, competitive threats and customer needs.
- Greater customer focus derived from reduced effort spent on internal integration and more effort spent on customer satisfaction, partner communication and efficient business processes.
- A business-focused IT organization that no longer must concern itself with assuring interoperability issues but rather can focus on forward-looking strategic issues.
- A flexible IT architecture, based on SOA and Web services, that facilitates superior business performance, enables world-class business processes and is highly efficient with all corporate resources.

Achieving the zero integration enterprise is no simple matter, regardless of how compelling the benefits

are. The zero integration enterprise must first be an executive mandate for the entire organization, and that mandate begins with three simple actions:

- 1. Implement SOA and Web services immediately.**
- 2. Stop integrating now (or integrate with integrating).**
- 3. Enforce asset and services reuse organizationwide ASAP.**

Issuing this mandate to your business and IT organizations is a commitment for the long haul. This is not a fad. This is not a quick fix. And it is not going to be easy. Moreover, the organizational and cultural issues of SOA are as significant as the technology and infrastructure issues. That said, SOA and Web services can help your business, they can enable efficiency, they can reduce costs, and they can create customer and partner loyalty. But you have to get started.

# Web Services: Managing The Building Blocks

**D**ANSKE BANK A/S's trailblazing work to build a SOA had gotten so advanced that it exposed more than 1,000 services from its mainframes and application servers. But the Copenhagen-based bank found itself in a frustrating predicament.

"We couldn't find them," says Claus Torp, the company's chief architect.

The problem threatened to wipe out one of the main benefits of SOAs — reuse. So Danske set about revising its concept of a service, refining its repository and establishing a governance process to enforce best practices.

The result was a collection of 140 services that is far more manageable.

An in-depth look at several SOA pioneers shows that the steps Danske Bank took are key to a company's ability to reuse code, build applications with greater speed and efficiency — and ultimately save money.

But it's not easy, and the implementation sequence is important. Sun Microsystems Inc., for instance, built a registry and set up an architecture review board. But the IT department is just now circling back to do a closer examination of Sun's 80 to 100 Web services.

Karen Casella, an IT director at Sun, recommends that a company starting down the SOA path first look at its business requirements and identify which Web services are needed. "We learned the hard way," she says. "We put some of the infrastructure in place before we completely understood what we needed to have in play."

## The Services

Companies need to figure out which business processes can be turned into services, carefully design and define the services and distinguish them from components.

When Danske Bank began building standard interfaces to expose its legacy programs, it defined a service as "one function." Now it describes a service at a higher level, as a logical grouping of functionality and data, such as "customer" or "account." The company's 140 services are each composed of about 10 "operations," or components, that are essentially more granular services. There are currently more than 1,365 operations. Danske expects to eventually have 250 services.

How well a company can break down its business processes and application functions into services will determine the level of flexibility and reuse it gets, Torp says.

Danske uses modeling tools to develop logical maps of the functional building blocks and business processes. Then it matches the business processes to the services to make sure it has solved the right problem.

"A lot of doing service-oriented development is making sure you can run different business processes on top of the same service building blocks," says Torp. "If you want to

be effective, you have to make sure there is only one place to do the same function."

Cendant Corp.'s Travel Distribution Services division spends a considerable amount of time determining the optimal granularity of its services and service components, according to Chief Technology Officer Robert Wiseman. A service is something that can be called externally through Cendant's business domain model, dubbed Rosetta Stone. A service component, such as logging, is called only internally.

So a "get hotel" service might call several low-level services, such as a latitude/longitude "destination finder" that the company makes available to customers. But Cendant's currency converter is a component, since it currently isn't exposed to customers.

Cendant expects an ongoing project to extract components from monolithic applications to have a big payoff, Wiseman says. For instance, passenger name record (PNR) is a basic unit of data used by booking engines and global distribution systems such as Cendant's Galileo. By making "Super PNR" available as a service, the IT department won't have to maintain six or seven instances of PNR in different applications. The Hartford Financial Services Group Inc. built pockets of Web and other services over three years ago, but its enterprise-scale SOA work didn't start until 18 months ago.

