

The SOA Magazine

Feature Article



Essential Components of an SOA Quality Foundation

by Jim Murphy

Published: March 1, 2007 (SOA Magazine Issue V: March 2007, Copyright © 2007)

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

Abstract: There are a significant number of companies investing in SOA initiatives, but only a small number are at the point of realizing a positive return on their investments. These organizations have embraced SOA as part of their core business strategies, yet they have also taken a pragmatic approach, knowing that a successful service-oriented architecture depends upon the quality of the architecture itself.

Creating an SOA quality foundation means understanding the overall goals of a service-oriented strategy, and optimizing the environment for successful execution of that strategy. This article explores the major components that make up the foundation of SOA quality and discusses the need for communication, trust and control to ensure that a service-oriented solution meets or exceeds business and technical expectations – whether it's in the form of cost savings, productivity, better customer service, or reduced time to market.

Introduction: The SOA Quality Foundation

SOA quality is measure of how an SOA implementation accommodates business and technical expectations. To meet these increasingly high expectations, SOA must consistently yield value in the form of cost savings, productivity, and time to-market. In fact, "SOA quality is fast becoming one of the next critical issues that enterprises, consulting firms, and vendors alike must face when implementing SOA" [REF-1]. But, the question remains: How does a company go about achieving SOA quality?

The key is building a proper foundation for success. And, to accomplish this requires a proper plan. A recent survey confirmed that organizations with a quality plan are more likely to be satisfied with their SOA deployment, and that "creating a viable SOA initiative is contingent on the ability to ensure software quality and gain trust of all constituents. Without an articulated quality strategy rigorously enforced SOA will not scale and will not deliver on its promises" [REF-2].



Figure 1: The common building blocks of an SOA foundation framework.

Achieving business objectives from an SOA requires both a top-down approach to mapping out those objectives and a bottom-up strategy for incorporating SOA quality. Figure 1 illustrates the common building blocks of an SOA foundation. Let's take a closer look at the five fundamental components that comprise the bottom layer.

Prototyping

As business and IT groups strive to work together on SOA initiatives, prototyping is one of the most effective ways of mapping out collections of services prior to actually investing in their development. Top-down projects typically start out this phase with up-front analysis efforts. It is here that service modeling processes are carried out in order to produce a service inventory blueprint, which is essentially "a conceptual blueprint of all the planned services for a given inventory" [REF-3]. The service candidates produced by the service modeling phase provide a starting point for service design processes that end up producing service contracts before any actual code is written.

By iterating through service-oriented design, multiple contracts can be developed and refined, allowing business analysts, architects, and developers to work with and agree upon concrete technical interfaces early in the process. This helps tune individual services in support of reusability and also allows consumers and testers to get involved in the analysis and design processes, reducing the overall burden on the eventual development cycle.

Prototyping with Web services involves the use of WSDL definitions to build a simulation of a service-oriented application. While a WSDL contract is static, a simulation is dynamic, providing a better understanding of how services will behave and interact. The prototype facilitates early testing for non-functional issues of compliance and interoperability, as well as functional and architectural issues such as interface granularity and data mapping.

If Web service contracts are available to and easily understood by business analysts and architects, they can begin the testing and quality process early on in the project cycle to identify potential problems and reduce the overall time to deployment. When prototyping is carried out by those working on the business side, it enables the notoriously elusive alignment of business and IT so that new services are created in full support of business goals.

Compliance

Fundamental to achieving a meaningful level of SOA quality is a consistent compliance to standards. Non-compliant services pose a high risk to attaining positive business returns and simply cannot exist if an SOA is to be successful. Even well-written services cannot guarantee broad interoperability unless standards and best practices are adopted and adhered to throughout an organization.

Industry standards, such as the WS-I Basic Profile, provide fundamental guidelines for interoperability. However, corporate standards, particularly for XML Schema, are also required. Once in place, both industry and corporate standards need to be enforced. Therefore, it is critical to incorporate compliance early on by providing the right tools to those creating the WSDL definition and associated schemas. This positions standardization as a core characteristic of each service (as per the Standardize Service Contract principle [REF-3]). Conversely, governance solutions that “police” compliance late in development and deployment processes can make conformance a challenge, especially when carried out with burdensome tools.

One example of how compliance and quality are being applied early in the delivery lifecycle is the aforementioned contract-first approach in which the customized service interface drives the development process. Contract-first design can reduce initial development cycles while also supporting ongoing compliance over time.

Current Web service tools make contract-first development possible, but many far from encourage the practice. Developers will only embrace standards, rules, and policies by seeing value in what they provide. Architects and governance teams must educate others as to what the standards are, why they are in place, and most importantly, how developers can improve their projects to become compliant.

