



Programmers unproductive?

Try an efficient alternative to SOA.

Smart Web App Development

By Roger Smith

CIOs AND SYSTEM ARCHITECTS find themselves stymied as they try to sledgehammer complex SOAs into their enterprises. Top-down, “if you build it, they will come” approaches to service-oriented architecture often wind up failing—sometimes spectacularly. Instead of better aligning business processes and IT departments, as was promised, too many employees remain oblivious to expensive SOA initiatives. And those optimistic ROI projections? Forget ‘em.

The lesson is that more often than not, simpler is better.

A growing number of companies are finding that lower-visibility Web-oriented architecture (WOA) developments, spawned through grassroots movements, are a better route to the service-oriented architecture. WOA, like SOA, is an architectural approach to system design, though WOA is resource-oriented rather than service-oriented. What’s the difference? While the core SOA design unit is a reusable service that fulfills a distinct business function, resource-oriented services are more limited and data-focused.

SOA and WOA work at different layers of abstraction. SOA is a system-level architectural style that tries to implement new business capabilities so that they can be consumed by many applications. WOA is an interface-level architectural style

that focuses on the means by which these service capabilities are exposed to consumers. Governance, quality of service, security, and management are of equal importance, whether the functionality is being delivered via SOA or WOA.

Although some pundits argue that only startups and Web-centric companies are getting behind WOA, IBM WebSphere CTO Jerry Cuomo has become a champion via Project Zero, IBM’s WOA-based framework, now in beta and scheduled to be released as a product later this year. Cuomo offers the following example gleaned from helping his 12-year-old son do a homework assignment:

“He decided he wanted to produce his assignment as a Web site. With a little coaching, I convinced him to look at Google’s and eBay’s services and have a little fun placing banner ads on his homework. Using the Google and eBay Web APIs, which support simple REST URLs, he was able to easily cut and paste the code needed (without really understanding what the code did) into his assignment, and presto, he was linked into both Google’s and eBay’s powerful SOAs. Several months later he has made over \$5 on his homework from ad revenue. This is a real motivator to do homework!”

One IT exec who’s really been doing his Web-oriented architecture homework is

Mick Coullas

IN DEPTH / WEB DEVELOPMENT

Steve Bjorg, co-founder and CTO of MindTouch, a company that marries an open source wiki collaboration and content management platform with IT governance. Launched in 2007, MindTouch's Deki Wiki is now deployed by FedEx, Fujitsu, Gannett, Microsoft, Siemens, the U.S. Army, and other organizations worldwide. "By going the WOA-plus-REST route instead of SOA-plus-SOAP, the requirements for extending the application dropped considerably," Bjorg says. "There is no SOAP processing stack with complicated WSDL documents, an SOA registry, and what have you. Instead, someone can easily create an extension to Deki Wiki using any number of computer languages." (More on the alphabet soup of these architectural acronyms later.)

Companies worried about whether a Web-oriented architecture can replace SOA should relax. WOA and SOA are complementary architectural styles, even if pundits say you should use one or the other. In some cases, a WOA will serve the purpose. In others, you may need to scale up to a SOA. But what's certain is that companies shouldn't let themselves be pressured into jumping headfirst into SOA if they're not ready.

It's understandable that companies are attracted to the SOA concept, which seems to offer a way to get applications written in different languages and on various platforms to act as if they all were written in a single

programming language and reside on one computer. Distributed computing often is considered the holy grail of programming, and it's something vendors such as IBM, Microsoft, and Sun have been working on for more than 20 years, with varying degrees of success. SOA is the latest, and thus far greatest, attempt to solve the distributed computing problem, but it seems to be running into many of the same troubles that bedeviled earlier approaches based on Corba, DCOM, and Java/RMI standards, where software from different vendors didn't always interoperate.

IT pros have expressed skepticism about SOA's promised return on investment. A 2007 *InformationWeek* Web survey of 278 IT pros found that 32% of those using SOA said those projects fell short of expectations. Of those, 58% said their SOA projects introduced more complexity into their IT environments, and 30% said they cost more than expected. Out of all respondents using SOAs, just 10% said the results exceeded expectations.

InformationWeek's survey results were confirmed in an August 2007 report by Nucleus Research, which found that only 37% of 106 organizations it surveyed actually were realizing ROI from their investments in SOA technology and programming. Earlier this year, Burton Group analyst Anne Thomas Manes expressed a similarly negative view of the current business value

Impact Assessment: Web-Oriented Architecture

● Benefit

● Risk

IT organization

WOA uses simple Web formats and protocols such as HTTP, XML, REST, and JSON as the glue to allow Ajax applications, mashups, and more to be brought together quickly with Rails and other rapid-development tools.



SOA uses WS-Security and other sophisticated standards to lock down data, while WOA primarily employs the less-stringent HTTPS. Rapid development may mean governance, quality of service, and management get short shrift.



Business organization

Pilot WOA projects can boost employee productivity and speed up stalled SOA initiatives, which often have fallen short of desired levels of integration.



Used in the wrong place, WOA transactions may put sensitive information at risk, with serious competitive and legal ramifications. Information loss may result in fines or lawsuits.