A good candidate for an enterprise service is one that two or more

## Implementation

applications need, says Benjamin Moreland, manager of application infrastructure delivery at the Connecticut-based insurance company. "But not everything should be a service," Moreland warns, noting the potential performance hit from exposing services.

### Establishing the Registry

Vendors may have expected Internet-based registries based on the UDDI standard to spread like wildfire. But early SOA adopters care more about internal registries.

That doesn't mean UDDI is dead, though. UDDI was so important to The Hartford that it chose its registry based on the product's conformance to UDDI 3.0. (Officials declined to name the product due to a company policy against endorsing vendors.) The registry includes metadata describing the services and the means to connect to services via particular transports.

But the UDDI registry isn't meant for everything. Departments continue to maintain local registries for some services they create, because The Hartford is selective about what goes into its enterprise registry.

"We don't want to create a junk drawer of services," says Moreland. "What we feel should be in the enterprise UDDI are services that will give us leverage and flexibility across the enterprise."

Providence Health System uses the Infravio Inc. management framework for its service library, and much to the surprise of company skeptics, its developers are actually reusing services, now that they can find the WSDL files defining the interfaces. "We commonly refer to this as 'Google-izing' Web services," says Michael Reagin, Providence Health's Portland, Ore.-based director of research and development. "They can reuse services with minor modifications in a couple of hours. People are more productive. Everyone's happy." Providence Health's greater concern these days

is managing its growing number of Web services and SOA framework from an operational standpoint. The company has close to 50 composite services, each one comprising one to 20 more granular subservices.

Early adopters that couldn't find a registry to suit their needs built their own. Danske Bank maintains separate repositories for components from its mainframes and J2EE- and Microsoft .Net-based application servers. The repositories replicate between each other, forming one logical repository that essentially is a superset of a UDDI registry, adding operational parameters for functions such as load balancing, says CIO Peter Schleidt. A service integrator agent dynamically selects

the most efficient way to call a service, using SOAP over HTTP or more efficient, proprietary protocols.

Danske also has a structured library for its services and their corresponding interfaces. The library also houses information about the relationships between its functional and process models. There's even a librarian that developers can call for help. But the library didn't launch until a year ago — "a lot later than we should have been doing that," Schleidt says.

### Governance

When push comes to shove, a governance body can help a company stick to its SOA principles. Danske Bank has steering committees in 18 different business areas for product, process and IT development. But when business managers are anxious to beat the competition, they're sometimes tempted to forgo the generic SOA approach if it takes longer to complete. "You need a governance process where you can handle situations like that," says Schleidt. "We always have the time to change things afterwards, so why not try to turn it around and do it right the first time?" The quick-hit approach can have long-term consequences. Danske now has two personalization engines, four interactive customer communication services and four payment-handling applications, Torp says.

Years ago, The Hartford formed a central group called the Property and Casualty Architects Collective to examine how it would adopt a SOA across the enterprise. The group put together a reference architecture outlining recommended approaches, practices and products to be used in a particular context.

"It's about shared architectural thought and reuse of thought processes. That's where the hard work and value is," says James McGovern, an enterprise architect at The Hartford.

A separate application infrastruc-

## Four Challenges

The development of service-oriented applications requires the following steps:

**1. UNDERSTANDING** which processes can be turned into services.

**2. BUILDING** a foundry of application processes. This will come increasingly from business applications that are designed as a set of services.

**3. ESTABLISHING** the granularity of services at the right level to ensure that services are effectively reusable. Too much granularity makes services too specific to be used; too little granularity makes them too general to be used.

**4. FOSTERING** a reuse culture is essential to consistent, repeatable success in capturing and using business processes. It enables an organization to deliver processes as a well-defined set of services and to make those services easily available to developers.

SOURCE: GARTNER INC., 2004

## Implementation

---

ture delivery group is responsible for selecting and implementing the management platform, business process engine and UDDI registry, as well as making sure the WSDL files used to describe service interfaces conform to standards. An architect not involved with a particular project reviews the project's application design to make sure services aren't duplicated.

