Automatic Generation of Regression Test Cases for Web Components using Domain Analysis and Modeling

Gagandeep
Deptt. Of Computer Science
Punjabi University
Patiala, India

Dr. Jyotsna Sengupta
Deptt. Of Computer Science
Punjabi University
Patiala, India

ABSTRACT
Web components and their underlying parts are evolving rapidly to provide services and information over the internet. Web sites are becoming a basic part of business quality of service. As web services play an important role in providing different functionality, their correctness becomes an important factor. The diversity and complexity of web systems derived the need for test automation. Test Automation is the use of software for automatic execution of test cases and their comparison with expected outcome. In this paper, we present an analysis based modeling and regression testing technique for web applications. Model based testing is a test automation approach that generates and maintains more useful and flexible tests from explicit descriptions of the application. Models describing web components structure before and after the changes are incorporated. It helps to identify test cases generated for the initial version of component that can be re-run on the changed component. The regression test-suite generated for the component is optimized using "all-path" coverage criteria. The proposed approach yields substantial results with optimized test data generation.

General Terms
Component, Regression Testing, Modeling

Keywords
Web Components, Domain Analysis, Test Automation

1. INTRODUCTION
In today's scenario, with the development of internet and intranet technology, the Web applications become more and more complicated with strict requirements of correctness, quality, reliability and inter-operability. The development cycle of web component differs a lot from the traditional software component; as web applications consist of static as well as dynamic pages which require a lot of interactions between users and Web browser. Thus, testing is an important means to ensure quality of Web components. Different testing techniques such as Unit testing, Integration testing and Regression testing are important for Web based application. Unit testing is used to test an individual Web application whereas Integration testing is used to test Web application in conjunction with other components. Web applications evolve over time therefore, they need to be retested whenever there is a change to ensure components correctness and stability, thus assuring software quality. Hence, Regression testing needs to be carried out.

In this paper, we focus on regression testing of web components. We introduce a notation for domain analysis and representation of Web component which is traversed to generate test sequences. The test sequences are further optimized using optimization criteria. We start with a survey of related work carried out in testing of web components. There after we provide basic concepts and technical background necessary for regression testing strategy of Web components based on Domain analysis and modeling. A case study to demonstrate testing approach is discussed and results are validated.

2. RELATED WORK
Modeling and testing a web application is more complex than the traditional application. A large number of researchers have worked in the area of web applications testing. Each researcher has focused on different areas of web applications like static and dynamic pages, links and frames, architecture, model and scenarios.

A number of challenges to test web applications have been discussed by Hieatt and Mee [6]. They proposed writing a unit test before the new code will generate independent pieces of code. Moreover comments can be given inside the test cases which will enhance their understandability. Briand and Labiche [3] proposed a testing method for automatic generation of test cases based on UML. Andrews et al.[1] proposed a hierarchical approach to model potentially large web applications. It incorporates system-level testing based on finite state machine representation of Web application. Gutierrez et al. [4] studied how to generate test cases using use cases for Web applications. They used activity diagrams to represent the behavioral model of the system's requirements.

Huang et al. [5] proposed a software tool, named WASATT (Web application scenario automated testing tool) that facilitates automated testing for scenario of Web-based applications. It uses activity diagram model to represent functional scenario of web applications which is transformed to testing code. Qian et al. [10] developed a web testing model for web applications. It constructs a Page Flow Diagram of the web application, which is translated into a Page Test Tree to generate test cases. An approach to web application regression testing based upon repair of user session data is discussed [2]. Kam and Dean [8] conducted a survey of different testing techniques used for testing web applications. They reached to the conclusion that a single testing technique cannot test all the bugs of a web application. Some methods can test only controlled flow of the transactions while others can handle uncontrolled flows. Many techniques need to be integrated with some other techniques to fully validate and verify a web application.

