Testing a SOA Application

Srikanth Inaganti and Sriram Aravamudan

Abstract

Conventional software testing methodologies often do not provide an efficient and accurate test plan for SOA. Since SOA is a loosely coupled, distributed architecture based on reuse, it presents many potential sources of failure. This necessitates a more robust test methodology that covers several new areas, such as reuse, agility, vulnerability; composability testing, etc., that are unique to SOA.

Services specifically designed to function as independent entities may work perfectly well in isolation, but might not function as expected when integrated into applications [1] due to underlying platform and network specific issues. Secondly, when a service has potential for reuse, testers need to ascertain whether the service can really scale up to the reuse expectations and contribute to agility. Also, ensuring agility requires testing business rules, configuration parameters and policies, etc. Services also need to be tested for the data translation and information delivery for different consumers. Covering all these test areas for SOA is not possible only with the support of testing tools. A complete SOA test coverage would in fact involve extensive design reviews targeted at reuse, agility measurement with the collaboration of domain specialists as well.

The hallmark of quality SOA is the seamless modeling of change requirements into policies and service contract adjustments by business analysts and architects in order to achieve enterprise agility. Because of the above factors cited above, SOA quality assurance needs special attention during enterprise SOA transformation to achieve agility by ensuring adherence to SOA practices and principles,.

This paper discusses an overall SOA testing methodology and additional test areas specific to SOA. It also highlights the importance of an SOA tester, who needs to be knowledgeable not just of testing tools, but also of some amount of SOA philosophy/domain knowledge to be a genuine value-add to the SOA testing process.

SOA Test Methodology – The big picture

An enterprise SOA program usually has multiple projects involving applications and services developed in parallel. Ideally, service development should lead composite application development by at least one activity or cycle for smooth integration and testing of applications and services. Careful planning is required to ensure that all dependencies are accounted for between services and composite application development. This means that before unit testing the (composite) application, the services required for it should have already been unit tested. Services should have also undergone additional tests for reusability, agility, inter-operability and security before they are consumed by composite applications.

Note that SOA transformation involves iterative development of services and applications and hence involves testing in all iterations with automated regression testing as shown in figure 1. GREEN colored circles represent composite application testing with interspersed service test areas represented in BLUE colored circles.

Traditional systems would undergo *functional testing* comprising unit, integration, system, acceptance, regression testing and *performance testing* comprising stress, fail-over, break and endurance testing. SOA functional testing focuses on testing the functionality for the composite application that is built using sub-systems or modules that are independent of the rest of the IT eco-system. It also involves testing services that are woven into the application logic with dialog control handled by either BPEL engines or non-BPEL implementations. The concept of using and

composing reusable services to construct new applications thus brings the following additional test areas into overall SOA testing as depicted in the diagram shown below.

1. **Service Agility Testing**
   - Configurations
   - Business Rules
   - Policies

2. **BPEL Level-1 Testing**
   - Compensating transactions
   - Service unavailability impact

3. **BPEL Level-2 Testing**
   - Workflow Testing
   - Events Testing

4. **Security Testing**
   - Denial-of-Service (DOS)
   - Vulnerability
   - Context propagation/Federation

5. **Service Design Testing**
   - Interoperability Testing
   - Service-App Integration Testing
   - Reuse Testing (Consumability and Composability)
   - Service Data Testing