At Cendant, project managers have that responsibility. The service name and input and output fields are accessible through an XML-based layer in its Rosetta Stone busi-

ness domain model. A single group is responsible for updating the business domain model.

"This is how we control reuse," Wiseman says.

If a business domain owner spots a service already in the registry, the service is flagged as a candidate for reuse. A SOA governance board, largely consisting of IT managers, then takes over. Developers need not apply.

"The programmer is the last person that should make the decision," says Wiseman. "They will always want to write something new."

---

# Web Services Security: Trouble in Transit

**A** TRANSPORT COMPANY'S TRUCKS are scheduled for bogus pickups. A financial services firm's investment data is given away for free. A health insurance provider's private patient data is exposed. These are the disastrous situations that can occur when Web services data is nefariously snatched midstream.

The shareable design of Web services, which gives companies the benefit of easily exchanging data and applications with business partners, also makes them vulnerable to security breaches. Hackers have found ways to tweak the XML code used to tag the data so activity that's actually an attack appears to be valid.

"XML standards are being constructed in bits and pieces, and that's the kind of event that leads to holes that someone didn't think about," says Randy Heffner, an analyst at Forrester Research Inc.

According to experts, hackers have three methods for breaching Web services and XML security: identity-based attacks, in which a hacker poses as an authorized user to gain access to Web services; malicious-content attacks, in which an intruder forces a Web server to perform an unauthorized activity; and operational attacks, in which a hacker manipulates an XML message to tie up server resources. But although the methods are known, safeguarding Web services is difficult because multiple elements must be locked down — the servers, the messages and the applications. Companies must first secure their Web servers and then decide which business partners and employees

will have access to them, how they'll connect to them and which authentication method to use.

## No Small Task

Defense manufacturer Northrop Grumman Corp. experienced that difficulty firsthand. Web services are a major component of its Myngc.com portal, which was expected to take about six months to complete. But because of security requirements such as user authentication, the project took three times as long, says Thomas Shelman, vice president and CIO.

The portal gives Los Angeles-based Northrop Grumman a way to

**XML standards are being constructed in bits and pieces, and that's the kind of event that leads to holes that someone didn't think about.**

**RANDY HEFFNER,**  
ANALYST, FORRESTER RESEARCH INC.

efficiently share ordering and billing data with customers and partners, he says. But while Myngc.com provides greater data access to more people, it also creates vulnerabilities because many users outside the organization have access to business applications.

"[The portal] was a significantly larger task than we thought going into it," Shelman says. "I know a lot of companies that are implementing the same sort of thing, and they don't address the security aspects. They're leaving themselves very vulnerable."

"The need for security goes up exponentially as you're trying to expose applications to your business partners," adds Raphael Holder, vice president of shared services operations at Northrop Grumman. He says the company first grappled with how to provide secure remote access to Web services applications for internal employees and ensure that all users entering the portal were authenticated.

"It sounds simplistic, but we're providing more capability here than we have through remote access in general," says Holder. "We're providing access to portlets that touch business data, not just e-mail."

To secure its data, Northrop Grumman deployed a public-key infrastructure system from RSA Security Inc., issuing tokens to all employees authorized to access the portal. Tokens will also be issued to appropriate partners and customers for different levels of access, Holder says. Eventually, the portal will have

## Implementation

---

more than 120,000 users.

Even with the current technology in place, Holder acknowledges that there are security risks with Web services. "We will have to manage [the portal] very closely, and access will be done on a business-case basis with those partners we highly trust," he says.

### Exploring the Options

As it explores Web services, Wyndham International Inc., a Dallas-based hospitality company, is also looking closely at how it can provide tight security. Wyndham's research shows that it endured more than 9.5 million attempts at information security breaches from May 2003 through May 2004. They included hacker attacks, Web site defacements and viruses.

