SOFTWARE ARCHITECTURE BASED REGRESSION TESTING

Harsh Bhasin
Delhi technological university, Delhi
i_harsh_bhasin@yahoo.com

Ankush Goyal
AITM, Laboratory, Address,
Palwal, Haryana, 121102, India
Ankush49892@gmail.com

Deepika Goyal
AITM, Laboratory, Address,
Palwal, Haryana, 121102, India
Deepika.goyal256@gmail.com

Abstract

Software architecture plays a significant role in development of a dependable system. The purpose of regression testing is to make the system fault tolerant. The amalgamation of these two, results in the development of a robust system. The earlier works uses the conformance technique to instill confidence on implemented system with code, architecture and behavior but has not considered many parameters. The present work includes the concept of software architecture, system behavior and regression testing to propose a new framework which is sure to reduce gaps in the present frameworks and thus improve the system reliability.

***Keywords*:** Software Architecture; Dependable Systems; Regression Testing; Architecture-Based Analysis and Testing.

1. Introduction

The advent of object oriented languages and newer design methodology brought concepts like data abstraction to the fore-front of software development. The concepts initially labeled as obscure, later went on to become the crux of the development process. They also helped in clearly defining the components and depicting the interaction between them. This was important as many studies blamed these interactions as the major cause of failures [1]. To understand the system, we need to understand the parts of system as well; though abstraction hides the inner details, the communication between units remains essential to bring out the errors.

As many studies suggested, testing the system on the basis of test cases generated by unit testing, does not always reliably test the overall system. To install the confidence in the robustness of the system, it is essential to have test cases which are better than the test cases of unit testing. Betterment is defined in terms of testing the flow of data from one unit to another. As per the literature review, the type of tests used to test the system before 2000 were majorly based on the behavior of units rather than the system. Therefore, such methodologies were inapt to handle interaction problem. In such situations software architecture comes to our rescue. Software architecture helps us to detect problems which cannot be detected via conventional tests. As discussed earlier the designing of software plays a pivotal role in helping achieve the above. The system level and these segment level abstraction helps to untangle the knots created due to faults in communication.

As per the review carried out, software architecture helps in creating early test and hence reduces the cost of testing. The work presented intends to amalgamate the virtues of software architecture along with the concept of regression testing to propose a technique which is robust and better then the techniques proposed till now.

The paper is organized as follows. The second section of the paper presented the concepts of the literature review, the third section presents the premises of the paper and its goals, the fourth section proposes the technique. The last section presents the conclusions.

2. Related work

An extensive review was carried out in order to understand the intricacies of the techniques which we wish to use. The following section gives a brief overview of the techniques studied.

According to Muccini [1], testing is needed to increase the dependability of the system. The previous techniques have shown the application of conformance testing to achieve the confidence on the implemented system in their expected behavior and architecture level. The work by H.Muccini [1] applied regression testing at software architecture level to reduce the cost of retesting the modified system. It may also be noted that, SA based analysis methods can be used in various scenarios like deadlock detection, performance analysis, component validation. SA based testing method check conformance of the implementation behavior and compare it with SA level specifications of expected behavior. Muccini [1] provides a technique to reuse previous information to get conformance of modified implementation with respect to the modified or initial architecture.

According to work by D.S. Rosenblum [11], The regression testing for analyzing the system during its life time is very expensive. The interaction between the components, however, can help us to reduce this cost. For a software system, the aim of Selective regression testing strategies to choose subset of test cases from previously run test cases, based on information about the changes made to the system to create new versions. In the paper some computationally efficient predictors of the cost-effectiveness of the two main classes of selective regression testing approaches are presented. In the work proposed by Mary Jean Harrold [2], which is based on the specification of software architecture, it was observed that the amalgamation of SA proves very effective in terms of cost and time. The software architecture testing focuses on the cost of the software while other techniques focus only on development.

Another work by M. J.Harrol [2], provides an approach that uses the integration of code level regression testing with architecture based regression testing. It uses the selective testing for the architecture based regression testing and code level regression testing. It is based on comparing nodes of the two graphs, where the first graph represents the program and the second one represents the modified version of original program. The work provides a novel approach for regression testing at architecture level by comparing both graphs.

The work by A. Bertolino [14] also affirms the use of Software Architecture in testing. The work also confirms the effect of SA on implementation. The paper proposes the extended approaches to SA bases testing. It shows how a architectural style conform the mapping among SA based and code based test cases. According to the work, Software Architecture can be used for code conformance testing and to check if implementation fulfills to its specification at the SA level. This paper extends the previous approaches to software architecture based testing and how a specific architectural style which supports implementation and facilitates the mapping among SA-based and code-based test cases can be used to deliver a completely systematic SA-based testing approach.

