



Technical White Paper

Testing of SOA Applications

Prakash D.,
IT Consultant,
Sonata Software Limited

STATEMENT OF CONFIDENTIALITY

Information included in this document, in its entirety, is considered both confidential and proprietary to Sonata Software and may not be copied or disclosed to any other party without its prior written consent.

Abstract

This White Paper discusses the reasons for increasing popularity of Service-Oriented Architecture (SOA) applications and the need to test them. It also explains the methodologies and processes that must be used to test SOA applications.

About the Author

Prakash D. is working as an IT Consultant with Sonata Software. With Sonata for about four years, he has an overall experience of over six years in the area of Software Testing.

Prakash handles the Quality Team for a financial application of one of the leading banks based in the Nordic region.

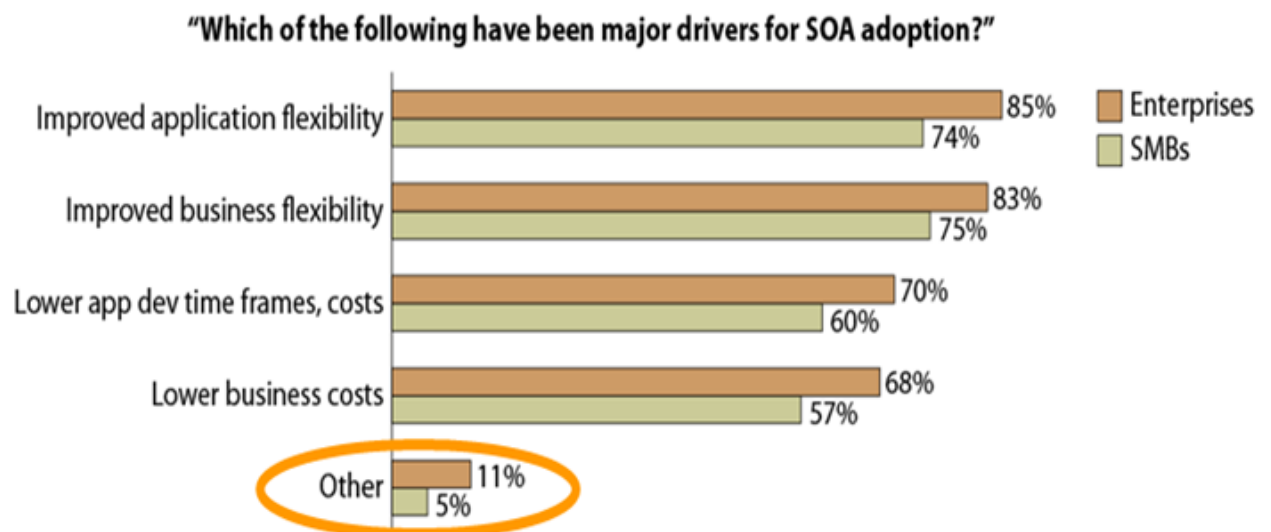
Table of Contents

- 1. The Momentum for Service-Oriented Architecture (SOA)1**
- 2. Challenges in SOA Testing.....2**
- 3. Successful Approaches to SOA Testing2**
- 4. SOA Testing: Phases4**

1. The Momentum for Service-Oriented Architecture (SOA)

Given a highly competitive business landscape, it is imperative for enterprises to be agile in order to survive the rapid changes in technology and business environments and stay ahead of competition. In practice, this translates into the ease of adding new business models, services and functionalities in response to changing market conditions, competition, etc., with minimal disruption and alterations in the existing IT implementation.

Leading companies around the world are embracing Service-Oriented Architecture (SOA) as a route to benefit in terms of component re-usage, increasing business agility, and reduction of costs and risks. A recent survey of over 300 corporate IT executives – Enterprises and SMBs -- in North America and Europe revealed the following factors as key drivers for adoption of SOA:



Source: Forrester Research

A critical success factor for adoption and deployment of SOA applications by organizations and the realization of its intended benefits, as depicted in the figure above, is rigorous testing. With loosely coupled application components delivered as services, QA teams should focus on complete testing of business workflows across multiple technology layers of the SOA application.

This brings to sharp focus the approach adopted by companies for testing SOA implementations. This White Paper focuses on the following aspects of SOA testing:

- Challenges in testing SOA applications
- Explanation of the test model and approach
- Different levels of testing in conjunction with governance best practices.

