Web Application Testing: A Review on Techniques, Tools and State of Art

Arora A., Sinha M.

Abstract— Web applications are meant to be viewed by human user. While this implies that quality of web application has importance in our daily life. Web application quality is our prime concern. To ensure the quality of web application, web testing is having a dandy role in Software Testing as well as Web Community. Web Applications are erring because of features provided for rising of web application. In the last years, various web testing problems have been addressed by research work. Several tools, techniques and methods have been determined to test web application efficaciously. This paper will present the contribution of researchers in the field of web application in previous years and state of art of web testing and challenges primarily because of distributed and heterogeneous nature of web application.

Index Terms— Web application testing, software testing

1 INTRODUCTION

Testing is major component of any software engineering process meant to produce high quality application. Testing aims at finding errors in the tested object and giving confidence in its correct behaviour by executing the tested object with input values [11]. Web applications are the fastest growing classes of software systems today. Web applications are being used to support wide range of important activities: business transaction, scientific activities like information sharing, and medical systems such as expert system-based diagnoses. Web applications have been deployed at a fast pace and have helped in fast adoption but they have also decreased the quality of software. Therefore, all entities of web application must be tested. In order to make web based application to be widely and successfully adopted, testing methodologies must be flexible, automatic, and be able to handle their dynamic nature [3].

The testing of web based applications has much in common with the testing of desktop systems like testing of functionality, configuration, and compatibility. Web application testing consists of the analysis of the web fault compared to the generic software faults. There are various non-equivalence issues between traditional software testing and web application testing. Following are the some issues:

- Web applications typically undergo maintenance at a faster rate than other software systems.
- Web applications have a huge user population, thus propose a high demand to the server's performance and the ability of dealing with concurrent transaction. Moreover, when a large number of users access that at same time will have to be looked for web content rendering capability
- Unexpected state change like via browser back button or direct URL entry in the browser. A malformed URL in a dynamically constructed web page is a web specific fault, while an index outside the array bounds is a generic fault which can occur in any program regardless of any Web technology involved.
- Web applications should be tested to check its working on different types of web browser and running on different operation system.
- One main difference between traditional and web specific testing is architecture. In web application testing process, it is often difficult to pin point where the error occurs and in which layer because of web application multi tier architecture. Web based applications provide a variety of services, but are commonly build as three tiered architecture. Web application uses multi tier architecture for designing, developing and deploying component based, enterprise wide application.
- Web applications are able to render software components dynamically at runtime according to inputs given by user as well as on the basis of server response.

Filippo Ricca and Paolo Tonella presented a fault model [12], in this fault model, among the collected faults some can be classified as generic faults that can be found in almost every software system. On the other hand, other faults are strictly dependent on the interaction mode because of web application multi tier architecture. Some web specific faults are authentication problem, incorrect multi language support, hyperlink problem, cross-browser portability problem, incorrect form construction, incorrect cookie value, incorrect session management, incorrect generation of error page, etc.

- *Anuja Arora is currently pursuingDoctor of Philophy degree in Computer Science in Banasthali University,india, PH-9810982939. E-mail:anuja.arora@jiit.ac.in*
- *Madhavi Sinha is currentlyguiding in Doctor of Philophy degree in Computer Science*

Main characteristic of web application is that web applications are enormously heterogeneous in nature. Web application heterogeneous execution environment composed of different hardware, network connection, operating system, web services and web browser [16].Web applications includes large variety of software components that makes it heterogeneous in nature. All components can be constructed on different technologies (i.e., different programming language etc). Web application testing is a tedious task because of features provided by different technology to design an efficient and feature emerged application. Various technologies merged at one place affect testing complexity.

In this paper, main concern will be to bring up issues regarding testing web application functionality implemented using numerous technologies. Generating a test environment to test this type of application and exercise each of them is a quite tough task. Various testing methodologies and tools implemented to detect failure in the functionality, in order to verify the conformance of the application with the defined behaviour. Functional requirement testing of web application will be considered in this paper. So after discussing challenges of web testing in section two focus of the remainder paper will be on compendious of various existing testing techniques for web application in section three, discussed web testing tools in section four and state of art of web application testing for defined challenges mentioned in section five.Finally, Section six is brief discussion about future trends of testing scopes in web applications and concluding remarks.

