An approach of Regression testing for Service Oriented Architecture

Bharat Choudhary, Vineet Richhariya

Department of Computer Sc. & Engg. HOD, Department of Computer Sc. & Engg.
Laxmi Narain College of Technology
Bhopal, Madhya Pradesh, India
E-mail: hi.bharat2002@gmail.com E-mail: vineetric100@gmail.com

Abstract - Service Oriented Architecture (SOA) is an application architecture in which services are described using a description language. It has changed the way business enterprises with technology with a very fast demand of re-alignment time very short. Besides this the testing of SOA is also a challenging factor for implementation of enterprise architecture. Component testing and integration are on key testing strategy. On the other hand the regression testing is identifying unintentional errors that may have been introduced as a result of changing a program module. In this paper we define an approach of regression testing for Service Oriented Architecture.

Keywords - Services, Service Oriented Architecture, Regression Testing.

I. INTRODUCTION

Regression Testing in SOA is one of the major matters of concern because of the dynamic nature of the Service binding and the adaptability to accommodate the changed requirement. The actual configuration of the service is known only at run time, so it becomes very complex to verify whether the changes made in earlier version of the system are correct or not and it does not affect the functionality and performance of the existing system.

Regression testing help identify changes between a selected product release and a previous release of the product called a baseline. A baseline is recorded snapshot of desirable product behaviour. This expected behaviour is then used to ensure that nothing has been broken in the system as a result of changes introduced in a program module.

The paper structure is as follows. In next section, interpretation of the survey is given. The next section discusses about the basic approach of Regression Testing. In section 4 a comparison table showing the merits and demerits of each approach is given. In section 5 different challenges involved in the Testing of SOA are described and then in subsequent sections Testing tools, Future Research Issue, Scope of the survey and Conclusions are covered.

II. INTERPRETATION OF THE SURVEY

The survey is mainly concentrated on giving an approach of regression testing for Service Oriented Architecture. This paper is mainly finding the new testing approach for SOA. There we overcome the drawback of existing testing approaches and tries to solve the problem which are present in existing approach.

Testing is the practice of making objective judgments regarding the extent to which the system (device) meets, exceeds or fails to meet stated objectives. [1]

Web services are a concrete implementation of the SOA concept. They incorporate applications that provide a set of operations accessed by other applications through the Internet, using XML standards such as SOAP, WSDL, and UDDI. This technology enables to interconnect different enterprises and to build complex business applications which are used in safety critical environments.

SOAs and web services add new factors that need to be considered during software development. Distribution, lack of control and observability, dynamic integration with other applications, and XML standards usage are key features of this new type of software.

In this context, software testing is essential to guarantee a high degree of service quality and reliability. Consolidated test approaches have been reused in SOA applications. However, many testing techniques cannot be directly applied due to the dynamic and adaptive nature of SOA [7].

III. AN APPROACH OF REGRESSION TESTING

Regression testing is an silent hero of the software testing world – this indispensable testing strategy is used daily by many software companies, worldwide. Wikipedia defines regression testing as: “Software testing that seeks to uncover new software bugs or regressions in existing functional and
non-functional areas of a system after changes, such as enhancements, patches or configuration changes, have been made to them. The intent of regression testing is to ensure that a change, such as a bug fix, did not introduce new faults. One of the main reasons for regression testing is to determine whether a change in one part of the software affects other parts of the software”.

An issue that makes service-centric systems very different from traditional applications is the lack of control a system integrator has over the services one is using. System integrators select services to be integrated in their systems and assume that such services will maintain their functional and non-functional characteristics while being used. However, this is not the case: a system exposed as a service undergoes-as 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 impact 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 [4].

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 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.

In Figure 1 the SOA Testing Life cycle depicts that Regression and Governance Testing are the activities that span all over the testing life cycle of the SOA Application. Testing SOA applications is different from traditional testing approaches.

IV. COMPARE THE VARIOUS REGRESSION TESTING APPROACH

Regression testing is an expensive process in which test suites are executed ensuring that no new errors have been introduced into previously tested code.

Regression testing is defined [3] as “the process of retesting the modified parts of the software and ensuring that no new errors have been introduced into previously tested code”.

There are various regression testing techniques and the behavior of each technique is discussed:

A. Selective Regression Testing

Selective regression testing includes test selection from previously run test suite and test coverage identification. Suppose n number of test cases are there to completely test the system then we will select subset of the tests and repeat that sub set of the test cases again to ensure that the changes in the system does not affect the functionality of the other part of the system. It construct control flow graph using algorithms. It takes less time for retest. But there problem is to find where the additional testing is required.
B. Risk Based Regression Testing

Risk based regression testing is a standard technique for reducing this risk is to periodically rerun all test cases to make sure they still pass. A risk area is identified which is considered as the most coverage area that get affected by the changes made in the system. Though the Risk Based Regression Testing does not guarantee the success of the Test. Mainly this technique finds the risky area of the code. It is simple for implementation and also supports scalability.

C. State Based Regression Testing

A set of coverage criteria is proposed based on control and data flow in UML state diagram and it is shown how to generate test cases satisfying these criteria from UML state diagrams. First control flow is identified by transforming UML state diagram in to Extended Finite State Machines (EFSM). This technique use the DFS and BFS method for traversing and there nodes and edges are can’t be change. [12].

D. Code Based Regression Testing

In code based regression testing, modified code and the code associated with that are considered and testing is performed to ensure whether the changes in code is working properly or not. It is a well approach for unit testing. But it is not support scalability.

