Journal of Theoretical and Applied Information Technology
15th February 2012. Vol. 36 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

A ROAD MAP TO REGRESSION TESTING OF SERVICE-ORIENTED ARCHITECTURE (SOA) BASED APPLICATIONS

[1]RAJANI KANTA MOHANTY, [2]BINOD KUMAR PATTANAYAK*, [3]BHAGABAT PUTHAL, [4]DURGA PRASAD MOHAPATRA
[1,2,3:] Siksha 'O' Anusandhan Deemed to be University, Bhubaneswar, INDIA-751030
[4] National institute of Technology, Rourkela, INDIA-769008

E-mail: rkm.bbs@gmail.com[1], bkp_iter@yahoo.co.in[2], bputhal@gmail.com[3], durga@nitrkl.ac.in[4]
*: Corresponding Author

ABSTRACT

Modular approach to application software development has become significantly popular among the enterprises. However, Service-Oriented Architecture (SOA)-based applications represent the majority of modular application software. The major concerns of developers of these applications are oriented around the reliability and fault-free implementation, which necessitate proper testing of the application. Testing of SOA-based application plays a critical role in ensuring a successful deployment of applications. Testing strategies of such applications like unit testing, integration testing and system testing may somehow resemble to that of traditional application software. However, regression testing of SOA-based applications, which can be conducted during the maintenance phase, may present several challenges. To begin with, in this research paper, we attempt to explore a road map to regression testing of SOA-based applications.

Keywords: *Software, Service, SOA, Testing, Regression Testing*

1. INTRODUCTION

In the global scenario of software engineering, testing of application software represents an inevitable component. Testing consumes around 40-85% of effort as well as time pertaining to Software Life Cycle [1]. The motivation behind testing is to identify the short comings in it, assure the desired quality and validate the software as per requirement specification. Besides the apparent straight forwardness of checking of a sample of runs, however, testing includes a variety of activities, techniques, actors and poses a set of challenges [2]. A common testing strategy is to subject the product to several phases of testing such as unit, integration and system testing. These testing phases consume significant project resources as well as time. As software companies continue to search for ways to reduce cycle time, cost of labor, cost of tools and technologies etc. while increasing quality, software testing processes emerge as a prime area of investigation [1]. Different surveys in regard to this have shown that testing can facilitate finding of errors in the application at the earliest, and therefore, time and cost related rectification become more viable economically.

Like traditional software testing, a significant portion of service oriented architecture (SOA)-based development project resources is consumed by testing phases [3].At the core of SOA is the concept of services. The dictionary meaning of service is "a unit of work performed by one for another". In SOA, services are business tasks performed by an external component according to a predefined specification or contract. SOA is fundamentally a collection of services delivering functionality that can be used and reused while developing an application or integrating within the enterprise or the business partners [4]. SOA is more of an architectural style for an enterprise, not in the sense of the word "architecture". The new architecture style SOA-based application brings about its own testing challenges. The dynamic and adaptive nature of SOA makes most of the existing techniques not directly applicable to test service-centric applications [5]. The distributed nature of SOA-based application, lack of a user interface, inaccessibility of service code and improper information about the run time topology of the services impose significant challenges in making a viable testing strategy for SOA-based application or system [4, 6, 7].

Testing of SOA-based application plays a critical role in ensuring successful deployment in any enterprise. SOA testing must span several levels from individual services, inter-enterprise federation of systems and must cover functional

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

and non-functional aspects. SOA's unique combination of features such as run time discovery of services, ultra late binding, Quality of Service (QoS) aware composition, and Services Level Agreement (SLA) automated negotiation challenge many existing testing techniques. Service reliability, testing and verification techniques are not mature enough to support dependable and trustworthy computing. Service-orientation however posses a new and different challenge to testers especially when it comes to the testing the interaction between heterogeneous and loosely coupled independently developed services. Moreover, regression testing of SOA-based applications significantly differs from that of traditional application software. To address these issues subsequently, we resume with a road map to regression testing of SOA-based applications.