3. Background

3.1. *Premises of the paper*

The proposed technique is based on the concept of Muccini [1]. In the work, Software Architecture Specification (SAP) has been taken as base and the behavioral model of the software serves as a test oracle. As per the above work, topology is described in terms of components, connectors, and configurations. This is followed by the application of SA behavioral.

The above is followed by seeing SA in a way so that non relevant actions are hidden. This helps in the abstraction of state machine based model. This generates what is referred to as, Architecture Level Test Cases (ALTC), which is based on the audit sequence of events. So as to accomplish the above task the mapping function is used which maps SA level function tests to code level tests.

3.2. *Goals*

The goals that are to be accomplished by the proposed technique are as follows:

The first goal is to use the existing implementation-level test cases test the conformance between modified code and the architectural specifications. It is to test the conformance of a program with respect to the system, while reusing previous test information for selective regression testing, thereby reducing the test cases.

The second goal is to reuse architecture-level test cases to test the conformance of the source code with respect to the evolved software architecture.

The first goal is accomplished by generating a control graph and comparing the previous graph with the new graph. So as to facilitate the task, the information so as to how a graph is traversed is stored. Test cases selection for the new program P’ with the help of test history and graph comparison, is then carried out. The concept relies on integrating code-level regression testing with architecture-based regression testing. Selective testing technique is used for code-level regression testing and for architecture-based regression testing also. The test cases are generated on the basis of software architecture. The expected behavior and the history is then mapped with the output. In the conformance testing we generate graphs for both the program and the changes in the programs. On the basis of comparisons between the old graphs and new graphs, the changes are recorded. The techniques developed earlier are not suitable because it is possible that changing the program might not result in the change in the graph. There can be many more such scenarios. If there is a change then that change may affect the software architecture but not the behavior. Moreover, it is also possible that change in one part of the program may effect on other part of program due to coupling [15]. Previous technique has neglected the above issues.

The work intends to, merge software architecture concepts with a concept of regression testing to propose a novel technique. The technique is being tested by a set of programs. In the proposed technique, the software architecture of the program will be studied and the purposed technique will be used to find errors. The concept will be compared with a technique purposed in the work by Muccini [1]. A set of 20 programs is selected, divided into three categories small, medium and large programs.

4. Purposed Technique

The section proposes a new framework which clubs together the best of worlds, regression testing and software architecture. The proposed work is different from the base work since it also takes into account the behavior and not just the abstract model.

4.1. *Software architecture based regression testing with behavioral blent*

Step1: SA specification. SA based conformance testing start from behavioral specification of SA and topology of SA. In the SA structure topology describes the components, connectors and configurations. For describing the topology we use ADL (Architecture description languages) and TS (transition system) is used for describing the SA behavior.

Step2: Testing Criterion. There are many events in model and sequence of these events is defined as ATC (architecture-level test cases). Designing architecture level test cases calls for checking the specifications and also checking the behavior of the model. In this step the portions, which are to be tested, are identified. The interactions between the levels is also seen.

Step 3: Test Cases. Changing or adding an extra component can change the behavior. So, a mapping function is to be used for mapping the SA-level test case to code level test cases. This also checks the behavior of the model.

Step 4: Test Execution. Since many test case have been generated, now the next step is to check the result of each test case and map the corresponding results.

The model is depicted in the following figure.

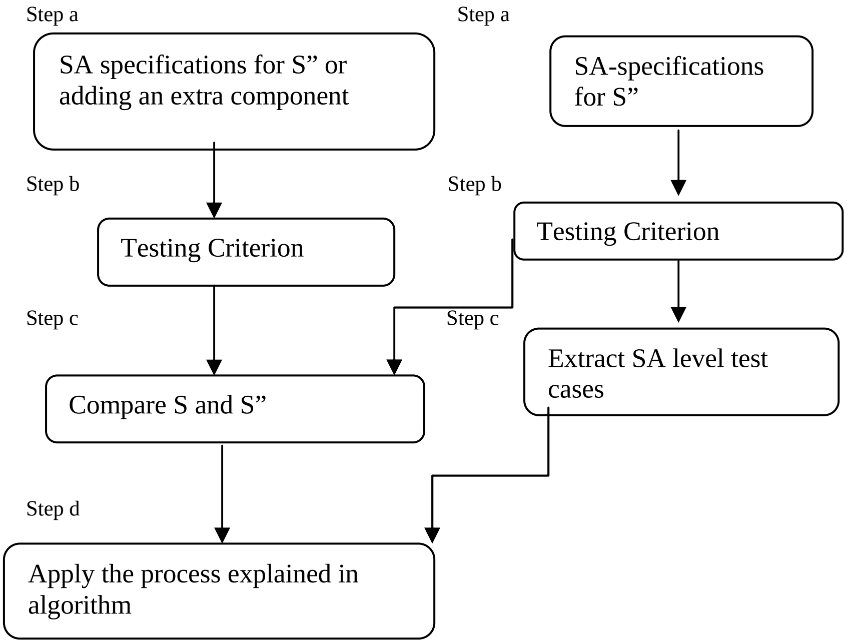
Step a
Step a
SA specifications for S" or adding an extra component
SA-specifications for S"
Step b
Step b
Testing Criterion
Testing Criterion
Step c
Step c
Compare S and S"
Extract SA level test cases
Step d
Apply the process explained in algorithm