2 WEB APPLICATION TESTING CHALLENGES

Testing of web application employing new technologies (like AJAX, Flash, Active X plug-in component, Structs, Ruby on rails, etc) is an area that has not been investigated so far. In this research work focus will be on web application testing because with the advent of these new technologies, novel testing problems raised and added to the list of already existing problems in web testing area. These novel problems turn out to be main sources of faults in web application. Web applications are fault prone because of stateful client, asynchronous communication, delta updates, untyped JavaScript, client side DOM manipulation, event handling, timing, back / forward button and browser dependence. Challenges of web testing because of embedded features of current web technologies are as follows:

2.1 State Navigation

State navigation was prime concern at the time of web testing. A process is required to fetch all dynamically updating states. State information may only be determined dynamically through event-driven changes in the browser's DOM. It is difficult to find changes in before and after events also because of various reasons like the entire page does not repaint, users may not perceive that anything has changed, address bar does not change even if the page changes. If users expect the back button to work using AJAX web application, it is difficult to manipulate changed parts of the page. Web applications run time manipulation creates difficulty in fetching all states and their behaviour for testing.Problems may arise in detecting all dynamically generated and updated states.

2.2 Transition Navigation

Testing of methods triggered by user events or server message and modifying the DOM is also a tedious task. With regard to transition, Marchetto [21] suggested that method invocation triggered by user events or server messages can affect DOM states. All other method invocations have no effect on the DOM state, so can be ignored [20, 21]. In Transition testing, identify set of method reacting to events and possibly affecting the DOM can be possible through static code analysis. The output of this analysis determines the set of methods that need to be traced. DOM state is logged after the invocation of each such method. Process of assessing the correctness of test case output is a challenging task because Static analysis will miss complex run time behaviour and when state space is huge, it becomes quite tedious task. One example of dynamic Dom manipulation is that the individual sections are editable right on the main page, and to customize the page, one can simply grab them with their mouse and drag them to their new location. So this type of DOM behaviour makes testing problematic.

2.3 Delta server message

Delta-Server messages [10] from the server response are hard to analyze. Most of such delta updates become meaningful after they have been processed by the client side engine on the browser and injected into the DOM. In testing process, retrieving and indexing the delta state changes from the server. Delta states can be retrieved only through proxy between the client and the server and this could have the side-effect of losing the context and actual meaning of the changes. That is why delta state testing is really a challenging task. Most of such delta updates become meaningful after they have been processed by the JAVA SCRIPT engine on the client and injected into the DOM [26].

2.4 Asynchronous behaviour

Web application nondeterministic behaviour because of asynchronous behaviour is also a great testing challenge. This nondeterministic behaviour can be because of network delays, asynchronous client/server interaction, non sequential handling of requests by the server, randomly produced or constantly changing data on real time web application. Some problems of asynchronous behaviour are swapped call back. Assume that in swapped call back there are two semantically interacting events e1, e2.Let r1 and r2 be the associated request sent to the server and let c1, c2 are corresponding call backs. The following execution sequence may occur: <r1, r2, c2, c1>. In this sequence c2 starts before c1. It produces an incorrect final state. The reasons of c2 starts before c1 may be network delays, scheduling of threads on the server(second thread terminates before first starts), scheduling of callback activation on client(second callback scheduled before first one), etc. Other problem is dependent request. So, all the asynchronous communication problems are problems for current web appli-

cation testing also. This asynchronous challenge may reach to an incorrect final state or some output values may be different from the expected ones [21].

2.5 Stateful behavior

The biggest problem with web applications is saving state and accommodating the familiar progression of the history controls (Back/forward buttons). AJAX web application technology allows the document to become stateful, but when the user instinctively goes for the history controls in the browser, a fault often occurs in AJAX the broken back button of the browser. A dynamically changed state does not register itself with the browser history engine[10].Stateful behaviour is a challenging task because states itself contains many information and state behaviour change because of content inside that. A method is required to test content of states and validate changes on the basis of content of a particular state.

3 WEB APPLICATION TESTING TECHNIQUE

Marchetto [20] discussed in his work that existing web testing techniques are not suitable appropriate to test the specific characteristics with respect to AJAX. Similarly, for other current web technologies also existing web testing techniques are not appropriate. However, we summarize long familiar effective web testing techniques, which are diffused in current web testing scenario.