2. Challenges in SOA Testing

SOA is a collection of loosely coupled services that integrate business services from multiple applications to deliver end-to-end support to business processes. Verification and validation of the interfaces and services (internal/external) that bring together diverse systems and platforms, coupled with allied performance and security considerations, make testing really complex.

Some of the other challenges that render conventional testing approaches ineffective while testing SOA applications are:

No user interface for the services in SOA applications

Lack of visibility into loosely-coupled business level services

Dependency on availability of any internal or external services that offer a business function to perform E2E testing

Multi-skilled test teams: domain, technology and testing knowledge.

Availability of test environment which may bring together multiple application/services

3. Successful Approaches to SOA Testing

SOA testing needs to be oriented towards the business process flow connecting components, services and data in order to execute end-to-end testing. The right approach should use the appropriate test model and test methodologies, and leverage a multi-phased testing cycle covering all layers of the SOA application.

3.1. Test Model

The V-Model is one of the most suited test models for SOA testing as it implements testing activities such as design, analysis, planning and execution throughout the SOA project lifecycle.

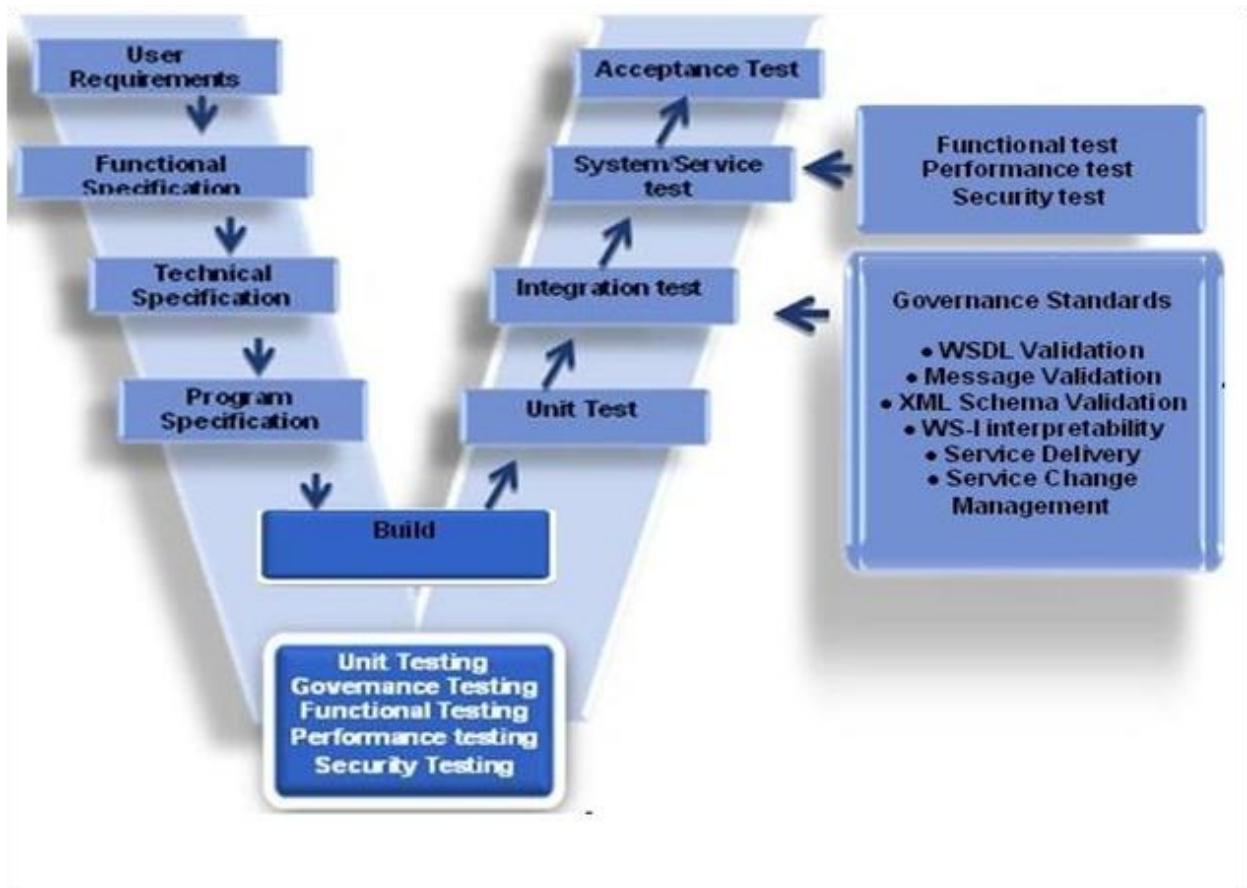


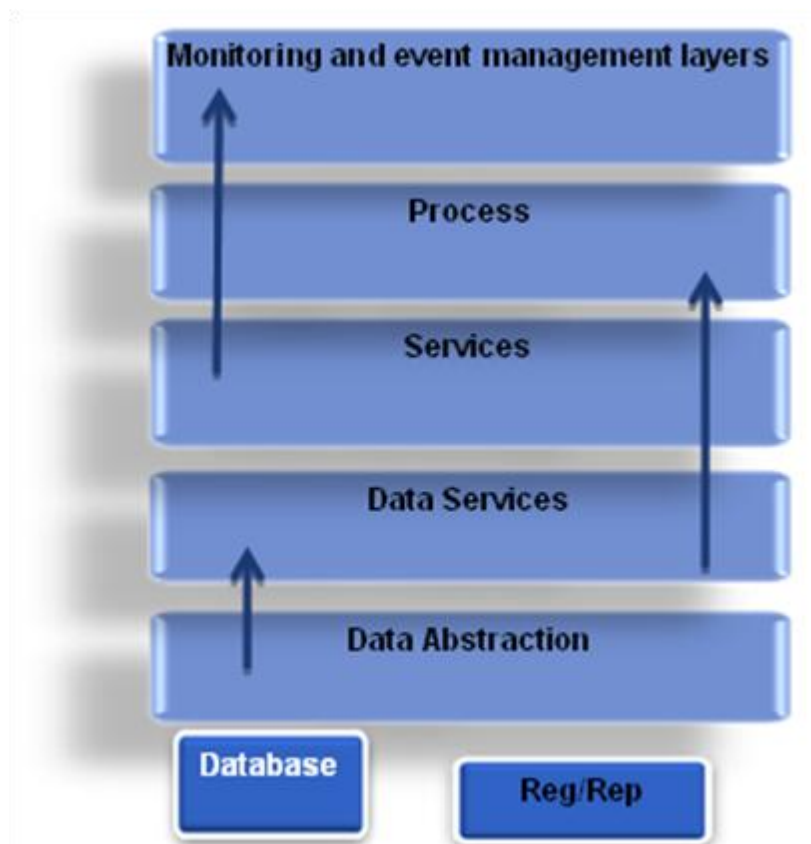
Figure1 V-Model

This model follows a top-down approach for defining the user requirements and a bottoms-up approach for testing all the services individually as well as collectively and finally, for testing the entire business system.

3.2. Test Approach

Test planning becomes crucial for SOA solutions since many of them do not have a user interface, while the ability to test multiple types of components simultaneously to form a business process requires the right approach.

In the bottoms-up approach -- adopted for successful SOA testing -- the application's architecture is broken down to its components, testing each component in the lower layers and then, moving up to components in the higher layers. The test plan includes details of how all of the components work independently and collectively.



The bottoms-up approach of SOA testing has several benefits, such as:

- Testing of each of the services of SOA architecture -- from data abstraction to data services, monitoring and event management layers.
- Testing of individual functions within a service for Governance, Functionality, Performance and Security.
- Formal peer reviews against each service to ensure that it complies with the organizational standards as well as to identify potential interoperability, performance and security defects.
- Execution of compliance testing throughout the project lifecycle to ensure enforcement of the standards and policies.
- Managing changes to the existing policies.

SOA testing approaches, in conjunction with Open Source tools, can be used to plan, manage and automate the functional and performance testing of SOA applications, as well as integrate them with Web Services-Interoperability (WS-I) standards formulated by the Web Services-Interoperability Organization.