"Obviously, security is a huge concern of ours," says Mark Hedley, the company's senior vice president and chief technology officer. He says Wyndham plans to get Web services from its provider of central reservation systems technology, Micros Systems Inc., using industry-standard protocols. Hedley says his

### Security Advice

**"Put security risks in perspective. Security is an issue but it should not be allowed to stop an enterprise from benefiting from Web services. Address security issues and put the risks against the benefits of Web services. If consumers had let security issues stop them from using credit cards to purchase from web sites, there would not be an Amazon.com today."**

SOURCE: THE YANKEE GROUP, 2004

company expects Web services to make it easier for customers to obtain reservations and for Wyndham to share data with suppliers. Wyndham plans to finalize its security plan and implement Web services by the first quarter of 2005.

Heffner notes that there are several ways to approach Web services security. One of the most common is to secure applications at the transport layer, with two-way Secure

Sockets Layer connections or a dedicated virtual private network link. SSL can include mutual authentication such as client certificates. Another option, which Heffner says is more suitable for connecting with multiple partners, is to use XML security gateways. These are network appliances that protect XML and Web services from attacks. They feature XML encryption, digital signatures, access control and other capabilities. Many in the industry expect that new standards will also help bolster security.

Heffner says Web services security is improving because of new standards and products. "XML security gateways provide better solutions for attack protection than existed a year ago," he says.

Security concerns shouldn't get in the way of Web services implementations, says Heffner. "You may have to spend more now on security than in three years," when more safeguards are built into Web services products, he says. "The bottom line is, if there's business value for Web services now, you shouldn't be holding it up."

# Message Received?

**W**HEN PLACING AN ORDER OVER THE WEB, it's not unusual for the site to lock up during the transaction — it's all part of the Web experience, and customers know to call a customer service rep to see if the order went through.

But take the human factor out of the equation, as in a Web services transaction. Who's 100% sure that their messages got through, unduplicated and in the right order? After all, you're relying on two applications to communicate with each other to complete the order, using standards such as XML, HTTP and SOAP.

"HTTP is a nice, lightweight mechanism to send HTML or almost any type of format. The problem is, there's no guarantee that the message will get delivered," says Anne Thomas Manes, an analyst at Burton Group in Midvale, Utah. "It offers 92% to 96% reliability, but if you need 99.9% reliability, or at the very least notification if a message doesn't get there, HTTP won't do that for you."

Although analysts don't think reliability is the only problem impeding Web services adoption, it certainly casts a pall of uncertainty over the idea of building anything mission-critical on such a platform.

Some users, such as Peter Osbourne, director of Internet research and development at Dollar Thrifty Automotive Group Inc. in Tulsa, Okla., say they do just fine with TCP/IP and HTTP. "We use Web services extensively [for Web-based car reservations] but have not encountered message reliability problems," he says. Dollar has mechanisms to detect message bottlenecks, but Osbourne says the company hasn't lost any messages.

At companies that need a higher degree of reliability, it's a matter of

"hacking their way through the jungle with a machete," says Ron Schmeltzer, an analyst at ZapThink LLC. These users build reliability into the business logic of the application or employ a mature messaging platform such as IBM's MQSeries.

## Once Isn't Enough

ShopNBC is one organization that has given Web services reliability a lot of thought, according to Steve

**HTTP is a nice, lightweight mechanism to send HTML or almost any type of format. The problem is, there's no guarantee that the message will get delivered.**

**ANNE THOMAS MANES,**  
ANALYST, BURTON GROUP

Craig, chief technology officer and vice president of IT development at the shopping network, which is broadcast into 56 million homes.

ShopNBC relies on Web services to fulfill customer orders over the Web and to receive real-time pricing updates from its participating merchants. Because it's such a mission-critical endeavor, Craig and his group structured a three-layer system that supports three types of Web services messaging: real time, near real time and periodic. Each layer reinforces the others to create message redundancy in case the first message doesn't get through.