B. Zhu et al.[13] formulated the use of a object-z formal language to analyze formal specifications to identify preconditions required to construct environment Transition Graph (ETG). Test sequences are generated that satisfy test adequacy

criteria. Zhongsheng Qian [12] proposed Component Interaction Diagram (CID) based on specification of the web application. Component Test Sequences are generated form their traversal. Another work based on graph theory approach known as maximum network flow for generating minimum number of test cases is proposed [11]. They made use of web navigation graph to model the system and to generate optimal test cases. Lain Yu et al. [9] proposed a new technique for user acceptance testing of web applications. It is different from traditional methods as it includes beta-testing which is carried out by end-users. A methodology to automate generation of test cases based on Model Driven Architecture is proposed by Yang Liu et al.[14]. A sytem model and a test model is designed using Web-application system and test cases are generated using model to code conversion method. Bo Yang et al. [4] proposed regression testing method for composite web services. The tester can find the faults and make test data and test behavior independent of each other.

3. IMPLEMENTATION FRAMEWORK

Web Applications need to be analyzed and modeled prior to testing. The web model is traversed to generate test sequences. We have worked upon "all path" coverage criteria to generate optimized test sequences. The proposed framework consists of the following steps.

1. Domain Analysis and Modeling
2. Model traversal and test case generation
3. Optimizing test cases using coverage criteria
4. Regression test suite generation

Initially we start with detailed analysis of a web application and explore specifications of its different parts. Complete study of the web application need to be done and creation of a manual document is performed for all type of functions and is termed as Specification Document. It is the theoretical explanation of the whole functionality (links, buttons, database interaction etc.) of the web application. In this document, type of page: static or dynamic, title of the page, number and label of links/buttons on the page and the transitions is given. A page id has also been assigned to each page to provide easiness in using the pages implementation.

From Analysis, we move to design representation of the web-application. We adopt Model–driven representation for designing and implementation. Model-driven approach offers a new and promising approach for the task of designing and automation .With the help of Specification document, a Graphical Web Model of the component is constructed. The model is traversed using 'All Link Coverage' to generate test sequences. By using this criterion we confirm that all the links available in the web application will be tested at least once.

Definition: Web Application Test Sequence WTS: T0→T1→T2→... →Tm is a sequence starting from root page to some leaf page that can be traversed, T0 is the initial page and 'm' represents total number of Test Sequences where (m>0). WTS represents available navigation from page T0 to Tm by traversing different intermediate pages.

The last step of the framework is to generate regression test suite. The modified version of web application may incorporate or delete some pages or links. Once again it goes through Domain Analysis and Modeling activity. Optimized set of test sequences are generated in such a way so that all the links are covered. Test sequences following the same traversal path that differs only at the end points are merged.

For example, if first test sequences is WTSi: T1→T2→T3→T5→T6 and second one WTSj: T1→T2→T3→T4 then we can merge these two test sequences as T1→T2→T3→T4→T3→T5→T6.

To reduce the number of test sequences we can follow up another coverage criterion i.e. "All Page Coverage". According to this coverage criteria, remove all the copied pages which are the leaf nodes and links corresponding to these copied pages from the web model. By doing so, the length of the test sequences will automatically be reduced as it ensures that all the pages will be tested at least once.

4. CASE STUDY

To demonstrate the functionality of approach, we consider login system of web application and name it as Web Sign Up and Login System (WSL). It is analyzed and Graphical Web Model (GWM) is constructed, which is traversed to generate test sequences. The effectiveness is finally shown by incorporating changes and generating Regression Testing Suite. We initially considered a static page which further contains three links namely Login, Sign up and Search. Login page is static and contains one button namely, Submit. Logged in page contains two links Read and Log out. The Information page is dynamic and contains two links – Continue and Search. Finally Logged Out page contains three links Home, Login and Search. The complete web component is modeled and is represented through Graphical Web Model shown in Figure 1.