2. OVERVIEW OF SOA BASED APPLICATIONS

A SOA is an information technology approach or system in which applications make use of services available in a network. Implementing service oriented architecture can invoke developing applications that use services making applications available as services so that other applications can use these services or both. What distinguishes a SOA from other architectures is loose-coupling. Loose-coupling here implies that the client of a service is essentially independent of the service [8, 9]. Most of these concepts of SOA-based systems are not new. SOA has evolved from distributed computing which has been around for last two decades. But, the difference between SOA and distributed computing lies in how the business logic is partitioned, where the units reside and how the units interact with each other [7]. In SOA, components that implement services interact with each other in a more dynamic manner. SOA-based applications are a combination of web components, middle-tier components, servers and legacy systems [10]. The unique characteristics of SOA-based systems can magnify the intensity of the testing challenges significantly. SOA-based application consists of services provided by independent providers and its development process is not any more under the control of a single stake holder/organization. Thus, if testing a service under development could be assimilated to traditional unit testing, it is evident that the situation drastically changes in testing a service behavior while interacting with the "real world". Similar is the case for integration testing that will ensure smooth interoperability of services in a highly dynamic and adaptive environment.

In general, any software testing technology is built over two pillars: control and observation [10]. In each given context, a cost effective test approach relies on strategies to understand how the systems under test can be better controlled (e.g. making invocations to the system, simulating and changing the execution environment), observed (i.e. watching how the system reacts and behaves in response to a test request) and also on methods to relate control and observation, i.e. the test inputs with the observed responses. A SOA-based application developer also intends to diligently test a new application according to the recommended test practices for SOA testing, and can choose between on-line and off-line testing. SOA-based application needs more stricter organizational discipline in testing than conventional development paradigm [10].

3. PROPOSED ROAD MAP TO REGRESSION TESTING OF SOA BASED APPLICATIONS

In this section, we present a road map to regression testing of SOA-based application. The proposed by us road map is depicted in Fig.1.

Modular application software can be globally classified into three categories: Component based software, Reuse-oriented software and SOA-based software.

3.1. Component Based Software

Component based development of software emphasizes on the separation of concerns in respect of wide-ranging functionality available throughout a given software system. This practice aims at bringing about an equally wide-ranging degree of both the long-term and short-term benefits for the software itself and for the organizations that sponsor such software. Software engineers regard components as part of the starting platform for service-orientation. Components play this role, for example, in Web services, and more recently, in service-oriented architectures (SOA), whereby a component is converted by the Web service into a *service* and subsequently inherits further characteristics beyond that of an ordinary component [11].

3.2. Reuse-oriented Software

In the majority of software projects, there is some amount of reuse of software is incorporated. This usually happens informally when people

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

working on the project know of design or code which is similar to that required. They look for these aspects, modify them as required and incorporate them into their system. This approach relies on a large base of reusable software components which can be accessed and some integrating framework for these components. Sometimes, these components are systems in their own right (also known as Commercial Off-The-Shelf System (COTS)) that may be used to provide specific functionality [12].