E. Model Based Regression Testing

Model based regression testing is applied when the code is unavailable. In model-centric approaches, modifications are first done to the models, rather than to code. In model based regression testing test cases are selected if they traverse the modified model parts. UML class and sequence diagrams are used to implement Model Based Regression Testing but in this approach the identification of the modification is a challenge, which can be marked at towards a stable methodology for SOA Regression Testing. In this approach less effort is required and has better complexity management.

F. Observation

Various approaches have addressed the need and methodologies of regression testing in traditional and Object-Oriented software development. SOA based system development caters to the agility of business strategies. In code based approach, regression testing is done for code changes. Also it can cater to the need for SOA-based regression testing when a service is upgraded with new versions. A combination of various UML diagrams can help us to get the dependency graph like component and state chart diagram. [8]

V. CHALLENGES IN TESTING OF SOA

Randy states that "accessibility to services" or the "headless nature of services" creates new challenges for testers. Testers will have to test many services without the aid of user interfaces and must perform rigorous testing to validate these services. Testing services reminds him of the challenges of testing drivers which require test harnesses and testing frameworks. The importance of strong technical knowledge. He states that testers come from different areas of the enterprise (business, development, etc.) and may not necessarily have the proper skills for SOA. He also stresses the importance of the need for more business orientation which he calls the "forgotten area of testing". Testers, who focus solely on the technology, miss the business process side of the equation.

How services are always available which is different from n-tier architectures where "you are in control". With SOA you have services calling other services and in the world of mashups, testers don’t always have knowledge of how these services will be used before deployment. Scale and performance are also high on Frank's list of challenges. Depend on the above discussion we can categories these in different points as follows:

A. Service code and structure can’t observable:
   The users and system integrators are just like a medium, because of that it prevents the white box testing method that require knowledge of the structure of code and flow of data.

B. Security Model:
   Security models built into an application may no longer suffice when an application exposes its capabilities as services that can be used by other applications. So we have to consider about security model of service oriented architecture. That is, application-managed security is not the right model for securing services.

C. Insufficient Controlling:
   While components/libraries are physically integrated in a software system, this is not the case for services, which run on an independent infrastructure and evolve under the sole control of the provider.

D. Accommodate and Dynamism:
   For traditional systems we always able to determine the component invoked in a given call-site, or, at least, the set of
possible targets. This is not true anymore for SOA, where a system can be described by means of a work flow of abstract services that are automatically bound to concrete services retrieved by one or more registries during the execution of a work flow instance.

E. Effect on testing cost:
If we Invoke actual services on the provider's node then has effects on the cost of testing is too, if services are charged on a peruse basis.

VI. WHY USING THE REGRESSION TESTING
Regression test helps to identify changes between a selected product release and a previous release of the product-called a baseline. A baseline is recorded snapshot of desirable product behaviour. This expected behaviour is then used to ensure that nothing has been broken in the system as a result of changes introduced in a program module. Establishing a regression testing framework is crucial for building reliable and stable software products [2].

Regression testing is usually performed by running some, or all, of the test cases created to test modifications in previous versions of the software. Three metrics, size reduction, precision, and recall, are reason to use this testing method. We first select tests from the regression suite that execute any of the modifications in the old program and refer to this technique as a modification-based test selection technique. This includes tests that have to be used for revalidation. Then, depending on the available resources, a trade-off between what we should ideally do in regression testing and what we can afford to do is applied to determine which tests, among those necessary, should be re-executed first, and which ones have lower priority or are to be omitted from re-execution.

VII. DIFFERENT TESTING TOOLS
A. TestMakertexting tool:
It is freely available tool and it is developed at the Department of Industrial and Organizational Psychology at the RWTH Aachen. It is web-based software for presentation, administration and evaluation of psychometric tests. First and foremost testMaker is tailored to the needs of Web-based self-tests with performance feedback, but it can also be used in other online surveys. Using testMaker neither programming nor HTML knowledge is required.

B. Soatest:
Soatest is a testing tool suite for testing and validation in a Service Oriented Architectures. In this testing tool Basic testing functionality include static analysis through WSDL testing, functional unit testing, regression testing, security testing, and load testing.

C. SoapUI:
SoapUI is an open source web service testing application for service-oriented architectures (SOA). Its functionality covers web service inspection, invoking, development, simulation and mocking, functional testing, load and compliance testing. A commercial version, SoapUI Pro, which mainly focuses on features designed to enhance productivity, was also developed by eviware software.

D. E - Test:
Suite for Web services provides ways to generate Web services test scripts, validate XML responses, and identify performance bottlenecks by server-side monitoring.

E. IBM Rational tester for SOA Quality:
A testing tool which is used for the various types of testing of SOA based application. The testing of the application is done through generating the script for various actions. This tool significantly reduces the time and effort required for the SOA based application. A functional and regression testing tool that enables code-free testing of GUI-less web services [9].

VIII. FUTURE REASEARCH ISSUES
In model based regression testing the Identification of modification is quite difficult and little bit time consuming for testers. Therefore our future research we will be focusing more on the development of the standard approach which will be easy to implement and will take care of the issues of identification of the modification of changes in the system.

IX. SCOPE OF THE SURVEY
To the best of our knowledge no survey work has been published on Regression Testing of SOA. Hence our focus in this paper is to present a comparison of the various testing methodologies and pros and cons of those approaches. The survey will help to develop an automated testing approach which would be more relevant considering scenario and challenges posed by SOA based application.
X. CONCLUSION

Regression Testing in Service-Oriented Architecture is one of the important testing activities which require a lot of effort to test the system because of the dynamic nature of the system. In this paper we made an effort to put the different issues involved with Regression Testing and the analysis among different approaches which will be helpful for the automation of the SOA based Regression Testing and by using the model base regression testing we give an approach to test the Service-Oriented Architecture applications.

REFERENCES