6. **SOA Performance Testing**

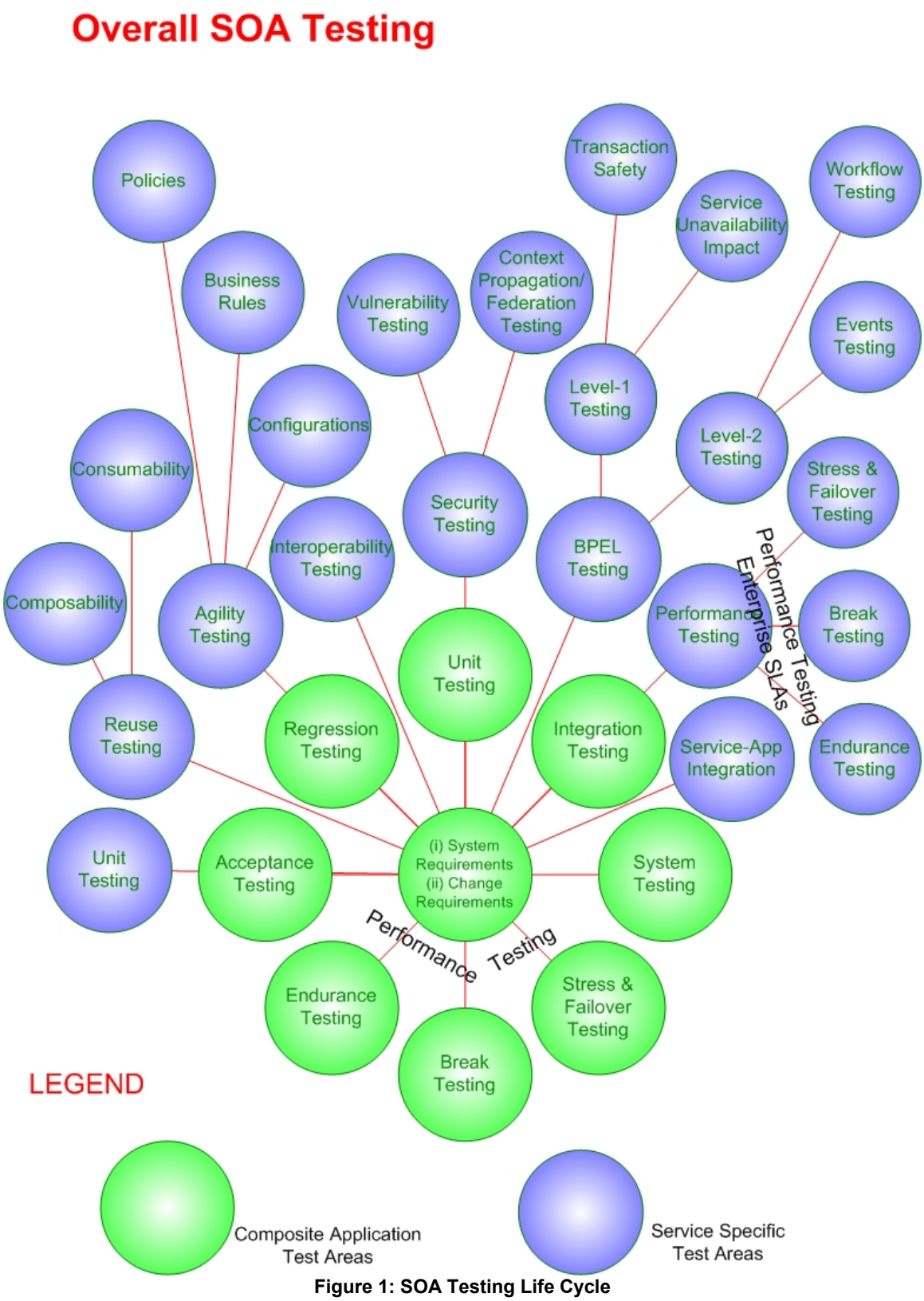

Figure 1: SOA Testing Life Cycle

SOA Test Areas

SOA Testing and quality assurance, ideally, should be carried out by experts with considerable experience. Broad experience is needed because SOA testing involves not just functionality checks, but also the reuse and agility checks that require support from domain specialists, architects and designers in terms of design reviews.

SOA testing however has seen lot of demand in the past couple of years, and the trend promises to continue in the years ahead [5]. Because of the increased demand on testing resources, there has been considerable impact on the testing resource standards in some instances. Thus, design reviews have become even more critical to SOA quality assurance.

