

The SOA Magazine

Feature Article



An SOA Practices Checklist for Building Implementation Roadmaps

by Nitin Gandhi

Published: August 28, 2006 (SOA Magazine Issue I: September/October 2006, Copyright © 2006)

[Download this article as a PDF document.](#)

Abstract: It's been well documented how service-oriented architecture (SOA) can help organizations achieve strategic benefits that can ultimately result in streamlined IT environments and increased profit margins. However, to successfully carry out the process of incorporating SOA into an organizational environment requires that various IT groups be coordinated in an effort to adopt SOA in a standardized manner. This is where accepted practices become useful. This article provides a master list of common practices, field proven by a number of SOA projects. Also supplied is a template that can be used as a checklist for developing SOA implementation roadmaps specific to an organization's transition project requirements.

Introduction

Although SOA is an accepted means of organizing automation logic in such a manner that it fosters reuse, growth, and interoperability throughout the evolutionary cycles of an enterprise, it is not itself a solution to domain specific problems. By applying practices, we address the unique considerations that come into play when having to phase service-orientation into enterprise domains. Planning this staged approach leads to a controlled and scheduled delivery of key design specifications, governance plans, and, ultimately, actual services - all part of a master implementation roadmap.

For example, it is difficult to think of the technical service interface as a business interface. The propensity is to use Web services as just another technical extension to a particular automation environment. However, if an IT group comes up with a set of rules to design service interfaces as business interfaces and enforces this practice across projects, the probability of creating services that represent "truly" service-oriented automation logic is significantly increased.

This document attempts to provide a set of concrete practices and considerations for implementing service-oriented architecture. While a number of attempts have been made to define a global SOA roadmap, it is a difficult proposition because every organization is unique with different priorities. The approach proposed here is to first identify all practices that may need to be in place and *then* to define business needs, risks, strategic objectives, and all the other factors that funnel into a customized roadmap.

SOA Practices

Let's start with identifying some of the key areas in which practices need to be considered as part of typical strategic SOA planning efforts:

1. Service Monitoring
2. Exception Management
3. Version Management
4. Service Management and Deployment

5. Policy and Security Considerations

6. Service Level Agreements

7. Service Directory

Recommended practices and considerations in each category are described below:

Service Monitoring	
Availability	<p>The active availability of a service is generally defined within its accompanying service level agreement (SLA). Common availability considerations and guarantees include:</p> <ul style="list-style-type: none"> • When will the service be active and available? (This may be expressed in a formal schedule or timetable.) • Will all dependencies for this service also be available during the service's scheduled availability? (If not, certain service capabilities may not be fully available, even if the service is active.)
Logging	<p>The supporting service infrastructure should store (log) data about each service's access and usage patterns. This data is useful for troubleshooting, monitoring, and security control. Formal review processes may need to be in place to ensure that logs are routinely checked. This is especially important to raise awareness of certain usage trends that may indicate a need for infrastructure changes (usually associated with scalability and security measures).</p>
Auditing	<p>Auditing is the analysis and reporting of logged information. The data collected at this stage should answer questions such as:</p> <ul style="list-style-type: none"> • How exactly is a service being used? • Who is using the service? • What do the common request and response messages look like? (In other words, what are the most typical content exchange scenarios.) <p>In addition to better understanding how a service is being used, these reports will also highlight the capabilities of the service <i>not</i> being utilized.</p>
Performance Metrics	<p>Performance metrics and statistics for each Web service are kept so that services can be monitored to avoid potential runtime issues, such as performance bottlenecks and fault counts. It is frequently required (and recommended) that automated notification mechanisms be attached to collected statistics so that action can be taken well before an infrastructure's limitations are tested.</p>
Debugging & Tracing	<p>The ability to optimize and track service activity during development and after deployment is important, especially for maintenance purposes. This focuses more on the service hosting platform's ability to provide front-end tools that allow for targeted investigation of specific activities (or sub-activities) as opposed to common general reporting features.</p>
Synthetic Transactions	<p>The ability to utilize of synthetic (dummy) transactions is helpful to simulate service usage scenarios during development, testing, and debugging phases. Synthetic transaction modules can be configured to send periodic service requests according to pre-defined settings.</p>

Exception Management

Error Trapping	Runtime errors needs to be recorded and reported as efficiently as possible. For example, the recorded response times and any logged timeouts need to be monitored to ensure that the service is kept inline with corresponding SLA requirements.
Root Cause Analysis	It has become widely accepted that SOAP faults are not that useful. An SOA infrastructure needs to be able to drill down into the real reasons as to why a service is failing. Often this reveals issues such as code faults and hardware failure. With the right tools, a true root cause analysis (one that goes beyond just the vendor runtime) can be completed.
Notification Services	A services infrastructure should be able to send out notifications when specific events are triggered. This notification mechanism needs to be configurable so that it can apply to individuals and broadcast groups.

