



CresTech Software Systems

Your Testing Partner

CresTech

www.crestech.in

White Paper | SOA: A Testing Perspective

Abstract:

Service Oriented Architecture (SOA) makes software quality both more important and more difficult to achieve. While enough material is available focused on the design and coding of Web services and service oriented architectures (SOA), not much has been written about testing of such applications. This paper is focused to provide a perspective to QA community regarding SOA and its testing.

A Problem:

There is manufacturing giant (xyz), which has hundreds of suppliers (vendors) associated with it. A customer logs in to xyz's sales portal and places an enquiry for an order. At the run time, xyz has to deliver back information to the customer whether it's possible to fulfill the order and the possible delivery date.

The order fulfillment process mentioned above can be divided in following steps.

- a) From its own internal system, xyz finds out status of its inventory and capability to fulfill the order.
- b) If xyz needs to get material from its suppliers, it connects to supplier's inventory system to get the delivery information.
- c) Based on the information gathered, xyz sends the order fulfillment information back to customer.

Issues:

- a) XYZ is having a system which is different from its suppliers. For example, XYZ has order management system from SAP, while supplier is using its own tailor made order management system. Both the systems are based on different platforms and technologies.

- b) For processing of information at the run time, there should be a seamless integration between XYZ's system and supplier's system.

A solution:

We need an architecture which allows seamless integration between the two heterogeneous systems mentioned in the above problem. To achieve the same, both the systems should provide interfaces, based on common standards, which each other can discover and access on demand.

Introduction: SOA

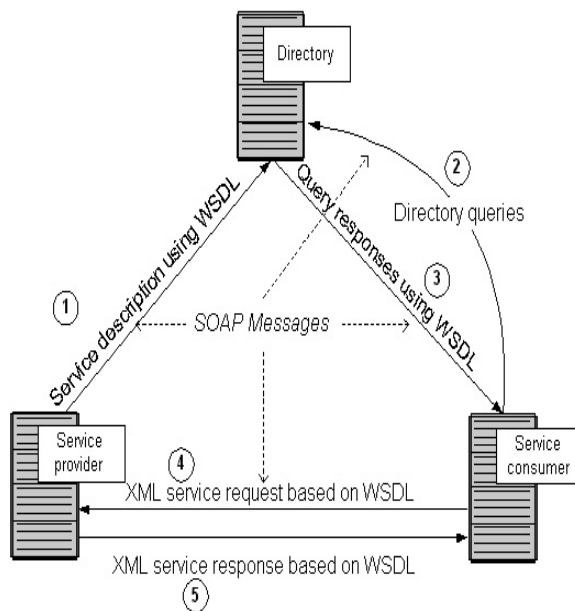
Service-oriented architecture is an architectural approach for building systems or applications that use a set of services and not just a system that is built as a set of services. A service is an implementation of a well-defined piece of business functionality, with a published interface that is discoverable and can be used by service consumers when building different applications and business processes.

SOA is an architecture made up of components and interconnections that stress interoperability and location transparency. This architecture is particularly applicable when multiple applications running on varied technologies and platforms have to communicate with each other.

Web Services: Backbone of SOA in web paradigm

Used primarily as a means for businesses to communicate with each other and with clients, Web services allow organizations to communicate data without intimate knowledge of each other's IT systems. The term Web services describes a standardized way of integrating Web-based applications using the XML, SOAP, WSDL and UDDI open standards over an Internet protocol backbone.

WSDL (Web Services Description Language): WSDL forms the basis for Web Services. Using WSDL, a service provider describes its services and publishes it in the directory of services. When a service consumer queries the directory to locate a service, part of WSDL provided by service provider is passed to the consumer. The service consumer uses the WSDL to send a request to the directly to service provider.



UDDI (Universal Description, Discovery, and Integration): It's a directory service where businesses can register and search for Web services.