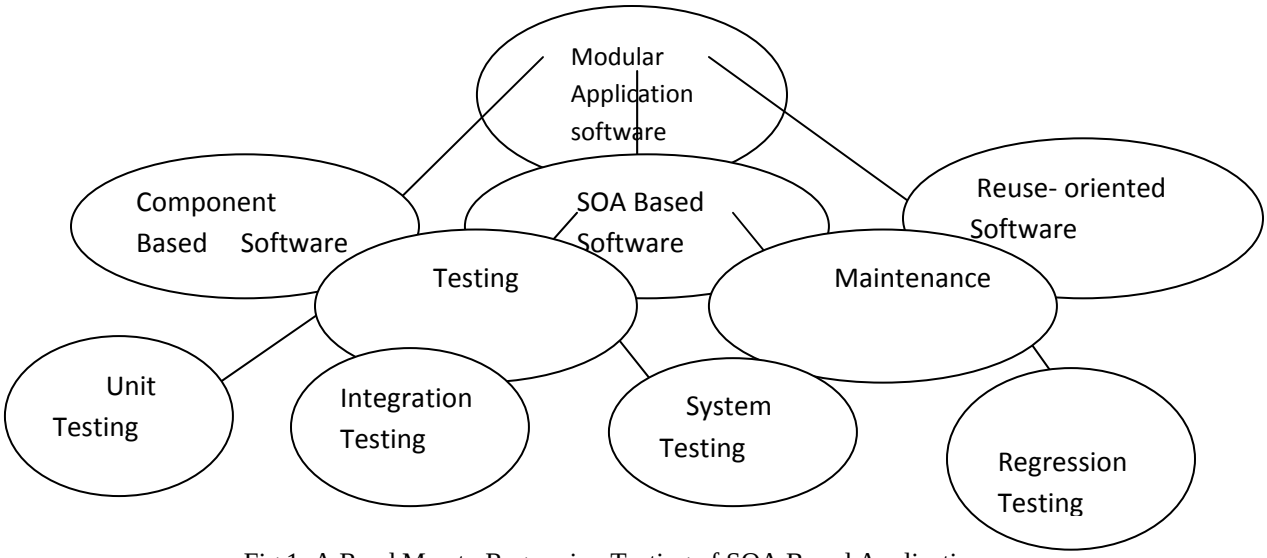

Fig.1: A Road Map to Regression Testing of SOA Based Applications

3.3. SOA Based Software

A service represents a repeatable business process or task. Services effectively encapsulate the functional units of an application thereby providing an interface that is implementation independent. Service orientation refers to integration of business applications and processes as linked services. Service oriented architecture can be understood directly from a person's role and context. From architecture point of view, SOA represents an architectural style that incorporates service orientation. From a business perspective, SOA accumulates a set of business services that capture the intended business design. From implementation perspective, SOA incorporates standardized infrastructure, programming model and technologies. And, from operational aspects, SOA comprises of a set of agreements and provides that specify the quality of service [13].

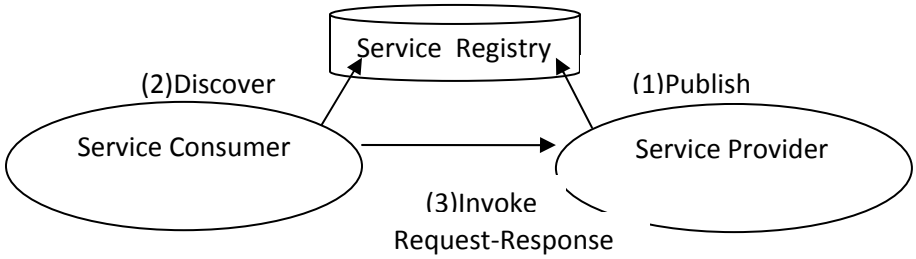

Fig.2: Service Oriented Architecture (SOA)

Basic components of SOA are demonstrated in Fig.2. SOA comprises of three components: Service Provider, Service Consumer and Service Registry. The Service Provider is responsible for publishing the service interface to Service Registry. Subsequently, service interface and access information can be made available to Service Consumer by Service Registry. The

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Service Consumer can locate the service in Service Registry and then invokes the desired service from Service Provider.

4. SOA TESTING

In contrast with traditional software testing, SOA testing imposes a wide range of challenges. In this section, we discuss different perspectives to SOA testing as well as different levels pertaining to SOA testing.

4.1. Testing Perspectives

Unlike traditional application software, SOA imposes different needs and challenges to different stakeholders involved in the testing activities, as detailed below [14].

a) **Service Developer:** The service developer needs to test the service to detect maximum number of failures in order to release a highly reliable service. Since the developer owns the service implementation, he/she can perform white-box testing. However, test cases derived by the developer may not be able to yield in a real usage scenario [14].
b) **Service Provider:** The service provider can test the service to ensure the requirements as per the SLA with the consumer [4, 15]. However, the service provider cannot implement white-box testing.
c) **Service integrator:** testing by the service integrator is carried out in order to explore the functional and non-functional assumptions made during the design phase of a service.
d) **Third-party Certifier:** A third party certification can be used by the service integrator for the assessment of fault-proneness of a service.
e) **End-User:** The end-user, in fact, does not have any scope for service testing, rather his/her concern may be only around the fact that the service used by the end-user works properly or not.