"If a real-time transaction fired and was missed, there'd be another job downstream to cover that need," Craig explains. Say, for example, a merchant system sends a message to the Web services system at ShopNBC to take a 10% markdown on a product that's airing on TV. If the real-time message wasn't received, another message sent to another server would eventually lower the price — maybe not while the product was airing, but on the next database sync.

Another example is when customers order products over the Web. The Web server attempts to commit the order in real time with the back-end server, but if the server doesn't respond, the order is sent to a separate server to be synced up later.

## Build, Don't Add

As ShopNBC demonstrates, reliability must be considered from the very beginning. "You never add reliability to something after the fact — it's not something you can just slap on top of the application," says Don Reeves, vice president of engineering at Black Pearl Inc. in San Francisco.

Black Pearl's B4 platform uses

## Implementation

---

Web services to collect data from disparate sources to provide real-time customer profiles and data analysis. For instance, it sends financial services agents daily lists of high-priority prospects, as well as real-time alerts on products that might interest particular customers. The system needs to collect data from a variety of sources, such as legacy customer databases, marketing campaign systems and live data feeds on stock prices.

Reeves estimates that half the code that makes up B4 is aimed at functionality, and half is intended for error checking. "Whenever you connect to a data source, you have to assume it will fail and have a planned response," he says.

Another approach is to employ a reliable transport protocol, such as a message queuing (MQ) service, rather than using HTTP to transmit SOAP messages. The dominant MQ service is IBM's WebSphere MQ,

which acts as a mediator to guarantee delivery by storing messages locally until it receives delivery acknowledgment.

"You can send it over SMTP or FTP or an MQ system. Web services is completely independent of the underlying protocol," Manes says. The trouble is, MQ systems are proprietary — IBM's WebSphere MQ doesn't talk with, say, Sonic Software Corp.'s MQ system. This is less of a problem if you're just using Web services internally, but even then, it's an expensive proposition. According to Manes, MQ deployments can go into the seven figures.

### Using Your Header

To get beyond dependence on a proprietary protocol, companies will have to code specifications into their SOAP headers, Schmeltzer says — such as having a system try to resend a message if an acknowledgment of receipt isn't received

within 300 milliseconds.

Today, Schmeltzer points out, individual companies need to do this sort of coding themselves, which can be extremely difficult. However, this coding will eventually be built into standards such as WS-Reliability and WS-ReliableMessaging.

In the midst of this industry confusion, the lack of a reliable messaging specification is pretty far down the list of what's stopping people from implementing Web services, says Dwight Davis, vice president of Summit Strategies Inc. in Kirkland, Wash. "There's a lot of baggage that comes into play on the list of factors," he says, including the relative newness of the technology and the need to focus on day-to-day crises rather than long-term architectural change. Just the same, before mission-critical Web services applications enter the mainstream, reliable messaging will have to become less complex and costly.

# Hammering Out Web Services Links

**V**ERIZON COMMUNICATIONS INC., a New York-based telecommunications company, averages about 2.5 million to 3 million Web services transactions a day, anchored by its mostly homegrown SOA, a platform that was two and a half years in the making. Dubbed the IT Workbench, the SOA supports the design, deployment and management of Web services. It went operational early last year and has helped the company slash its IT budget by 50% by eliminating redundant systems inherited from the merger of Bell Atlantic and GTE, which spawned Verizon.

Verizon has also tackled some of the most vexing hurdles associated with Web services as part of the IT Workbench project, such as managing and securing the services, charging for reuse and monitoring the performance of service-enabled transactions.

## IT Workbench

The project was born in 2002 as executives began looking to reduce inefficiencies in software development, says Shadman Zafar, Verizon's senior vice president of architecture and services. Consolidating application development was key for Verizon, which found itself with multiple groups often duplicating efforts after the merger.

Executives focused on the 250 most-important business transactions the company performed, such as verifying customer credit histories and looking up customer information. On average, each transaction had been developed five to 25 times; one was deployed 45 different times, Zafar says. The duplication was draining developer productivity and created needless ongo-

ing maintenance costs.