- ▶ UDDI stands for Universal Description, Discovery and Integration.
- ▶ UDDI is a directory for storing information about web services.
- ▶ UDDI is a directory of web service interfaces described by WSDL.
- ▶ UDDI communicates via SOAP.

SOAP (Simple Object Access Protocol): SOAP is a platform and language independent, simple XML based communication protocol to let applications exchange information over

HTTP. It's protocol for accessing a Web Service.

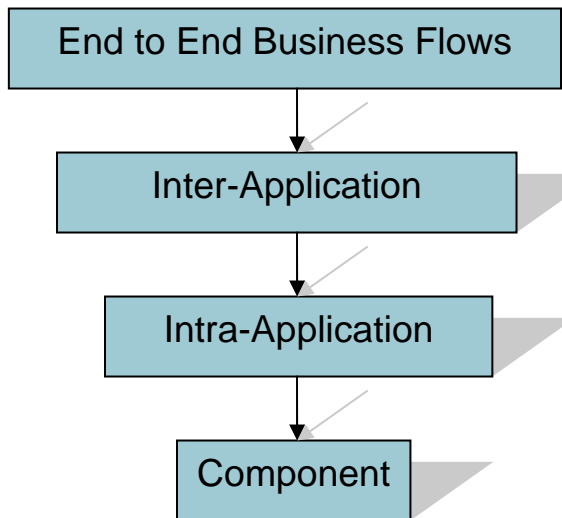
Implication of SOA on testing

How web-services testing are different and challenging: While this architecture holds great benefit in terms of scalability and extensibility of the system, it poses a great challenge for the tester. Some of main challenges are listed below.

- ▶ In case of web services based architecture, overall system may actually be derived from components constructed by different stakeholders. It is difficult to establish consisted code quality of code.
- ▶ Web services-based integration highlights the importance of validating interface points and message formats rather than simply testing at the graphical user interface level. Using testing tools that utilize GUI-driven automation is simply inadequate for a Web services project.
- ▶ It's very difficult to manage the complexity of testing huge no of services.
- ▶ Infinite consumers possible for services (thick client, thin client etc.);
- ▶ Conceptualizing test bed close to the end user environment;
- ▶ Traditional testing tool haven't evolved yet to support the web-services standards.

Testing Approach

Functionality: SOA applications are built with interoperability in mind, resulting in business processes that span across departments (intra) and organizations (inter). The test designs have to be specified in what is called a "Top Down approach", ensuring the functional tests address end-to-end business flows. The specified flows are dissected to isolate the inter application tests. From the inter-application tests, intra application tests are discovered, finally resulting in unit tests.



Interoperability: Today, mainstream development platforms—such as Microsoft's .NET and Sun's Java 2 Enterprise Edition (J2EE), as well as open source alternatives (e.g., Perl and PHP)—provide frameworks to implement Web services. Components implemented in disparate platforms using different languages can interact transparently through a call-and-return mechanism. That is possible because Web services define the interface format and communication protocols but do not restrict the implementation language or platform.

To simulate interoperability tests, building a test environment which is

close to a real life scenario is a great challenge. Taking advantage of software virtualization is one good approach.

Creating reusable test assets that work on many different platform combinations would also be of added value.

Performance: When dealing with a distributed system with multiple intermediaries, it is important to perform stress/load testing in a pre-deployment mode.

The greatest challenge when it comes to performance testing of service oriented applications is to design the end user workload and model the end user environment (applications involved, software, hardware etc). The predominant practice is to define and design usage modeling for the entire suite of applications utilizing user community modeling techniques.

Once the appropriate performance scenarios have been defined, multiple test tools/techniques are required because of the presence of different platforms and technologies.

During test execution, monitoring application performance and collating data would be a challenge since there is no "one stop shop" tool which gives insight into the overall big picture.

Summary

This paper attempts to introduce SOA and some of the risks and challenges associated with it from the perspective of testing. Testing SOA applications is still in its early stages and new techniques are evolving as testing SOA based applications becomes more mature.