3.1 White Box testing

White box testing design test cases on the basis of code representation of application under test. To traditional software, white box testing of web application is based on internal structure information of the system under test. White box testing approach has applied to web applications using two main families of structural models. Either on the basis of level of abstraction of code of the application or using navigation model between pages of application. Various web testing techniques has been introduced under this category. The White box technique proposed by Ricca and Tonella is Model based testing technique [1,12,13,23]. Mainly Model based techniques uses reverse engineering and web crawling techniques to build a model of a web application.

Navigation model based testing built a model using graph in which each node is a web page and edge is a link. Limits of this navigation model are that it does not test asynchronous behaviour and dynamic changes of a web application. This navigation model does not consider response of a request, does not includes the states that a HTML page can reach during application execution. This approach cannot dynamically analyze the whole web application structure.

Code coverage based testing [20] follows primarily two testing methods- object based data flow testing and Control flow model based testing. Object based data flow testing [14] client tier interaction behaviour not server tier interaction behaviour of a web application. Object based data flow model captures the data flow information of web applications and consist of two models- object model and structural model. Object model component are modelled as objects that contain attributes and operations and structural model captures the data flow information of functions within or across objects. In this four types of graphs are employed- Control flow graph, interprocedural control flow graph, object control flow graph, composite control flow graph. Restriction in this approach related for testing web application is that the client server request, response, navigation and redirect relationship not representing in object model and even not testing Extreme dynamism of web application. Control flow based testing uses reverse engineering and web crawling techniques to build a test model of a web application. In this techniques nodes represents statements that are executed by a web server and edges represent control transfer. This technique can be applied to web application with alteration like change in technological nature of coverage tool. Coverage tool should support and trace web code of mix of web application technologies like: HTML, Java Script, JSP and AJAX etc. This approach appears to be partially adequate due to high dynamicity of web applications and asynchronous behaviour of web application nowadays. These days web applications are designed using DOM element during execution and adds a dynamically constructed callback to it. Callback cannot be traced using this approach.

Other than these techniques logging user session data on the server is also used for the purpose of automatic test generation [3,25].This requires sufficient interaction of real web users with the system to generate the necessary logging data. Session based testing are merely focused on synchronous requests to the server and lack of complete state information required in AJAX testing.

3.2 Black Box Testing

Black box testing is to generate test cases on the basis of mentioned functionality of the system under test. This testing technique does not check code structure and implementation of system under test.

Main issue with black box testing is the use of suitable model for specifying the behaviour of the web application to be tested. Black box testing approach proposed by Andrew is Finite State Machine (FSM) for generating test cases from web application. This approach takes state dependent behaviour of web application in consideration and derive test case from them [13]. Andrew proposed a system-level testing technique that combines Test Generation based on Finite State Machines with constraints[13]. The approach builds hierarchies of Finite State Machines (FSMs) that model subsystems of the Web applications, and then generates test requirements as subsequences of states in the FSMs. Several methods for deriving tests from FSMs have also been proposed [4, 13, 15, 17]. The constraints are used to select a reduced set of inputs with the goal of reducing the state space explosion otherwise inherent in using FSMs. Web applications can be completely modeled with FSMs, however, even simple Web pages can suffer from the state space explosion problem. So, web application behaviour depends on state of data managed by application and user input, with the consequence of state explosion problem. For resolving this problem, various solutions are investigated and presented in the literature. Sachoun Park presented a method for avoidance of state explosion problem using depen-

dency analysis in model checking control flow graph [22]. The fFSM is a model for describing the control flow aspects. fFSM, like a Statecharts, supports concurrency, hierarchy and global variables. In this paper presented the model reduction technique based on dependency analysis to avoid the state explosion problem. There are two more solutions available to solve this problem. First solution given by Di Lucca[16] that exploits decision table as a combinatorial model for representing the behaviour of web application and generating test cases. Second solution proposed by Andrew that model state machine using state dependent behaviour of web application and generates test cases.