Testing

The design considerations raised by service-orientation require changes to how software testing processes are carried out. Traditional approaches emphasized the testing of business logic through application user interfaces, which has proved to be critical when deploying new solutions. With SOA, the need to test the business logic still exists (in fact, it is even more important with disparate services). However, Web services lack a user interface, making it very challenging for QA teams to ensure that solutions comprised of shared services can properly support business requirements.

SOA project teams therefore need tools that can provide the necessary user interface, simulate unavailable services, and ensure that all team members (including those without programming or XML knowledge) can test services. It is also important that test scripts be accessible to allow for continuous regression testing, a form of testing particularly relevant service policies change or when consumers need to use services in different ways.

An implemented SOA has many moving parts. It is virtually impossible to test every Web service and its interaction with its dependencies. Additionally, unlike traditional software applications, testing occurs while many services are in various stages of development (including production). New testing methods are needed in order to fulfill the unique aspects of a service-oriented solution, because “testing of SOA environments is taking a quantum leap in complexity... the trust needed for SOA to attain widespread use will be built from the successful use of well-designed and tested services” [REF-2].

Diagnostics

Determining if a Web service can perform its intended function is often a time-sensitive issue that might require fast identification of a root problem. This can pose significant challenges to many teams and individuals because Web services in an SOA tend to be more complex. No matter how well Web services and business processes are tested, problems will still occur; after all, they are still software and prone to the same logic defects as ever. Within a service-oriented enterprise, problems often need to be solved in real-time, and may further require the involvement of disparate teams and systems. These circumstances require a means of carrying out *collaborative diagnostics*.

As a core component in the foundation, strong diagnostics are essential to providing and maintaining SOA quality during runtime, and preventing runtime issues by catching them at design time (and “change time”). Within a service-oriented environment diagnostics are especially critical at the XML abstraction layer. It is here where we need to pinpoint the nuances of message exchanges that can hinder the performance and trust across Web service activities.

Specialized diagnostic tools are also needed to test areas that systems management tools and code analyzers tend to

miss. It is important that these tools eliminate the complexities of the XML language but still provide rapid problem identification and resolution capabilities. Furthermore, each member of a project team should have the ability to diagnose problems, which means that usage of diagnostic tools cannot be limited to technical professionals only.

Support

Reproducing problems within the more complex service-oriented solutions can be a significant challenge. Support departments will often need to involve several groups and individuals to solve a given problem, many of which may not have the necessary technical expertise to understand the problem in the first place. This can lead to confusion and unproductive collaborations that can further result in finger-pointing and blame. These situations can be avoided when there is a common understanding of the problems and an effective means by which any project team member can reproduce them.

Also, support is fundamentally different in an SOA because it involves two phases that span design time and run time simultaneously. As services are exposed for use, consumers will require support to assist in the development of their applications. Automated “pre-support” or “consumer jumpstarts” involve exposing documented contracts and providing a way for team members to investigate services and try different scenarios.

In production, support teams need to understand and resolve problems quickly and often need to reproduce scenarios as failure occur. When service developers get involved, complete problem data needs to be shared with other team members with the ability to simulate different scenarios to more effectively diagnose these problems. It is absolutely essential to have a mechanism for supporting the disparate groups that use services, without overwhelming the development teams.

Pre-support and run-time support creates another primary building block in the foundation of SOA quality – one that requires solutions that can simplify and automate the entire support process. This means providing support teams with the ability to effectively collaborate with others, document accepted processes and scenarios, and easily capture, reproduce, and resolve problems.

Communication, Trust, and Control

The ability to communicate effectively is an essential part of a successful SOA initiative. It is equally important to service-oriented technologies as it is to the people involved with the environment “...because SOA reflects an extensive ecosystem [and] successful SOA implementations are by nature collaborative” [REF-2].

However, communication is more than just interoperability and collaboration – it involves the interaction between people and technologies, which is often where quality breaks down. In addition, trust is what drives service reuse and is therefore relevant in nearly every aspect of an SOA. As advocated by the Service Discoverability principle [REF-3], it is critical for teams to be aware of existing services *and* understand what they can and cannot do, and most importantly, have trust and confidence that the services will execute as intended.

Lastly, control is essential at design time, change time and run time. It involves policy enforcement to ensure quality, and requires changes in human behavior, development concepts, and process. It involves reducing the number of production issues during run time and resolving those issues quickly. By factoring in the need for effective communication, trust and control, teams will be better able to work in a cohesive, effective manner throughout the entire project lifecycle.

SOA Quality Management

There are numerous tasks that can be part of a quality management approach for service-oriented solutions and service inventories. Figure 2 highlights some of the key parts of a common SOA quality management framework.



Figure 2: Common tasks and processes required to effectively manage SOA quality.

Visualization

When evolving an SOA, developers need a design-time view of services within a registry. More than just a list of services, visualization requires access to service documentation, the ability to try out (test-drive) the service without the need to program or know XML, and an understanding of compliance status. For non-developers (architects, QA, etc.), visualization provides the ability to read and understand a Web service without deep knowledge of XML.