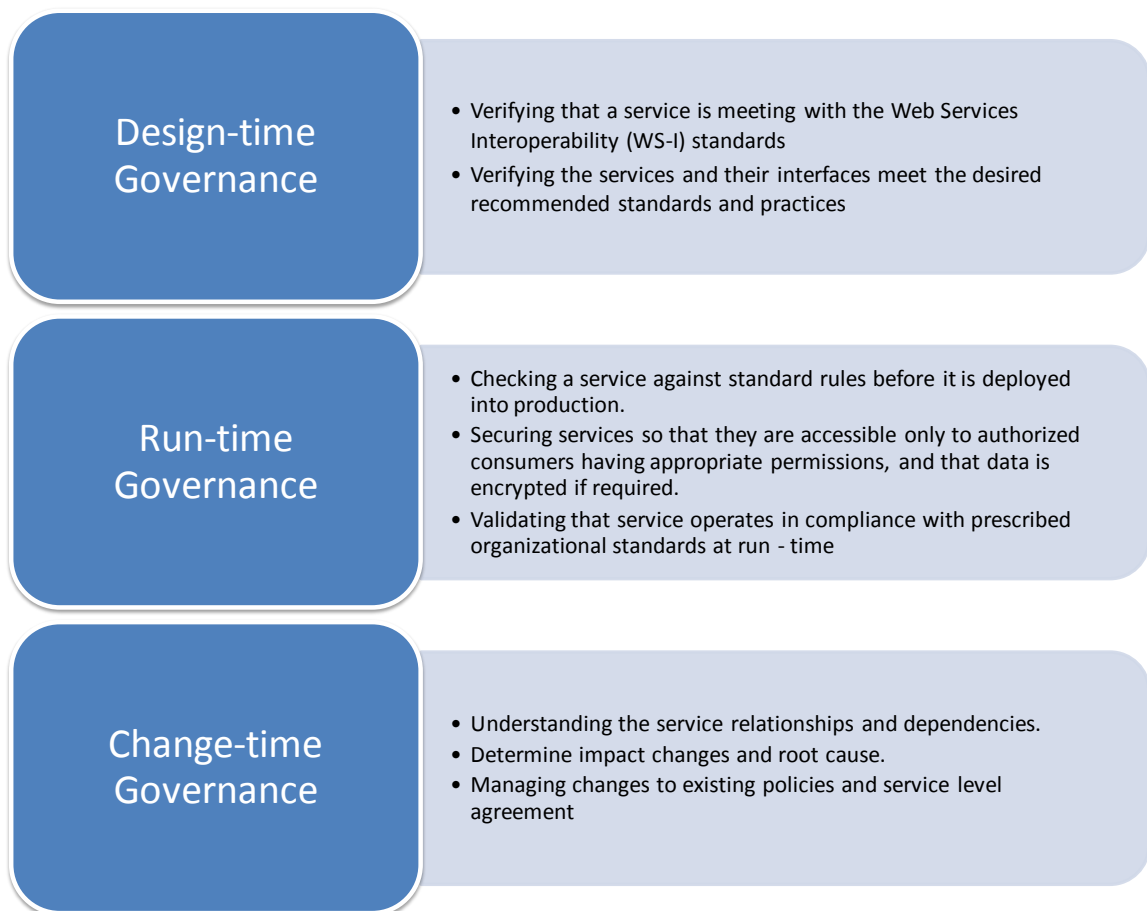
4. SOA Testing: Phases

SOA testing can be classified into the following phases:

- Governance Testing
- Unit-Level Testing
- Service-Level Testing
- Integration-Level Testing
- System Testing

4.1. Governance Testing

SOA governance refers to the standards and policies that govern the design, build and implementation of an SOA solution, and the policies that must be enforced during its design-time, run-time and change-time.



The following checklist identifies the key technical and functional requirements of an SOA governance solution, and can be used to assess the completeness of a governance implementation strategy.