The company decided to use Web services to expose the application programming interfaces of common transactions as XML, which could be consumed by the SOAP standard and used by multiple lines of business. Zafar says he spent much of 2002 and 2003 evangelizing to developers about the potential benefits of Web services—especially to the two “religious” camps of .Net and Java developers. Despite some initial resistance, these groups warmed to Web services once project leaders demonstrated that the standards could allow .Net developers to consume Web services-enabled Java applications and Java developers to use .Net Web services, Zafar says.

His efforts to promote the use of Web services were backed by Verizon's CIO, who included target metrics for Web services usage as performance measurements for company vice presidents. “Verizon took a very aggressive view of Web services,” Zafar says. “We were not toying with it. We took it as a business metric, and we had to meet a very tough business metric.”

Zafar set an initial target in 2004 of building 10 applications and 10 transactions on the IT Workbench as Web services. The company instead built 57 applications and 200 transactions. At the beginning of the year, Verizon was supporting 10,000 Web services transactions per day; by the end of 2004, the daily average had skyrocketed to 2.5 million to 3 million per day.

Anne Thomas Manes, an analyst at Burton Group in Midvale, Utah, says this number of transactions per day qualifies as one of the largest corporate deployments of Web services.

“Verizon appears to have embraced ‘true’ SOA,” she says. “Their goal was to reduce redundancies by building shared reusable services. A few other companies have also embraced SOA, but most companies that I’ve worked with are still more focused on using Web services for integration as opposed to a real architecture of applications based on SOA design principles.”

Now, Verizon is filing patents for some of the technology it built to anchor its use of Web services, and both BEA Systems Inc. and IBM have approached the company about licensing its homegrown platform so they can offer it as part of their application server software stacks, Zafar says.

## Institutionalizing Web Services

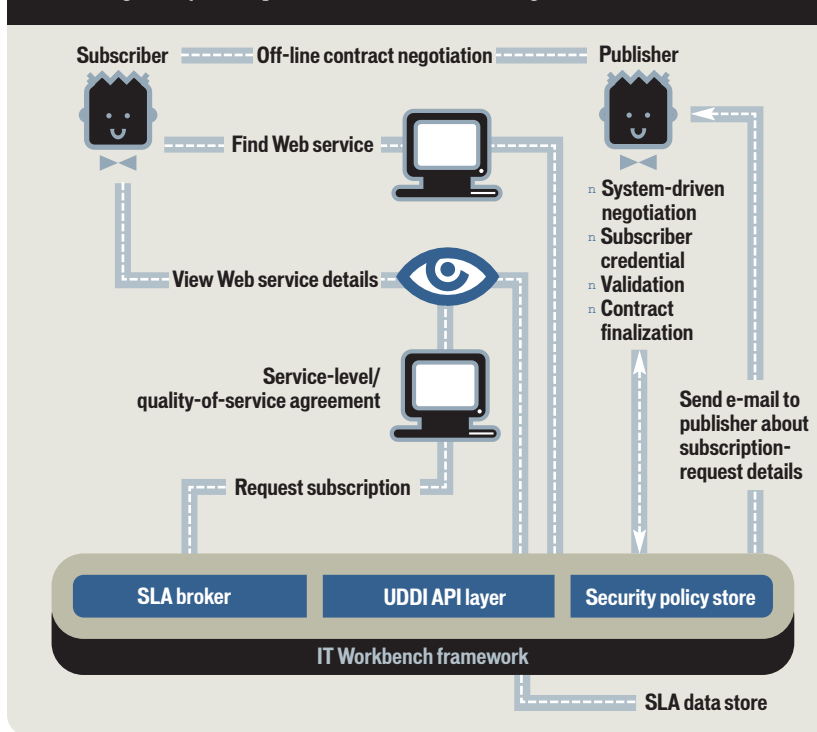
With the CIO's mandate in hand, Zafar's group in 2004 set about institutionalizing the use of Web services. After an initial test project to demonstrate that Web services would allow Verizon's billing and wholesale departments to pass a customer record between systems using SOAP, Zafar worked on a process to ensure that development