Test Generation and optimization: Algorithms are written and implemented in Visual C++. The model is implemented as a queue data structure and test sequences are generated with the help of list which is implemented with the help of 'Arraylist', an inbuilt class in Visual C++ which acts like list class. The output as test sequences and their statistics are shown in Figure 2 The process of optimization takes test sequences generated on the previous stage as its input and the output is the optimal test sequences itself

Regression Test Suite Generation: Some changes are incorporated in the web application to create a newer version of the component. Figure 3 shows Graphical Web Model of the component. After the generation of optimal set of test sequences we make comparisons with the already available optimal test sequences that are fully usable before modifications. The result of this comparison gives the old optimal test sequences which can be reused, discarded, updated and new added test sequences.

The regression test suite is generated for this component after its traversal and comparison with web model of the initial web component. Figure 4 shows test sequences for modified web component along with the new and updated test sequences.

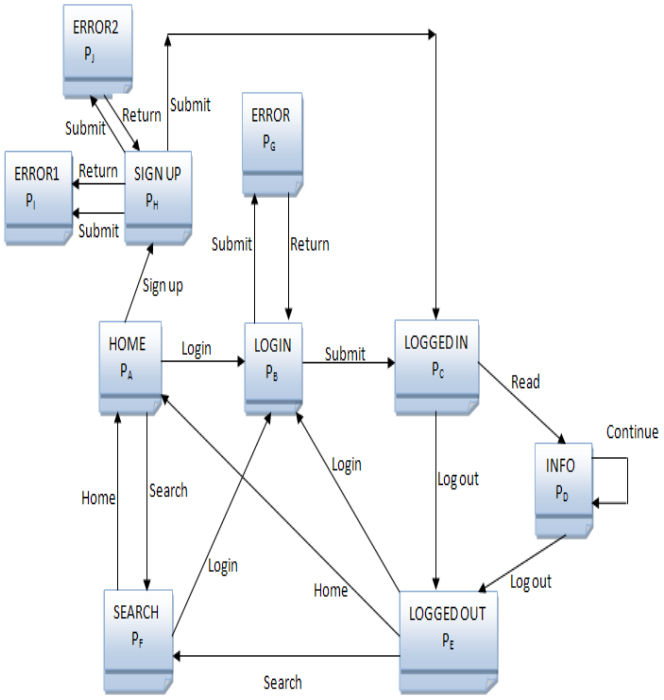

Figure 1. Graphical Web Model of Component

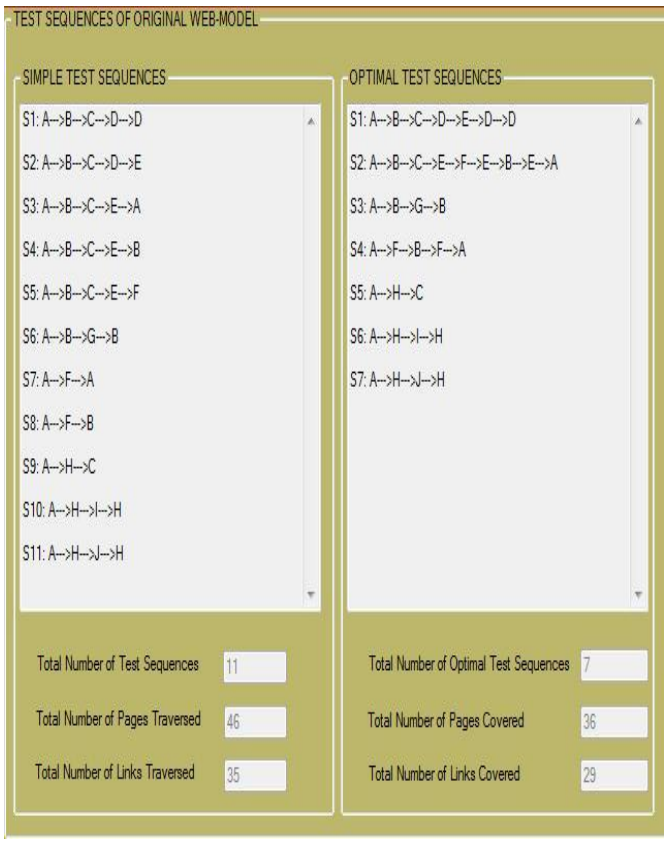

Figure 2. Optimal Test Sequences of web component

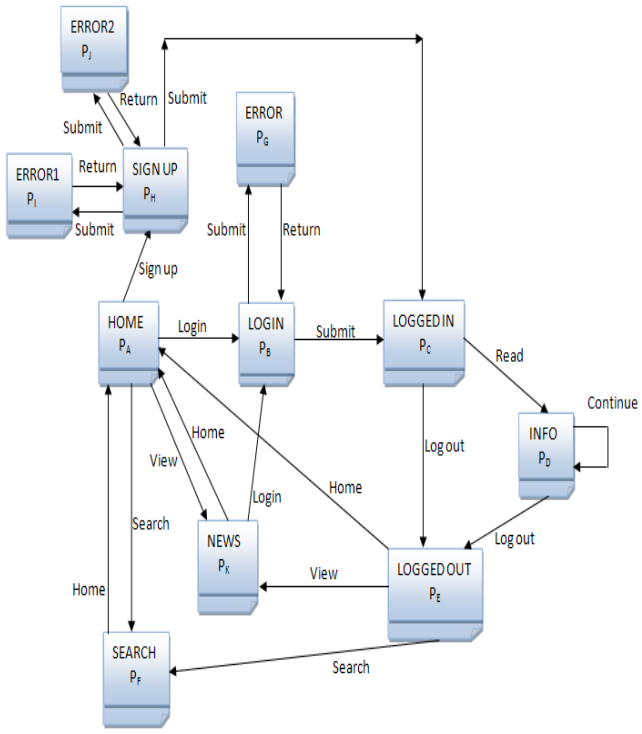

Figure 3. Graphical Web Model for changed web component

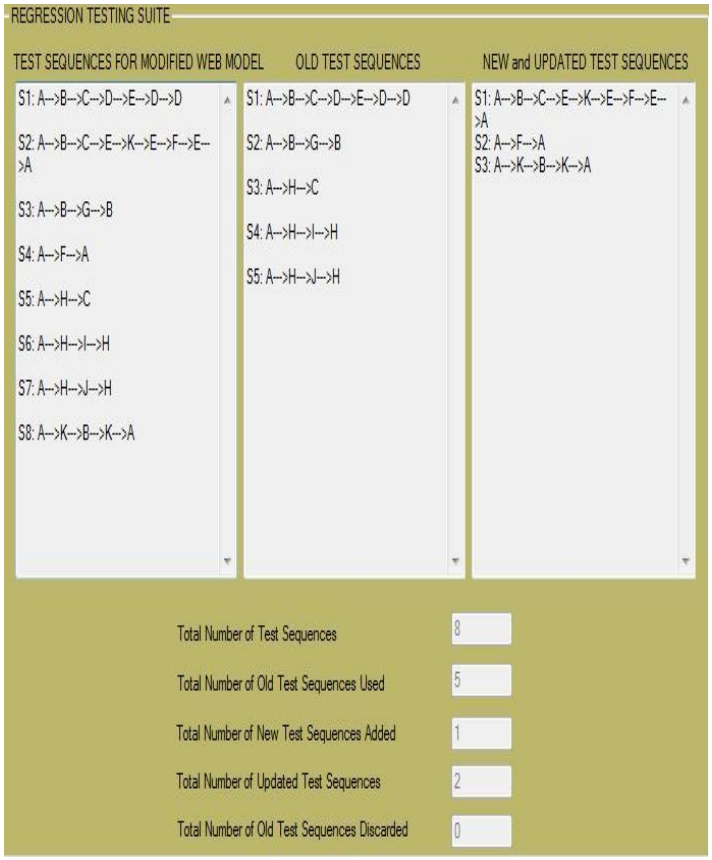

Figure 4. Regression Testing Suite for changed component

5. CONCLUSION

We adopted domain analysis and modeling technique for testing web applications that is based on model driven architecture of the system. This approach significantly reduces the effort and cost of rigorous cycle of software development and testing process, as regression test suite can be efficiently generated while maintaining high standards of software quality. We proposed the method, design and implementation of generation Graphical Web Model from the analysis of web application. A web application system has been tested as the research object. We further tried to automate and optimize test generation process. A Graphical Web Model for the system has been generated and implemented as web page model. All- path coverage criteria has been followed to generate optimal test cases.

