



SOA Practitioner's Crash Course **Techniques to Guarantee Success**

David S. Linthicum
CEO, Linthicum Group LLC & InfoWorld's Real World SOA Blogger
David@linthicumgroup.com

Overview

Service-Oriented Architecture (SOA) is the modern notion of connecting systems together at service levels, thus enhancing their value and promoting reuse and agility. Indeed, enterprises are racing to enable their existing applications to externalize services, as well as building the appropriate integration infrastructure around it.

So, what's an SOA? In short, an SOA is a strategic framework of technology that allows all interested systems, inside and outside of an organization, to expose and access well-defined services that may be furthermore abstracted to orchestration layers and composite applications.

Truth-be-told, while we've understood the value of SOA for some time now, the concept is still new to most enterprises. Not until the advent of Web services have we had a widely accepted standard and enabling technology that allows us to access all types of systems through a common services interface. In fact, we may be at a point in time where more is understood about the technology than the ways in which it fits into the enterprise. Organizations seem to adopt Web services without thought of strategic fit and function. Adoption is only half the battle.

To address strategic concerns, many enterprises are attempting to figure out how to best leverage SOA within their firewalls, as well as between organizations that are, or should be, part of their business processes. In other words, the intent is to both understand the enabling technology and put it to good use. This seminar will attempt to take the mystery out of that process.

About the Seminar

Many may understand the notion of SOA, but very few have any idea how to get there. Frankly, there is no hard and fast rule as to how one builds an SOA within their organization. Clearly, SOA is a situational thing and your mileage may vary.

However, some common patterns are emerging which may provide a step-by-step guide toward implementing an SOA, either on the fast track (revolutionary) or the slow track (evolutionary)—as long as you're on the right track.

The purpose of this seminar is to provide a step-by-step guide for those who design and build SOAs. This seminar will walk through all of the logical steps, including defining key artifacts and providing examples of those artifacts (e.g., design documents). In essence, this seminar serves as a meta-methodology of sorts, taking the mystery out of moving an organization to a more service-oriented and real-time state.

Instructor

David S. Linthicum
CEO, Linthicum Group LLC
InfoWorld's Real World SOA Blogger
David@linthicumgroup.com

David S. Linthicum is an internationally known application integration and service oriented architecture expert. In his career David has formed many of the ideas for modern distributed computing including EAI (Enterprise Application Integration) and B2B application integration, and Service-Oriented Architecture (SOA), approaches and technology in wide use today.

David is currently the CEO of BRIDGEWERX, the only company to offer integration on a SaaS (software as a service) platform, and is the former CTO of Mercator. He has held key technology management roles with SAGA Software, Mobil Oil, EDS, AT&T, and Ernst and Young. In addition he was an associate professor of computer science for eight years. David has authored over 500 articles for major computing publications, and has monthly columns in several industry magazines. He has authored or co-authored six books including *David Linthicum's Guide to Client/Server*, *Intranet Development* and *Next Generation Application Integration*.

Seminar and Workshop Outline

Step 1: Understand your business objectives, and define success. Let's face it; we're running a business, and the technology you layer into that business should add value. In other words, make for a better bottom line. Thus, it's very important to define the business objectives upfront, including what defines success. Pretty scary stuff for technologists, but absolutely essential if you're to create an SOA that benefits a business. If you find this difficult to define, perhaps you need to evaluate the need.

Step 2: Define your problem domain. You can't boil the ocean, thus you must define the scope of your SOA within an enterprise. Most SOAs are best implemented in small steps, such as moving a single division, or portion of a division, to SOA, if needed, instead of an entire enterprise all at once. Small successes lead to larger, more strategic successes over time, and you need to establish the demarcation lines at the beginning of the project to provide better focus and understanding.

Step 3: Understand all application semantics in your domain. You can't deal with information you don't understand, including information bound to behavior (services). Thus, it is extremely important for you to identify all application semantics—metadata, if you will—that exist in your domain, thus allowing you to properly deal with that data.

Step 4: Understand all services available in your domain. Service interfaces are quirky. They differ greatly from application to application; custom or proprietary. What's more, many interfaces—despite what the application vendors or developers may claim—are not true service interfaces, and you need to know the difference. Services provide behavior as well as information, thus they are service-oriented. There are some services that just produce information; those are information-oriented and should not be included in this step. We are only interested in the former at this point.

Step 5: Understand all processes in your domain. You need to define and list all business processes that exist within your domain, either automated or not. This is important because, now that we know which services and information sources and sinks are available, we must define higher level mechanisms for interaction, including all high-level, mid-level, and low-level processes. In many instances, these processes have yet to become automated, or are only partially automated.

Step 6: Define new services, composite services, and information bound to those services. This is self explanatory. You must define all new services that are to make up your SOA. These will fall into one of three categories.

First are services exposed out of existing systems, or **legacy services** such as ERP, CRM, legacy, etc. These types of services are defined for you, since the

services are one-to-one representations of internal application functions or interfaces exposed as Web services (typically) to facilitate integration. You should note that we are calling these 'new services,' even though many are pre-existing, because now they are accessible by your SOA, exposing true services and not proprietary interfaces. Some of these may be established through a simple upgrade of an enterprise application (ERP, CRM, ERP, etc.) to a service-oriented version (e.g., exposing existing behaviors as Web services).

The second type of services is **composite services**, which are services unto themselves that are made up of many different services. In many instances, these services are mere interfaces to many other services and don't add much, if any, additional functionality. These are complex services, since there are so many dependencies as well as information bound to composite services that you must understand before creating your SOA.

Finally, **scratch built services** are services that are built from the ground up to be a true service. These are typically newer applications and are more under your control and more useful since you're building them with a SOA in mind, and thus designing them to provide services.

Step 7: Define new processes, as well as services and information bound to those processes. At this point we should understand most of what is needed to define new processes, as well as bind them to existing processes, and automate processes previously not automated. New processes should be defined that automate the interactions of services as well as information flows to automate a particular business event or sets of events.

Step 8: Select your technology set. Many technologies are available, including application servers, distributed objects, and integration servers. The choice of technology will likely be a mix of products and vendors that, together, meet the needs of your SOA. It is very rare for a single vendor to be able to solve all problems—not that this reality has ever kept vendors from making the claim that they can.

Step 9: Test and evaluate. To insure proper testing, a test plan will have to be put in place. While a detailed discussion of a test plan is beyond the scope of this seminar, it is really just a step-by-step procedure detailing how the SOA will be tested when completed. A test plan is particularly important because of the difficulty in testing an SOA solution. Most source and target systems are business-critical and therefore cannot be taken offline. As a result, testing these systems can be a bit tricky.