While the functional testing can be done fairly easily by a moderately experienced SOA tester, using tools available in the market, the reuse and agility testing requires thorough assessment by the business analyst, SOA architect and the information architect. The business analyst would bring domain knowledge to the proceses, the SOA architect all the design and architectural decisions made, and the information architect would contribute knowledge on utilization of all data entities involved across the business processes.

Reuse and agility testing obviously require special attention as these are the main areas in which the enterprise leadership expects the maximum return on investment. Effective design reviews with special attention to reuse are a key factor in ensuring achievement of this goal..

The significant investment of time and financial and personnel resources required to realize the benefits of SOA transformation is often an impediment to management's wholehearted endorsement of a SOA initiative.. Hence several IT initiatives are being sprinkled with SOA with either no long term strategy [3] or without a major one-time investment. In either of these circumstances, in addition to institutionalizing effective governance practices to define and roll out necessary IT processes [4], strengthening quality assurance processes within micro SOA efforts would make overall SOA transformation a success. This would ensure that a piece of code or logic marked as a service is actually reusable in different contexts and contributes to required agility.

The following subsections describe various key aspects of SOA testing.

Service Agility Testing

This procedure requires complete understanding of the requirements for which the service is designed. The review should focus on identifying the volatile parts of the requirements, since agility is all about externalizing them into parameters via configurations, rules and policies [6]. All of these parameters would change the behavior of a service or component in different contexts. Configurable items influence system functions, rules would influence business functions, and policies would influence security, monitoring, performance etc. of services. The reviewer should make sure that all the volatility is captured in either configurations or rules or policies

This essentially reinforces the fact that most of the iterative development models like TDD, SCRUM, XP etc are pushing the design and domain specialists to be more involved in testing and quality assessment activities[8].

Another important factor in agility testing is to measure how quickly service can be provisioned to accept a new consumer. Please refer to interface, implementation, data and packaging considerations of service [6] for design guidelines.

The other essential factor that contributes to enterprise agility is how semantic integration has been designed into SOA. One of the measures is to define certain test cases for data integration by changing some data entity semantics and changing the appropriate map to see how quickly the systems in question come back to normal operations.

BPEL Testing

In the case of composite services, two levels of testing are required (i) Orchestration (BPEL) testing and (ii) Process level testing.

Level-1 Testing is aimed at making sure that the sequence of service calls made to various systems will not leave any of the systems in an inconsistent state. This test also helps in understanding the impact of one or more services in the chain not being available. Note that the more a service gets reused, the greater the impact associated with its downtime will be [7].

Services are composed to form the business processes either partially or completely by orchestrating them in a defined sequence. These services may be hosted on different platforms with different underlying transactional engines. In the event of failure of any service invocation in the sequence, BPEL provides compensating transactions (service methods) to maintain the data in a clean state. All the compensating transactions have to be tested thoroughly with exhaustive test cases and data. It is essential to test the level of impact in case of service unavailability to the business process as a whole, and to check the remedial mechanisms provided and their effectiveness.

Level-2 Testing is aimed at testing the flexibility in the workflow design provided and also the effectiveness of process visibility details captured via events.

Since SOA involves a lot of asynchronous communication across systems, it is possible that messages will not reach their destinations in an orderly fashion because of delays in processing at end-points (target systems). Some systems will respond quickly, and some slowly, depending on the load at any given point in time. Response messages must to be correlated centrally before they are dispatched to the target systems. Changes in workflow may result in changes in the delivery order of the messages from one system to another. Test cases should be aimed at testing this scenario to measure how quickly a workflow can be changed. The results are in fact a reflection on the extent of agility built in to the enterprise's SOA.

Testing the above scenarios requires thorough involvement of the development team during the testing process.