Fig. 1 Diagram of SA based regression testing

5. Conclusions

The work presented above adds a sprinkle of behavioral model to the Software Architecture based Regression Testing. The technique is being applied on selected set of programs. The programs have been selected in such a way that the technique can be verified by all the categories. The programs selected therefore are professional applications and some algorithm implementation. The application of the above technique to these programmers will instill the confidence on the technique.

It may be noted that the above work re-affirms the fact that the exclusion of behavior from the model, is not justified. The complete verification and validation will instill the confidence in the technique.

References

[1] Henry Muccini, Marcio Dias, Debra J. Richardson. Software Architecture-Based Regression Testing, 10 February 2006. In: The Journal of Systems and Software 79 (2006) 1379–1396.

[2] M. J. Harrold. Architecture-Based Regression Testing of Evolving Systems. In Proc. Int. Workshop on the Role of Software Architecture in Testing and Analysis (ROSATEA), CNR-NSF, pp. 73-77, July 1998.

[3] An Andreas Johnson. Architecture –Based Verification Of Software- Intensive Systems. Mälardalen University School of Innovation, Design and Engineering 2010 March 8,Västerås Sweden.

[4] Henry Muccini, Marcio S. Dias, and Debra J. Richardson. Towards software architecture-based regression testing. SIGSOFT Software. Eng. Notes, 30(4):1_7, 2005.

[5] Badri H.S. Systematic Software Architecture Based Testing Approach – A Case Study. In International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 9, September 2012.

[6] B. Beizer. Software Testing Techniques. Van Nos-trand Reinhold, New York, NY, 1990.

[7] A. Bertolino and P. Inverardi. Architecture based software testing. In Proc. of Int'l Software Architecture. Workshop, pages 62{64, October 1996.

[8] H. Leung and L. White. A cost model to compare regression test strategies. In Proc. of Conf. on Software. Maint. Pages 201{208, Oct. 1991.

[9] D. Richardson, J. Stafford, and A. Wolf. A formal approach to architecture based software testing. Technical report, University of California, Irvine, 1998.

[10] D. Richardson and A. Wolf. Software testing at the architectural level. In Proc. of Int'l Software. Arch. Workshop, pages 6{70, October 1996.

[11] D. S. Rosenblum and E. J. Weyuker. Predicting the cost-effectiveness of regression testing strategies. In Proceedings of the ACM SIGSOFT '96 Fourth Symposium on the Foundations of Software Engineering, Oct. 1996.

[12] H. Muccini, M. Dias, and D. Richardson. Systematic Testing of Software Architectures in the C2 style. Extended version of the ETAPS 2004 publication.

[13] A. Bertolino and P. Inverardi. Architecture-based software testing. In Proc. ISAW96, October 1996.

[14] A. Bertolino, P. Inverardi, and H. Muccini. An Explorative Journey from Architectural Tests Definition down to Code Tests Execution. In IEEE Proc. Int. Conf. on Software Engineering, ICSE2001, pp. 211-220, May 2001.

[15] Harsh Bhasin, Manoj. Regression Testing Using Coupling and GeneticAlgorithms.IN (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 3 (1), 2012, 3255 – 3259

[16] Perry, D.E., Wolf, A.L., 1992. Foundations for the Study of Software Architecture 17 (4), 40–52.

[17] Muccini, H., Bertolino, A., Inverardi, P., 2003. Using software architecture for code testing. IEEE Transactions on Software Engineering 30 (3), 160–171.

[18] Bosch, J., 2000. Design and Use of Software Architectures: Adopting and Evolving a Product-Line Approach. ACM Press/Addison-Wesley Publishing Co.

[19] Bernardo, M., Inverardi, P., 2003. Formal Methods for Software Architectures, Tutorial Book on Software Architectures and Formal Methods. SFM-03: SA Lectures, LNCS, vol. 2804.