Version Management

Data Contracts	Web services exchange data using pre-defined data models based on XML Schema. Changes to a published schema can therefore break the data contract and jeopardize all existing service consumers. If data contracts do need to be changed, a separate version control practice may need to be in place. This is primarily in consideration of the fact that an XML data representation architecture will often need to be maintained and evolved separately from the service layers that utilize its schemas.
Message & Operation Contracts	Service consumers are tightly coupled to the service provider through the service interface. However, extending a service contract without affecting existing service consumers may still be possible. A formal process is recommended even when just adding operations to an established WSDL definition.
Endpoints (Addresses)	The service consumer is tied to the service endpoint (address). If the address of the service changes all active service consumers will be compromised. Therefore, a separate endpoint or address management practice may be required.
Policies	Given the absence of an industry-wide policy expression language (cross-vendor implementations of WS-Policy still seem a distant reality), documenting and attaching policies to the technical service contract in a standardized manner can prove difficult. A common approach is to define policies as part of the accompanying SLA. Though the policies may be required, the SLA can often just request that service consumers adhere to policy rules. The enforcement of this practice sometimes requires that the service consumer owner sign a contract (or the SLA document) guaranteeing that all policies will be honored.
Internal Dependencies	A service may be implemented using numerous proprietary components, databases, and other system resources. Changes to these underlying dependencies can raise all kinds of runtime exceptions. Therefore, internal practices may be required to ensure the integrity of each individual service-level technology architecture.
Claims	The service provider may be secured using a common identity technology, such as active directory roles and users. Any security changes in the infrastructure can also impact all consumers of a service. Changes to established security management processes and practices therefore become important considerations and may require formal reviews by committees.

Service Retirement	Services do not live forever. When services are deprecated, service consumer owners must be notified in time to adapt their environments. Often this may require a graceful expiry period during which both old and new versions of a service contract co-exist for a pre-determined amount of time (sometimes this period can extend up to six months).
Dependency Analysis	Service consumers can be classified as "known" and "unknown" consumers. Keeping a list of all the consumers associated with each service can simplify some versioning issues by allowing for organized notification.

Service Delivery

Methodology	A methodology practice (such as Erl's mainstream SOA methodology [REF-1]) defines how a service is moved through lifecycle stages and also establishes any new phases required specifically in support of SOA. In addition to shaping service logic in preparation for implementation, these formal approaches also need to provide governance processes in support of post-implementation management and evolution.
Standardized Service Delivery Lifecycles	Requirements gathering, business analysis, service modeling, design, and other delivery lifecycle phases need to be aligned with the service-orientation paradigm. This does not necessarily supersede object-oriented approaches, but the service implementation (and especially its contract) must reflect the design characteristics required to achieve the strategic goals behind service-oriented computing.

Policy and Security Considerations

Identity Store	An identity store is a repository of users, their credentials and preferences. In order to access a secure service the service consumer must be validated against these identities. Various options exist for implementing an identity store including Active Directory or even a regular database. Ideally, an identity store is a standardized and centralized part of the overall infrastructure. Practices need to be in place to ensure it is consistently used and maintained.
Authentication & Authorization	The ability to verify the identity and permissions of service consumers raises a number of considerations, including how security claims from service consumers are validated and processed.
Exchange Policy & Contracts	This practice should address how policies and contracts are exchanged between a service provider, its consumers, and the owners of its consumers.
Internet Perimeter Security	The issues associated with securing Internet firewalls and Internet facing servers and ports are amplified when moving toward a technology environment consisting of a combination of internal and external Web services. Practices need to be in place to ensure that acceptable measures of security are always maintained, but also to provide a level of adaptability in response to unforeseen external access requirements that may be raised by newly published services.
Usage Control & Metering	Understanding the usage of a service may be important for a number of reasons. For example, we may need to bill the service consumer on a per usage basis or we may want to study service usage and load patterns.