Service Publication	WSDL (Web Service Definition Language) Validation	Validation of: <ul style="list-style-type: none"> • Web service endpoints • the structure of elements and attributes • namespaces
	Message Validation	Validation of: <ul style="list-style-type: none"> • message body against its schema definition • SOAP (Service-Oriented Architecture Protocol) envelope and structure in accordance with the WSDL and SOAP schemas • the header details of SOAP messages • HTTP requests and responses • secure end-to-end points identification in messages
	XML Schema Validation	Validation of: <ul style="list-style-type: none"> • the structure of elements and attributes • namespaces • data types in XML schema Verification of: <ul style="list-style-type: none"> • authorized access to services
Standard Interpretability	WS-I Interoperability	Validation of: <ul style="list-style-type: none"> • SOAP request • SOAP response messages
Service Delivery	SLA Management	Validation of: <ul style="list-style-type: none"> • performance time defined in the SLA • response times under an increasing load as defined in the SLA
Service Change Management		<ul style="list-style-type: none"> • Impact analysis to determine the implications of changing a particular service within the run-time environment • Managing changes to the existing policies and service level agreement

4.2. Unit-Level Testing

Unit-level testing is executed to ensure that the components and functions of a service are working as per the specifications.

In unit-level testing, the smallest piece of testable software in the application is isolated from the remainder of the code and tested to validate it against its expected behavior. All the components of the application are tested individually before integrating them into a service.

In unit testing, it is necessary to perform formal peer reviews of the code to ensure that it is in accordance with the organizational standards and to identify any functional, performance and security defects.

4.3. Service-Level Testing

Service-level testing is carried out with the objective of ensuring that the service meets the requirements as specified. More importantly, it should also verify that the service meets the business and operational requirements of the other processes using it.

4.4. Integration-Level Testing

The integration test phase focuses on evaluation of the service interfaces. The purpose of integration testing is to determine whether the interface behavior and information-sharing between the services are working as specified or not.

To ensure that all the services delivered to the test phase comply with the defined standards, format and data validation, the test phase also includes testing of the internal as well as external services of the organization.

4.5. System-Level Testing

System-level testing ensures that the application meets the business acceptance criteria at the overall system level.

The following table shows the various testing phases and the test types applicable in each phase:

Test Phase	Functional	Performance	Security	Inter-operability	Backward Compatibility	Compliance
Unit-Level Testing	Yes	Yes	Yes	Yes	Yes	Yes
Service-Level Testing	Yes	Yes	Yes	-	-	Yes
Integration-Level Testing	Yes	-	Yes	Yes	Yes	Yes
System-Level Testing	Yes	Yes	Yes	Yes	Yes	Yes

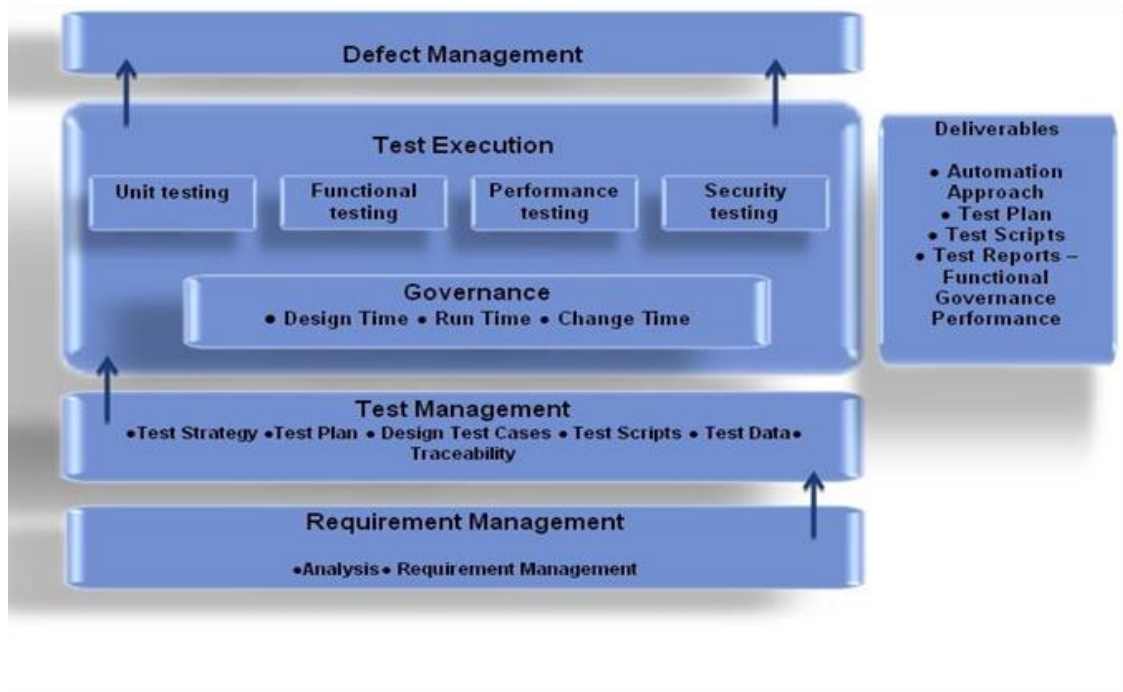


Figure 1.2 End-to-End SOA Testing

4.6. Phase Activities and Deliverables

The testing of SOA applications can be broken down into the following four activities, and in each process, certain deliverables are provided to the customer:

- **Initiation:** In this process, Understanding Document, Transition Plan and the Test Methodology Document are provided to the customer.
- **Test Management:** During this process, the Test Plan / Test Strategy and Test Cases / Test Scripts are provided to the customer.
- **Test Execution:** During Test Execution, the customer is provided with Test Reports, Web Services, Interoperability Report, Conformance Report and Performance Report.
- **Defect Management:** In this last process, the Defect Summary Report is presented to the customer.

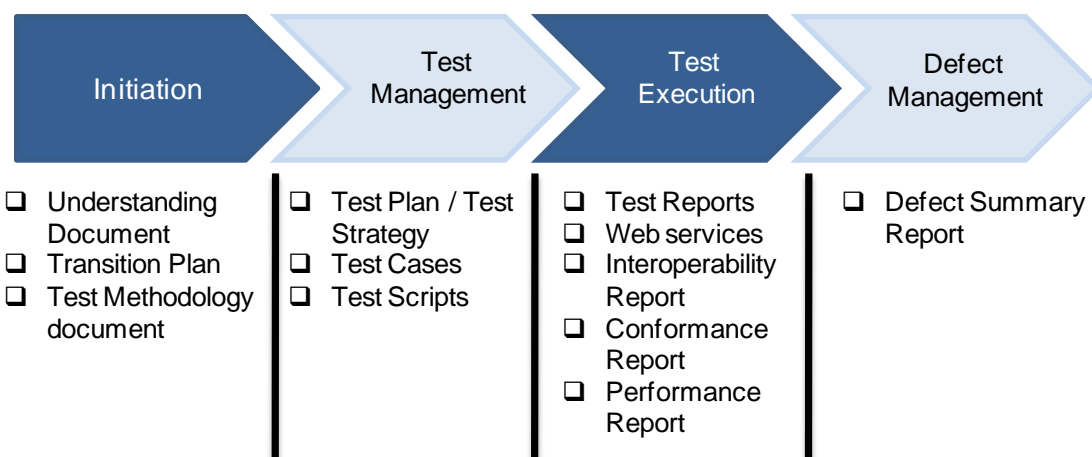


Figure 1.3: Deliverables

Summary

A highly competitive business environment, coupled with rapid technological changes, necessitates it for organizations to embrace SOA in order to stay ahead of competition. SOA is a collection of loosely coupled services that integrate business services from multiple applications to deliver end-to-end support to business processes. A critical success factor for the adoption and deployment of SOA applications by organizations is rigorous testing.

However, verification and validation of the interfaces and services – both internal and external -- that bring diverse systems and platforms together, along with allied performance and security considerations, make testing of SOA applications really complex. Moreover, there are a number of other factors that pose a challenge in testing SOA applications with conventional testing approaches.

End-to-end testing of SOA applications requires the use of the right approach – one with the appropriate test model and test methodologies, and which leverages a multi-phased testing cycle covering all layers of the application. The V-Model is one of the most suited test models for SOA testing as it implements testing activities such as designing, analysis, planning and execution throughout the SOA project lifecycle.

For successful testing of SOA applications, the bottoms-up approach is very widely used. In this approach, the application's architecture is broken down to its components. Initially, each component in the lower layers and subsequently, the higher layers is tested individually. The test plan includes details of how all of the components work independently as well as collectively.

SOA testing can be classified into five phases -- Governance Testing, Unit-Level Testing, Service-Level Testing, Integration-Level Testing and System Testing -- and four activities:

- Initiation
- Test Management
- Test Execution
- Defect Management.

Testing of SOA through these methodologies and processes not only ensures its re-usability and agility but also smooth functioning of its component services when integrated into applications.