4.2. Testing Levels

Different levels of testing of SOA-based applications are detailed below.

a) **Unit Testing:** Unit testing of SOA-based applications represents testing of a single atomic service and could be somehow equivalent to component based testing.
b) **Integration Testing:** Integration testing for SOA-based applications is motivated by the fact that SOA shifts the development paradigm from monolithically applications to applications composed of services, distributed over the network, developed and hosted by different organizations [14, 16, 17].
c) **System testing:** System testing is implemented to validate a fully developed SOA based application to ensure that it meets the desired requirement. System testing can be implemented in three variations: Alpha testing, Beta testing and Acceptance testing.
d) **Regression Testing:** Regression testing is the selective retesting of a system or component to verify that modifications have not caused unintended effects and that the system or component complies with its specified requirements [14, 16, 17].

5. REGRESSION TESTING OF SOA-BASED APPLICATIONS

5.1 Process:

Regression testing differs from other levels of testing in that it can be implemented during maintenance rather than during development, taking into account its run time behavioral aspects.

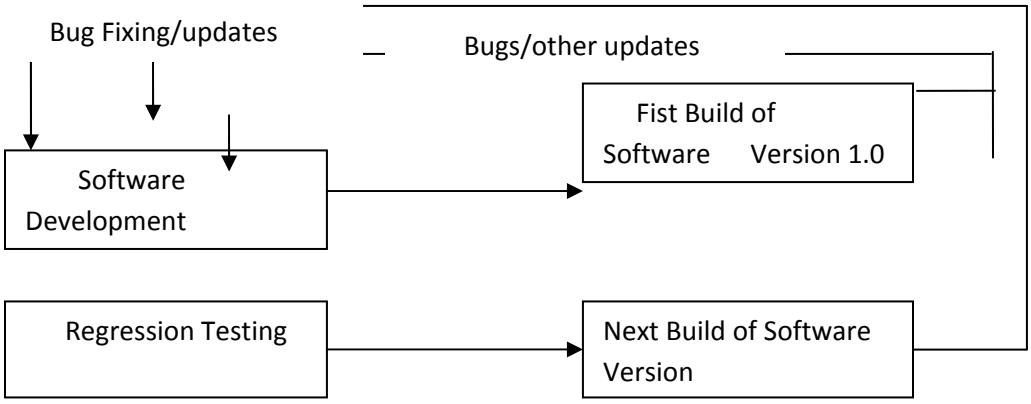

Fig.3. Regression Testing Produces Quality Software

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

It represents a selective retesting of components so that modifications incorporated into a component do not necessarily affect the behavior of other components as well as the system as a whole. After a program is modified, it must be ensured that the modifications work correctly and the unmodified portions of the program have not been adversely affected by these modifications. Thus, in course of regression testing, the modified application needs to be executed on all possible tests in order to validate that it still behaves in the same way except for the portions which are modified. Hence, regression testing can be defined as the software maintenance task performed on a modified program to instill confidence that changes incorporated are correct and have not adversely affected the unchanged portions of the program [18]. Unlike unit testing, integration testing or system testing which are performed during software life cycle, regression testing is performed during maintenance of SOA-based applications.

5.2: Challenges of Regression Testing:

Key issues that limit the testability of service-oriented applications include lack of observability of the service code and structure, its dynamicity and adaptivieness, lack of control, lack of trust and finally cost of testing [14]. An issue that distinguishes service-centric systems from traditional applications is the lack of control a system integrator has over the services he/she is using. System Integrators select services to be integrated in their systems and assume such services to be capable of maintaining their functional and non-functional characteristics while being used. However, a system exposed as a service undergoes like any other System maintenance and evolution activities. Maintenance and evolution strategies are out of the system integrators control, and any changes to a service may significantly affect all the systems using it. This makes service-centric systems different from component-based systems: when a component evolves, this does not affect systems that use previous versions of the component itself. Component-based systems physically integrate a copy of the component and, despite the improvements or bug fixing performed in the new component release, systems can continue to use an old version. In such a context, several evolution scenarios may arise: i) Changes that do not require modifying the service interface and/or specification (e.g., because the provider believes this is a minor update, and as a consequence, the changes remain hidden from whoever is using the service); ii) Changes that do not affect the service functional behavior of the service, but affect its QoS. Once again, these are not always documented by the service provider and, as a consequence, the QoS of the system/composition using such a service can be affected. iii) A service may comprise of other services. As a matter of fact, changes are propagated between different system integrators, and it may happen that the distance between the change and the actor affected by the change makes unlikely that, even if the change is advertised, the integrator can be able to get it and react accordingly [14].