Exploratory Testing

Providing developers and QA with the ability to perform ad-hoc tests, such as point-and-click service invocation and the ability to leverage previous tests to perform “what if” analysis, are all considered part of exploratory testing.

Service Simulation

Simulating a service based on its WSDL interface is essential to prevent developing and testing applications against production endpoints or when a service has not been constructed yet. This is particularly an issue when there is no convenient staging instance for all services involved – such as a mainframe system that offers a Web service interface.

Scenario Testing

Once a critical mass of services is available, orchestrating them together into business oriented scenarios is possible. Verifying the correctness of these scenarios is a great way tie business value to the quality process.

This includes:

- recording Invoke/Resend interactions to form a scenario or build from captured message traffic
- thread data between scenario actions
- set “expectations” for automated response validation
- running a test and then retargeting endpoint URLs to test different locations

By continuously assuring the quality of business processes, scenario testing can help organizations realize the true potential of their applications.

SOAP Message Validation

High-value services often have complex input and output message formats. To ensure services are behaving properly you need tools to verify that service response conform to the service contract as described in the WSDL. SOAP message and schema validation is a good place to start but might be just the beginning if your message formats require complex semantic validation.

Impact Analysis

When iterating internal or external changes to a service implementation or interface you need to understand the impact those changes will have on existing Web service consumers. Even when a change does not affect the service contract, the degree of semantic change is unknown unless you can exercise the service with messages and scenarios that establish a known baseline.

Regression Testing

Playing back a series of tests to compare the current version/iteration of a service with a previous version can reveal performance and behavioral differences. Regression testing is critical to SOA, because service-oriented solutions are comprised of service compositions which, in turn, are comprised of many shared services.

Functional Testing

In order to determine that a Web service functions as expected, we need the basic ability to send and receive messages. Yet this simple test may be a challenge to QA teams trained only to test user-interfaces and Web pages.

WSDL/XSD Compliance

A critical component to ensuring Web service and SOA quality is the ability to check any WSDL contract or message against industry standards. Ideally, testing should also include compliance with corporate standards.

Architectural Reviews

A common best practice for building service-oriented solutions is the previously described contract-first (or “contract early”) approach. Once an architect has a WSDL contract, having the ability to review and compare it to other services helps ensure interoperability and, ultimately, SOA quality.

It Starts with Planning

Every successful strategy is based on a plan and a team to execute that plan. To begin building a foundation for SOA quality, organizations can start with these three basic steps:

Assemble your Team

Today’s successful SOA implementations are commonly led by a cross-functional team representing business

analysts, architects, development, support and IT. These teams are often referred to as an "SOA Council" or "SOA Center of Excellence." The intent of this group is to ensure that business goals are met through the adherence to standards and best practices, the management of governance initiatives, and the effective collaboration of all affected departments and teams.

Top-Down and Bottom-Up

It is critical to begin by outlining business objectives that your SOA will aim to fulfill. But companies solely focused on ROI can be destined to fail. Excessive investments are made and rash decisions may occur when the patience of business teams begins to wear thin. A balanced approach to building a core set of services and composite applications with trained teams and short term goals will prove highly beneficial in the long run, and also yield positive short term results in productivity and agility.

Build Schools, not Prisons

SOA is a new concept and will require dramatic changes in your IT infrastructure, your application development processes, and even skills and roles of your team. The early stages of an SOA initiative are critical to promoting acceptance, adoption and trust among everyone involved. If you can simplify the concepts and complexities, the process of "service-orienting" will be much easier and more successful.

Strict enforcement and rejection of non-compliant services without identifying the conformance issues can lead to dissention and redundant or rogue service development. Therefore, teaching and empowering architects and developers concepts such as "contract-first" development will instill quality initiatives at the outset. Education, communication, training, and collaboration are essential to laying the groundwork for future success.

Conclusion

Although the ultimate goal of any SOA is to achieve business objectives, SOA quality is required to ensure that service-oriented solutions meet or exceed business and technical expectations - whether in the form of cost savings, productivity, better customer service, or reduced time to market. Pervasive quality enables organizations to achieve significant financial gains from agility and reuse in the forms of improved IT infrastructure management, reduction or better use of existing resources, and adoption of efficient methodologies.

SOA quality does not exist alone in any single part of a system and is not the sole responsibility of any one individual or team. It is an integral part of any SOA methodology that must be embraced throughout the enterprise from the conceptualization to the governance of service inventories and architectures.

References

[REF-1] "SOA Consulting: Current Market Trends", ZapThink, September 2006.

[REF-2] "Executive Survey: SOA Implementation Satisfaction", Hurwitz & Associates, November 2006.

[REF-3] "SOA: Principles of Service Design", Thomas Erl, Prentice Hall/PearsonPTR, 2007.

THE PRENTICE HALL SERVICE-ORIENTED COMPUTING SERIES FROM THOMAS ERL