Business competitiveness

WOA's rapid development can increase business agility, helping companies meet customer, employee, and partner expectations.



WOA projects can lack alignment with the overall enterprise architecture; this may lead to overlapping functionality and new IT silos, something SOA was meant to eliminate.



Bottom Line

In many ways, WOA projects are a no-lose business proposition, as lots of IT projects are built quickly using simple REST techniques, with the market deciding which are the moneymakers to concentrate resources on. Guard against inattention to governance and ensure that a focus on immediate results doesn't come at the expense of investing in long-term infrastructure.

of SOA: "Thus far I have interviewed only one company that I would classify as a SOA success story."

Could starting small with a WOA be the answer?

SOA AND WEB SERVICES

Many people confuse SOA and Web services. SOA is about design; Web services are a specific technology set that supports distributed computing. Web services make it easier to create a service-based system, but only if your developers are using SOA design principles, where functions are packaged into modular, shareable, distributable services that can be used and reused by multiple consumers. If they're not, you're likely to end up with limiting, "stovepiped" nonintegrated applications, especially because Web service middleware isn't getting any less complicated.

Web, Representational State Transfer (REST), and is simpler than WS-* but not as flexible.

Where WS-* has a standard message format and defined parameters for areas like security and reliable message delivery, the WOA approach means that business functions sometimes get hard-coded into an organization's infrastructure without the clear separation of concerns that WS-* promotes. WOA supporters make up for this limitation by promoting a set of best practices based on REST principles, such as using uniform interfaces to access all application resources. The most zealous advocates of this approach go so far as to call themselves "RESTafarians," and they're vocal in their claims that REST is a silver bullet that can be used for practically everything.

The WOA versus SOA debate is unnecessarily con-

When IBM and Microsoft introduced the first Web service middleware framework in 2000, called Web Services Framework, or WSF, it was based on a small set of specifications that included Simple Object Access Protocol (SOAP), Web Services Description Language (WSDL), and Universal Description, Discovery, and Integration (UDDI). Developers initially praised the framework for its simplicity, but over the last eight years the number of specs WSF must support has grown to more than 50, in various stages of completion and ratification.

In response to the complexity and fluidity of the WS-* standards, a small but vocal cadre of enterprise architects now espouse the back-to-basics WOA approach, using "plain old XML" over HTTP. This method is based on the architecture behind the World Wide

tentious. It's obvious that both REST and WS-* style SOA have their place, so IT groups should stay focused on architectural issues rather than implementation minutia. IBM's Cuomo offers this pragmatic assessment: By providing simple Web APIs, eBay and Google, which also support SOAP and other methods to access their systems, "have provided an on-ramp into their SOA that has opened up new business opportunities for all involved"—such as kids making money on homework assignments.

AMAZON'S STORE

Developers have shown their enthusiasm for lightweight REST-based Web services by voting with their mouse clicks. When Amazon.com first tested the pop-

ularity of SOAP versus REST interfaces to its Amazon Web Services, in 2003, more than 85% of developers chose the REST/WOA interface, says Jeff Barr, Amazon's chief Web services evangelist.

"REST is easy to demo—just load a URL into a browser," Barr says. "The results are self-evident, and the mental wheels start to turn when people see this in action."

Amazon's strategy since then has been to give developers almost unlimited access to its product database, whether they're inside or outside the company's firewall, and the result is that Amazon Web Services

DIG DEEPER

SERVICE IN THE CLOUD Both SOA and WOA allow the enterprise to get services from the Internet. Download this *InformationWeek* Report on all things cloud at: informationweek.com/1192/report_cloud.htm

See all our Reports at informationweekreports.com

Externally, Amazon Light, created by former Amazon programmer Alan Taylor, is one of thousands of independent sites incorporating Amazon's product data and programming

tools. Other examples of innovative new sites that use AWS include WeoGeo, which sells customized maps and aerial imagery, and TuneCore, which lets indie musicians upload and deliver their music for a flat fee to iTunes, Amazon MP3, and a dozen other digital retailers. Pay a small yearly maintenance fee for access to the service, which can also include an online merchandise store to sell CDs and T-shirts, and voila! You're ready to put your college garage band back together.

has been growing like gangbusters. In January, Amazon CEO Jeff Bezos announced that AWS now consumes more bandwidth than does the entire global network of Amazon retail sites.

Further proof that Amazon's Web services platform fuels innovation among developers and partner retailers can be found in recent product offerings, including Amazon Simple Storage Service (S3), an online storage service that startups and enterprise clients alike use for Web and image hosting as well as backup, and Amazon Mechanical Turk, which coordinates small units of work among large groups of people over the Web and which has been used to do things like scour satellite data for prominent missing people, including computer scientist Jim Gray and aviator Steve Fossett.

In addition to using a REST design philosophy, these last two sites, TuneCore and WeoGeo, were developed using Ruby on Rails, one of several frameworks, including Restlet (for Java) and Django (for Python), that incorporate REST-based modes and features. All these frameworks make it easy to do basic database operations while incorporating Web resources. The latest version of the Rails framework, for example, has REST baked in at very low levels, including tools to help with HTML, route incoming requests, and make it simpler to parse URLs, XML, and JavaScript Object Notation. JSON is a lightweight data interchange format that's very big on the Web and with startups, but it hasn't yet developed a large following with big companies. As a lightweight alternative to XML, JSON is especially

useful to store data to disk or transmit it over a network without the need for a verbose markup language.