## Field Report

### Using the IT Workbench

For developers, subscribing to a Web service is the central step in Verizon's Zero Integration process [QuickLink

53718] built on the IT Workbench. A developer must subscribe to a Web service before using it. The provider validates the consumer after agreeing to service-level parameters. The subscription is then stored, and a policy is created for allowing consumer access.



projects would focus on Web services. At the beginning of 2004, Verizon began funneling each development project for key business transactions through two groups. One ensured that new applications represented useful transactions that could be used by more than one group in the organization, and the other identified which transactions could be deployed as Web services. In addition, Zafar required all new development projects to get his approval before going into production.

"Key transactions have to be developed as Web services . . . so it is a common platform so other consumers of the same services . . . can use them," Zafar says. "If developers want to realize the dream of getting an application into production one day, they have to pass through this gate."

Verizon has limited access to the Web services to development projects to further reduce duplication. Web services transactions can't be added to applications already in production

"Each development group is a publisher of services and a consumer of other people's services," Zafar notes. "You are almost forcing a reduction of duplicate effort and leveraging cross-portfolio development."

On the technology side, project managers began designing a framework to handle the cataloging, security and management of the Web services.

The top priority was making the framework easy to use, says Ruchir Rodrigues, Verizon's executive director of architecture and eServices.

To keep maintenance costs low, Verizon chose a decentralized, agent-based approach to its SOA rather than a broker-based system where all service requests are handled by a server or set of servers, Rodrigues says. "We didn't want to get into the maintenance hassle of maintaining these servers with fail-over," he adds.

Verizon custom-developed agents or small code libraries that sit on all its application servers to intercept various pieces of information about the services, like usage, response time and IP address, as they flow back and forth. This information is put into a log file and shipped to the IT Workbench portal to be processed.

The agents support the subscription, management and dashboard layers of Verizon's SOA. The subscription layer allows developers to publish and consume Web services and to set up service-level agreements regarding their use of the services. By linking developers through the portal, Verizon supported its goal of making the use of Web services as nonintrusive as possible, Rodrigues says. "If I didn't have an IT Workbench platform and I wanted to use somebody else's Web service, I would have to call him," he says. "We wanted to start reducing that interaction and making it easier for people to get to the Web services." IT Workbench project manager Mehul Shah, who helped design the management layer, says he relied on his experience designing network management systems. The agents sitting on the application servers act as "sniffers" to log different management data at the Web services endpoints, he says.

The third layer of the SOA is a dashboard application that allows Verizon to track Web services transaction volumes by line of business on a daily basis. IT has until recently used this information to tweak hardware resources to support load levels. However, early this year, Verizon

## Field Report

---

IT management decided that because the system is now so big, the company will move responsibility for the operation of the IT Workbench to the data center, Rodrigues says.

### External Integration

The IT Workbench is focused on internal integration using Web services. For external integration with business partners, Verizon knew it wanted a hardware gateway to insulate its intranets from the outside world. But it struggled to find a vendor that could meet its performance demands—evoking a service in less than 10 milliseconds, according to Rodrigues.

Finally, Verizon tested a management gateway from Santa Monica, Calif.-based SOA Software Inc. (for-

merly Digital Evolution Inc.) that the company now uses for managing and securing Web services between Verizon and telemarketing partners to exchange customer data.

Verizon is moving deeper into the transactions used by each line of business and plans to support 33 million Web services transactions per year, says Zafar. But the SOA has already helped the company to break down many of the walls separating its 7,000 developers, he adds.

“This has almost become a virtual meeting place of developers,” Zafar says. Developers talk a lot more across organizations because of the platform. “When they have questions,” he says, “they will call another developer as opposed to developing something else.”

---

**See our full selection of Executive Bulletins at the Computerworld Store:**

**<https://store.computerworld.com>**

**Computerworld has had Executive Briefings and Bulletins on many subjects including Outsourcing, Wireless, Storage, ROI and Security.**