Future work includes automatic generation of Graphical Web Model from analysis of web component. Secondly, the complexity of the algorithm needs improvement. Further, work needs to be done to identify link and performance constraints on web graph model

6. REFERENCES

[1] Andrews, A., J. Offutt, R. Alexander, 2004. Testing web applications by modeling with FSMs. Software Syst. Modeling, 4: 326-345.

[2] Alshahwan, N. and M. Harman, 2008. Automated session data repair for web application regression testing. Proceeding of the International Conference on Software Testing, Verification, and Validation, April 9-11, Lillehammer, pp: 298-307. 10.1109/ICST.2008.56

[3] Briand, L. and Y. Labiche, 2002. A UML-based approach to system testing. Software Syst. Modeling, 1: 10-42, DOI: 10.1007/s10270-002-0004-8

[4] Bo Yang, Ji Wu, Chao Liu and Luo Xo (2010). A Regression Testing Method for Composite Web Services. Proceedings of the International Conference on Biomedical Engineering and Computer Science (ICBECS), 23-25 April 2010, pp. 1-4, DOI 10.1109/ICBECS.2010.5462461.

[5] Gutierrez, J.J., M.J. Escalona, M. Mejias and J. Torres, 2006. An approach to generate test cases from use cases. Proceedings of the 6th International Conference on Web Engineering, July 11-14, Palo Alto, California, USA, pp: 113–114.

[6] Hieatt, E. and R. Mee, 2002. Going faster: Testing the web applications. J. IEEE Software, l: 60-65. DOI: 10.1109/52.991333

[7] Huang, C.H., H.Y. Chen and J.U. Guangzhou, 2006. A tool to support automated testing for web application. Proceeding of the IEEE International Conference on System, Man and Cybernetics, Oct. 8-11, Taipei, pp: 2179-2184. DOI: 10.1109/ICSMC.2006.385184

[8] Kam, B. and T.R. Dean, 2009. Lessons learned from a survey of web applications testing. Proceedings of Sixth International Conference on Information Technology: New generation. April, 27-29,LasVegas, NV , pp: 125-130. DOI:10.1109/ITNG.2009.306

[9] Lian, Yu., W. Zhao, X. Di, C. Kong ; W. Zhao, Q. Wang and J. Zhu (2009). Towards Call for Testing: An Application to User Acceptance Testing of Web Applications. Proceedings of 33rd Annual IEEE International Vol. 1, pp: 166 - 171 , DOI 10.1109/COMPSAC.2009.31

[10] Qian, Z., H. Miao and H. Zeng, 2007. A practical web testing model for web application testing. Proceeding of the Third International IEEE Conference on Signal-Image Technologies and Internet-Based Systems. Dec. 16-18, Shanghai, pp: 434-441. 10.1109/SITIS.2007.16

[11] Song, B., H. Miao, Z. Chen (2009). Modeling Database Interactions in Web applications and Generating Test Cases. Proceedings of World Congress on Software Engineering, May. 2009, pp. 216-221.

[12]] Zhongsheng Qian, 2009. Testing Component-Based Web Applications Using Component Automata. International Conference on Information Engineering. ICIE '09. Vol. 1, pp. 455 – 458, DOI 10.1109/ICIE.2009.64

[13] Zhu, B., H. Miao, H. Zeng, S. Chen (2009). Generating Test Case from Functional Requirement of Web Application. Second International Symposium on Electronic Commerce and Security, May. 2009, pp. 465-468.

[14] Yang Liu., Yafen Li, Pu Wang (2010). Design and Implementation of Automatic Generation of Test Case Based on Model Driven Architecture. Proceedings of the Second International Conference on Information Technology and Computer Science, pp. 344-347.