These capabilities have led to great mindshare for Ruby on Rails. Developers especially like Rails because it can analyze a local or enterprise database schema and automatically generate most of the methods needed to work with data on the Web. Rails is a good choice for building WOA services that follow the REST application architecture, but it's not as strong for developing Web services based on SOAP and the WS-* standards.

SCALING RAILS APPS

WOA applications are often typecast as useful only for simple database operations, and not able to scale for enterprise-class performance. Both of these problems seem to be facing Twitter, the popular on-the-fly

While acknowledging that Rails will likely never scale as well as Java, Tate maintains that Rails is getting better, and quickly. "Work on new virtual machines will help tremendously," he says. "My main advice is to solve the problem in front of you. If you don't have a problem with productivity, then don't use Rails. If your problems are not related to technology, don't look for a technological solution. But if you're looking for better productivity and you're building new database-backed Web apps that will have some SOA implications, Rails and REST can help tremendously."

REST VS. SOAP: A MIDDLE WAY

Ed Lyons, enterprise architect at Keane, a global services firm whose specialty is IT and business transformation, says that if Amazon, Google, and Ya-

social networking site, which was developed with Rails. After Twitter failed repeatedly during Steve Jobs' keynote address at the Macworld Expo in San Francisco earlier this year, prompting frustration from end users and concern about the staying power and future of the company, more than a few pundits suggested Twitter should abandon Rails as its Web framework and start from scratch with PHP or Java.

One person who isn't advising Twitter to jump ship with Rails is Bruce Tate, a Java and Ruby on Rails expert and author who says REST applications can scale with Rails just fine. "Developers should not concentrate on scale first. It's the wrong problem," Tate says. "Developers should focus on productivity. Rails got Twitter to the point where they have to worry about scale."

hoo have made the move from SOAP to REST, then it's high time for other companies to consider following their lead.

"I tell my large corporate clients that their requirements can't be more strenuous than those of a \$30 billion-dollar Internet-focused company like Amazon.com," Lyons says. "The complexity of the SOAP WS-specifications is unnecessary most of the time. It's better to have simple standards that are broadly useful but don't cover everything."

Asked whether WOA or SOA is the best way to do business transformation, Lyons says it doesn't have to be an either/or answer. If you accept WOA/REST as the grassroots way of doing SOA, and WS-* style WSDL-based SOA as the top-down way, then there's a

strong business case for using WOA/REST as a way to get your feet wet with SOA—whether or not your company wades any deeper into the SOA stream.

Mozilla's chief evangelist, Mike Shaver, offers this explanation of why Mozilla went the WOA route in adopting MindTouch's Deki Wiki as the future platform for its developer community. "Mozilla has a large volume of developer-relevant information, ranging from traditional documentation and sample code to test suites and bug-tracking data, as well as a number of active discussion forums and RSS streams," Shaver says. "More than any other wiki system we looked at, Deki Wiki feels designed to be extended as a platform for Web applications. ... Whipping up a new extension or integration point is easy enough that even a chief evangelist can do it."

DESIGN IS EVERYTHING

As we said at the outset, SOA isn't fundamentally about technology; it's about design. It's possible to implement a service-oriented system using any type of distributed computing technology: HTTP/REST, SOAP, Corba, DCOM, RMI, Jini, Fax, Telex, or even (if they're handy) tin cans and string. A simpler technology such as REST can be a great way to make SOA successful because it emphasizes the importance of establishing communications between endpoints with your tin-can telephone, as opposed to fretting about the size, shape, and composition of the various receivers and transmitters. IT groups must decide if it makes more strategic sense to get their endpoints talking or to focus on the clarity and security of the conversations. One doesn't preclude the other; rather,

Surprisingly, the argument that WOA is compelling for simple ad hoc development but that companies need to stick to the full Web services SOAP stack if they want secure, reliable, manageable services is also being refuted by recent advances in a new generation of XML hardware appliances. These devices—by the likes of Cast Iron Systems, Cisco, IBM DataPower, and Sonoa Systems—secure, accelerate, and route XML and can make the various WOA/SOA/Web 2.0/software-as-a-service preferences of corporate customers and partners work together. To that extent, existing SOAs can be extended with resource endpoints, and new WOAs can be extended with service-oriented end points. In many cases, hybrid systems may make the most sense.

it's a matter of priorities and resource allocation.

It's important to remember as well that if you go with a lightweight, bottom-up WOA approach, you won't want to spend too much time decorating and WOA-izing your network tree, since at some point it's probably going to get thrown into the yard, along with the other discarded models that your end users have become enamored of over the years. This gets down to the fundamental difference between design and implementation: Implementations always get thrown away at some point, whether they're based on top-down SOA or bottom-up WOA—but good design is forever. In Web years, anyway.

Write to Roger Smith at rsmith@techweb.com.