Security Testing

Typically, major SOA initiatives involve exposing services to and consuming services from the outside world. This fact opens up a host of vulnerabilities, such as DOS (denial-of-service) attacks, penetration, high volumes of spam data, etc. Typical security policies have to be enforced at the network level, middleware level and at the end-point level to create bullet proof SOA. Specific test cases aimed at targeting these policies need to be designed to fully test SOA security.

Service Design Testing

Reuse Testing*:* Knowledge of design decisions made and the utilization of data entities within various business processes would be validated by the domain knowledge of the business analyst to make certain that the interface or contract would work in all possible business scenarios, thus ensuring consumability and composability.

Interoperability Testing: Since enterprise SOA development involves different technology platforms and development tools at the enterprise level, it is quite possible that developers use many different tools to generate the contracts (WSDL) and modify them in the process. Also, different runtime SOAP engines would interpret contracts differently, based on their implementations. Hence it is essential to make sure that all (web) services deployed are compliant with WS-I basic profile. Tools such as SOAPScope, SOAPTest and tools from WS-I.org, etc. provide the necessary support in this testing process.

Service – Application Integration Testing: In a big SOA program, it is quite common that different services and applications are developed independently in parallel by multiple vendors in different geographical locations. Defining the contracts or interfaces at the beginning would help the development effort to go smoothly. But at the time of integration, it is useful to have an integrated test environment where services can be deployed, tested and integrated with applications for overall testing.

Data Level Testing: For services, data must be tested for completeness, correlation across different consumers, separation of data for consumers, element level security as needed and ownership assessment by testing support services such as reporting and data archival mechanisms, etc.

SOA Performance Testing

Performance testing is needed for both services and composite applications. Performance testing usually involves a lot of time and repetitive test cycles, and needs careful planning for dependencies.

Performance testing depends on the following enterprise level and application/service level activities:

Enterprise Level Activities:

1. Standardizing and procuring the software required
2. Standardizing and procuring the hardware required
3. Setting up the required test environment/lab
4. Setting up the shared dedicated service hosting zone

Application/Service (development) Level Activities:

The following few line items add some additional effort, time and cost, and need to be included in the overall planning of the SOA projects.

1. Scripting necessary for load generation
2. Preparing test data
3. Providing a sample UI to test the service (for continuous monitoring by tools like Sitescope etc.)

Optimizations in both service and application code are to be expected from the issues that emanate from performance testing.

Part 1: Services Testing

Services are typically intended to have high usage and high reuse opportunities. Considering this fact, their performance requirements are more stringent compared to composite applications. For example, a service has been identified as part of a micro SOA effort and has been given a performance SLA of serving 50 concurrent users with 8 sec of response time. The service factory team [4] would therefore need to target a performance requirement of at least 5(50)= 250 or 10(50)=500 concurrent user load with 8 sec response time.

Typically, services are server side components that do not have a front end. It is therefore useful to have a sample GUI interface that can be used to load test the service from a web client. But in the real world, services would be accessed by different types of clients and with different date formats. For testing the real world service, it is a good ideal to select a test tool that can emulate different clients with different types of data (good and bad) and volumes to check if the service is responding as expected. This measure would improve the productivity of the development team significantly.

Part 2: Composite Application Testing

When a composite application is being tested for load, stress, fail-over and endurance, it is important to observe the throughput of the mediation component or service bus to see if a bottleneck occurs. For SOA architects in the IT services business, it might be a challenge to convince the customer-side business and IT management to accept reasonable end-to-end performance SLAs. In the pre-SOA era, where tiered architectures mapped on to a web server, application server and a database in different configurations, the 8-sec rule was used as reference point. While the same rule can be applied to SOA applications, test cases need to ensure that the response time is maintained under various levels of stress.

Note that in a distributed architecture like SOA where services developed internally and externally are used to build composite applications,, keeping the bar at 8 sec might be a challenge in certain business scenarios.

SOA architects must consider virtualization and GRID computing as potential performance solutions. And also, it is extremely important for SOA architects to involve the operations engineering team for required capacity planning to get ground realities before committing to SLAs.