Second approach is user session based testing approach suggested by Elbaum[3]. User session based testing collect user interaction and transforms them in to test cases. Data to be captured include clients request in form of URL's and apply strategies to these generated test cases. User session based testing having many advantages over white box testing technique. Advantages are as follows: (a) User session based testing generates test cases without analyzing the internal structure of the web application that reduce cost and time of finding inputs, (b) less dependent on heterogeneous and distributed web application technologies, (c)user session based testing depends on data collected. This technique will provide efficient result for wider user session data set. The tedious task of this approach is to capture web application states. Elbaum presenting a approach in which he is integrating white box and user session based testing and showing results of that and proving the effectiveness of technique applied. There are some limitations in this testing technique. Web sites are incorporated with extreme dynamism these days so how to control unwanted source of variation is main issue. There should be any fault taxonomy to find faults in web application and to evaluate adequacy of web applications. Simulation of existing fault with current testing technique is required.

4 WEB APPLICATION TESTING TOOLS

This section presents an overview of the current web application testing. Several techniques and tools have been presented in the literature to support testing of web application but most of them focus on protocol conformance, load testing, broken link detection, HTML validation, and static analyses that do not address functional validation. These defects can easily detected, without manual effort [3,24]. Tools that do approach functional components automatically are active research area [23]. Structural testing techniques require the construction of a model [12], which is usually carried out manually. Web application testing tools are able to automate test case generation, test case execution, and evaluation of test case results.

In the past years, large variety of web testing tools has been developed. A list of more than 500 tools listed in 13 categories [17]. Existing web testing tools can be used to support non-functional testing, navigation of web site. Some tools can be used to test functionality of a web application.

Benedikt et al. present VeriWeb tool for automatically exposing paths of multipage web site through a crawler and detector for abnormalities such as navigation & page error [2]. VeriWeb crawling algorithm has some support for client side scripting execution. A tool like CATCUS is for Unit testing of web application. This requires the developer to manually create test cases and oracles of expected output [19]. These tools are not able to test asynchronous nature and extensive dynamic nature of current web applications. Tools such as WAVES and SecuBat are automatically assessing web application securities [18]. These tools are generic and modular web vulnerability scanner that, similar to a port scanner, automatically analyzes web sites with the aim of finding exploitable SQL injection and XSS vulnerabilities[5]. Apolb is a tool for finding faults in PHP web application that is based on combined, concrete and symbolic execution and ReWeb is used to model web application in UML. JsUnit is a tool to test java script on a functional level but this tool is not able to cope with heterogeneous nature of web applications i.e. will not test all technologies used to develop web applications. Selenium [6] is a well known efficient capture/ Replay tool allows DOM based testing by capturing events fired by user(tester) interaction. where interaction with the browser are recorded and then replayed during testing like in Selenium IDE[6]. WebKing tool can be used to perform tests on paths, recording paths through the browser. But functional tests that involve them, that part of the path will have to be created manually. Sahi is an open source testing tool with the facility to record and playback scripts. This tool test simple Java script events in the browser.

5 STATE OF ART OF WEB TESTING

To bridge the gap between existing web testing techniques and main new feature provided by web application. The server side can be tested using any conventional testing technique. Client side testing can be performed at various levels. The selenium tool is very popular capture-replay tool and allows DOM based testing by capturing user session i.e. events fired by user. Such tool can access the DOM and shows expected UI behaviour and replay the user session. So today's need is a testing tool which can test user session and generate test cases on the basis of expected UI behaviour as per event fired by user.

State Based Testing: Marchetto proposed a state based testing technique [20, 21]. Idea is that the states of client side components of an AJAX application need to be taken into account during testing phase [21]. State based testing technique for AJAX is based on the analysis of all the states that can be reached by the client-side pages of the application during its execution. Using AJAX, HTML elements—like TEXTAREA, FORM, INPUT, A, LI, SELECT, OL, UL, DIV, SPAN, etc.—can be changed at runtime according to the user interactions. In this testing the HTML elements of a client-side page characterize the state of an AJAX Web page, and their corresponding values are used for building its finite state model. State based technique results indicate that state based testing is powerful and can reveal faults otherwise unnoticed or very hard to detect using existing techniques.

Marchetto used traces of the application to construct a finite state machine [21]. This technique was based on the dynamic extraction of finite state machine for a given AJAX application. Whereas in Marchetto's work, dynamic analysis was partial, using manual validation or refinement steps for model extraction. He accepted in his work that FSM recovery needs an improvement and is an unexplored area. Dynamic extraction of states is quite tough to explore and needs constant attention in AJAX testing. In Marchetto's work, dynamic extraction of states was manual and needs a proper approach. There is a need for Automatic Dynamic analysis for model construction.