6. CONCLUSION AND FUTURE WORK

In this research paper, we present a road map to regression testing of SOA-based applications. Different testing strategies pertaining to SOA-based applications are discussed along with the challenges related to it. Specifically, regression testing of SOA based applications imposes a set of challenges, which we need to address in our subsequent research. In particular, we intend to implement model based approach to regression testing of SOA-based applications and explore its viability in real world usage scenario.

REFERENCES:

[1] Dustdar, S., Haslinger, S.: Testing of service-oriented architectures - a practical approach. In: Weske, M., Liggesmeyer, P. (eds.) NODE 2004. LNCS, vol. 3263, pp.97–109. Springer, Heidelberg (2004)

[2] Antonia Bertolino, Software Testing Research,: Achievements, Challenges, Dreams,Future of SoftwareEngineering(FOSE'07),pages 1-19, IEEE Computer Society, 2007

[3] Ayaz Farooq, Konstantina Georgieva and Reiner R. Dumke, “Challenges in Evaluating SOA Test Process”, IWSM/Metrikon/Mensura '08 Proceedings of the International Conferences on Software Process and Product Measurement, Springer-Verlag Berlin, Heidelberg,pages 107-112 ©2008.

[4] Canfora, G., Di Penta, M.: Testing services and service-centric systems: Challenges and opportunities. IT Professional **8** (2), 10–17 (2006).

[5] Ribarov, L., Manova, I., Ilieva, S.: Testing in a service-oriented world. In: InfoTech 2007:

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Proceedings of the International Conference on Information Technologies(2007)

[6] Gold, N.; Mohan, A.; Knight, C.; Munro, M.; Understanding service-oriented software, Software, IEEE ,Volume: 21 Issue:2,on page(s): 71 – 77,March-April 2004

[7] Parveen, T., Tilley, S.: A research agenda for testing SOA-based systems. In: Proceeding of 2008 2nd Annual IEEE Systems Conference, pp. 1–6. IEEE Computer Society, Los Alamitos (2008)

[8] Tsai,W.T., Gao, J.,Wei, X., Chen, Y.: Testability of software in service-oriented architecture. In: 30th Annual International Computer Software and Applications Conference (COMPSAC 2006), 17-21 September 2006, Chicago, Illinois, USA. 163–170 (2006)

[9] Ed Ort, SOA and Web Services Concepts,Technologies,and Tools, Sun Microsystems,A PRIL 2005

[10]Antonia Bertolino , Approaches to testing service-oriented software systems, QUASOSS '09 Proceedings of the 1st international workshop on Quality of service-oriented software systems,PAGES 1-2, ACM August 2009

[11] SOA fundamentals, www.wikipedia.com.

[12] Ian Soomerville, Software Engineering,6th edition, Pearson Education ,2006.

[13] Torry Harris Business Solutions. White Paper. SOA test methodology.

[14] G. Canfora and M. Di Penta. Service Oriented Architecture Testing : A Survey, pages 78–105. Number 5413 in LNCS. Springer, 2009.

[15] Kontogiannis, K., Lewis, G.A., Smith, D.B.: A research agenda for service-oriented architecture. In: SDSOA 2008: Proceedings of the 2nd international workshop on Systems development in SOA environments, pp. 1–6. ACM, New York (2008).

[16] Rajib Mall ,Fundamentals of Software Engineering,2nd edition, PHI ,2007

[17] Wei-Tek Tsai, Xinyu Zhou, Yinong Chen, Xiaoying Bai, On Testing and Evaluating Service-Oriented Software, Computer (2008) ,Volume: 41, Issue: 8, Publisher: IEEE Computer Society, Pages: 40-46,2008.

[18] Naresh Chauhan, Software Testing: principles and practices, Oxford university Press ,2010