SOA Quality Assurance Processes

The following are the important testing activities that ensure SOA quality.

- Automate test case execution to improve productivity and to detect errors at the time they are introduced into the code by developers. Look for tools that facilitate static code analysis to detect standard code errors and enforce coding standards, and run-time analysis to detect resource leakages, potential bottlenecks, security breaches, etc.
- Ensure that real-world test data is used right from the outset [1]. This facilitates the detection of errors that creep in as newer architectural layers are added.
- Automate regression testing to account for short cycles of iterative development. This helps to manage the whole testing process.
- Maintain continuous regression testing through nightly batch process.
- Develop a system for dash board reporting of errors to the entire SOA team
- Maintain regular quality assurance audits to ensure that major scenarios catering to business processes are run without errors. This needs involvement of SOA architect, business analyst and executives.

SOA governance must institutionalize the processes to ensure that design reviews are targeted towards reuse and agility testing against the requirements. These are in addition to checkpoints proposed in [4].

SOA Test Roadmap

In the previous sections, we have differentiated the various aspects of SOA testing from conventional software testing. Putting it all together is another significant task in which timing is key to ensure minimum redundancy and maximum test efficiency.

The following indicative SOA Test Roadmap gives a bird's eye view of the overall sequence of activities during SOA Testing. From figure 2 below, it can be seen that Service level testing is iterative in nature and occurs throughout the duration of the project. Other test activities, such as Governance Audits, performance, reuse, agility and security testing are continuous activities that take place throughout the duration of the project.

Figure 2: Indicative Roadmap for SOA Testing

Conclusion

In this article, we examined the additional test areas for SOA and the need for a specialized testing methodology. The key to SOA quality is to involve business analysts, executives, and architects as an integral part of regular quality audits, in addition to test engineers as specified in this document to ensure reuse and required agility at the enterprise level.

References

1. Testing SOA: Peeling the Onion - Linda Hayes
2. SOA Test methodology - Torry Harris Business Solutions
3. The Seven Deadly Beliefs That Could Hurt SOA Efforts by Jim Sinur, A Business Process Trends Column, Jan 2008
4. Reuse Framework for SOA by Srikanth Inaganti, Business Process Trends Journal, June 2007.
5. Zapthink's 2008 SOA Forecast
6. Service Design Essentials by Srikanth Inaganti & Srini Chintala, Business Process Trends Journal, Feb 2008.
7. Trusting Reusable Business Components in SOA and Web Services by Para Soft
8. SOA Quality Assurance - http://www.ebizq.net/topics/soa/features/8347.html
9. Governance, Quality and Management by Jason Bloomberg, Zapthink at SOA India 2007 Conference

Glossary of Terms

Acronym/Abbreviation	Definition
DOS	Denial Of Service
SOA	Service Oriented Architecture
SLA	Service Level Agreement
SME	Subject Matter Experts
BPEL	Business Process Execution Language
WSDL	Web Services Description Language
UI	User Interface
TDD	Test Driven Development
SCRUM	????
XP	Extreme Programming

Acknowledgements

We would like to thank Dr. Udaya Bhaskar Vemulapati, General Manager, Wipro Technologies for giving us the required time and support in many ways in bringing this article as part of SOA Center Of Excellence efforts. We would also like to thank our colleague Srinivasa Rao Chintala for providing review comments.

Authors

Srikanth Inaganti is a Senior Enterprise Architect in the Enterprise Consulting and Architecture Practice division of Wipro Technologies. Srikanth Inaganti, PMP, E-Architect, ECAP Group, Wipro Technologies, srikanth.inaganti@wipro.com, Office # 91 40 3079 5110; Mobile # 91 9849058064

Sriram Aravamudan is an Enterprise Architect in the Enterprise Consulting and Architecture Practice of Wipro Technologies. He can be contacted at sriram.aravamudan@wipro.com. Office # 91 80 41365602. Mobile # 91 9845089962