Transport Security	Here we deal with how Web services traffic is secured on the wire. Of course this usually involves the positioning of SSL, but several WS-* specifications can also enter the discussion.
Load Balancing	Services with heavy or unpredictable volume loads often need to be dynamically load balanced across multiple servers. Various types of load balancing algorithms exist and services should be assessed individually to ensure that the most appropriate type of load balancing is always chosen. Sometimes, based on increased reuse or recomposition, services need to have their load balancing algorithms "upgraded" in response to new usage demands.
Geo Clustering	This consideration addresses issues that can arise from the physical proximity of clustered services within an enterprise. For example, service consumers may need to bind to services that are geographically closest to minimize server hops and latency issues.
Web Services Interoperability	While Web services are designed to interoperate, enforcing WS-I Basic Profiles is often the only way to guarantee interoperability, especially across disparate platforms and organizational boundaries. When building services with some of the new WS-* technologies and especially within vendor-diverse enterprises, extensions or custom variations of the WS-I Basic Profiles may need to be created (although their usage will likely be limited to internal, controlled environments).

Service Level Agreements

Quality	A dedicated practice is often needed to ensure that a service provider continually meets the quality contract promised to its consumers. Carrying out this practice may require a separate "quality assurator" and "quality broker" role.
Billing	Whether a service is consumed within an enterprise or outside of the organization, a billing structure is often required to guarantee that the service owner (a department or the organization itself) is compensated. Usually, billing is based on a licensing agreement that ties into a per consumer or even a per usage payment model.
Configuration Management	As each service is promoted through various environments it needs to be configured and tested. As the number of services increase, managing the overall configuration management effort can be complex and will likely require formal processes. Configuration management of services can be much more challenging than traditional efforts. Therefore, new practices will likely need to be developed and configuration management implications may need to be expressed in the SLA.

Service Directory

Awareness & Discovery	The service consumer must have knowledge of the existence of the service provider and the location of its contract. This awareness increases the overall reuse potential within an enterprise. UDDI exposes the required meta information in an industry standard manner. (Note that awareness alone does not enable a service consumer to invoke a service.)
Publish Process	Once a Web service is implemented, it will need to be registered in a local service directory or registry. Formal processes need to be in place to ensure that the registration and the documentation of associated metadata are performed in a consistent manner.

Subscription	It is ideal to allow service consumer owners to subscribe to service directory records. If changes to the service contract are added or should there be outages or other events associated with the implemented service (such as a change to billing rates), consumer program owners can then be easily notified.
Service Owner Contact Information	Service consumer owners or designers of potential service consumer programs should have the ability to contact the owner or custodian of a service for questions and recommendations. Especially in larger organizations, it is best if this contact information is published alongside the service directory records.
Documentation	The service registry should contain documentation to help owners of potential service consumers better interpret the service's capabilities. This information may exist in separately published documents or as annotations to the technical service contracts.
Rating	It can be beneficial for quality assurance purposes to allow service consumer owners to rate services and provide feedback. Typically, ratings are in response to guarantees and promises made in an SLA as measured against a service's actual runtime performance.

The SOA Implementation Checklist Template

There is no fixed order in which all of the previously listed practices need to be carried out. Each organization will have its own preferred sequence based on how their transition or migration plans are structured. To provide some guidance for going through these individual practices, a sample Excel template is provided. You can use this as a checklist to ensure that all considerations are eventually taken into account: [SOA Implementation Template.xls](#)

This template provides the following supplementary columns:

1. Phases

Allocate a practice to a phase based on your organization's priorities and risks. By default, four phases are represented in the template. Some practices may intuitively fall into earlier phases. For example, it is often desirable to implement some or all practices related to security as early in the lifecycle as possible.

2. Products

There are two aspects to introducing any practice, namely technology and process. The technology factor may be either a product or some custom development software. Process, on the other hand, require internalizing the technology so that it is used consistently and correctly. The process is represented by the practice. This column simply provides a means of referencing the technology associated with either applying or carrying out the practice.

3. Custom Development

Some practices can be better implemented through custom development efforts. For example, authenticating a service consumer against an existing membership repository will almost always require some extent of custom programming.

Conclusion

As part of our exploration of SOA roadmap planning we've accumulated a list of 37 recommended practices and considerations. When putting together a serious SOA transition plan, you will always come up with additional items for this list. Every organization has unique requirements that relate to the distinct nature of their existing technical environments. Many of these unique requirements will originate from the legacy systems that already comprise a fixed part of the infrastructure and impose very specific constraints on the SOA project plan. Either way, starting the process of charting your SOA roadmap with a simple checklist, such as the one provided here, will ensure that you address fundamental considerations, which is the first step to establishing formal practices.

References

[REF-1] "Service-Oriented Architecture: Concepts, Technology, and Design", Thomas Erl; Prentice Hall, 2005; ISBN-0-13-185858-0, available at <http://www.soabooks.com>

THE PRENTICE HALL SERVICE-ORIENTED COMPUTING SERIES FROM THOMAS ERL



[home page](#) [past issues](#) [official book site](#) [legal disclaimer](#)

Copyright © 2006 SOA Systems Inc. All Rights Reserved