Traces	Event Sequence
1	add
2	rem
3	add, add, rem
4	add,add, rem,rem,rem
5	add, empty
6	add, empty, rem

FIGURE1. TRACES FOR CART EVENTS ONLY [FIGURE TAKEN FROM SOURCE [21]]

Execution traces can be traced using log files generated by real user interaction. Indeed, one approach proposed by Elbaum [6]. Marchetto explained state based testing using traces of cart as shown in Figure1. Traces contain information about Dom states and call back. Dom states are abstracted from concrete states. No of possible concrete states are huge and unbounded so can cause state explosion problem. For reducing no of states, using state abstraction function in this research work. Figure 2 shows FSM obtained from the traces by means of state abstraction function. Generating test cases using generated FSM.

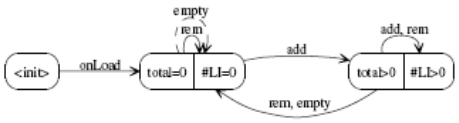

FIGURE2. FSM OF CART APPLICATION [FIGURE TAKEN FROM SOURCE [21]]

Later in his work Marchetto was mainly concerned to identify sets of "semantically interacting" events sequence, used to generate test suite of test cases [7]. His intuition was that longer interaction sequences have higher faults. The Conducted experiments showing that longer interaction sequence have higher fault exposing capability[7,8,20,21].This technique generates high number of test cases involving unrelated events, for minimizing test cases using notion of semantically interacting events. So here Marchetto's main contribution is for analysis of "semantically interacting" events sequence and result proves that more faults at the time of long interacting sequence analysis. Sequences of semantically interacting events in the model are used to generate test cases once the model is refined by the tester. In this work, he applied search-based algorithm, hill climbing and simulated annealing, to the problem of generating event sequence of various lengths. A FSM based testing technique generates high number of test cases. These high number of test cases reaches to a State Explosion Problem. In Marchetto's work for minimizing test cases used notion of semantically interacting Events and using abstraction function. However, in his work not explored state abstraction function completely. I worked mainly for semantically interacting events for minimizing test cases. It minimizes only few asynchronous communication test cases. It is not suitable to test case minimization for other AJAX features. He accepted in his work[7] that FSM recovery needs improvement, in order to automatically infer proper abstraction function.

Invariant Based Testing:

Static analysis techniques are not able to reveal faults due to dynamic behaviour of modern rich web application. Generating test cases for dynamic run time interaction of a web application is really a tedious task. Mesbah proposed an "Invariant based Automatic testing of AJAX user interface". In his work, first task was crawling of the AJAX application using CRAWLJAX1 tool, simulating real user events on the user interface and infer the abstract model from state flow graph [9,10].This CRAWLJAX tool design state flow graph of all(client side) user interface states. Mesbah identified AJAX specific faults that can occur in such states. In this work automatically generating test cases from the path discovered during crawling process. Mesbah testing AJAX states through invariants. Mesbah's suggested further AJAX research topic in his paper research issues in the automated testing of Ajax applications[26]. Out of all research issues one issue is to automatic invariant detection. His invariant based testing was dependent on CRWLAJAX and in current research issues described in his paper best path seeding practice in web application is capture and replay which was not used in his work. Mesbah proposed in his latest work that invariant based testing is a weak form of an oracle, which can be used to conduct basic sanity check on the DOM-tree or transition in the derived GUI state machine [8].For dynamic extraction of states best approach is using any capture and replay tool like Selenium otherwise AJAX is too dynamic that not able to test that correctly. State space reduction is also current issue related to state based web testing. State space reduction is an unexplored area. Indeed, path seeding capture replay technique will help in state space reduction.

6 FUTURE WORK AND CONCLUSION

As more and more web technologies have moved a long way to create web application. Web testing plays an important role. Here in this paper we discussed two well known testing techniques:-state based testing and invariant based testing. While these approaches are tested successful on various case studies, many problem remains, related to mainly scalability issue. How to capture user session data? How to avoid state explosion problem or how to reduce state spaces? How to improve FSM recovery steps, in order to automatically infer user session based test cases. In this research work, DOM manipulation of code into an FSM needs a proper technique. The expe-

riments conducted in this direction are able to generate test case for semantically interacting events and proofs are available that long sequences generates huge test cases and having higher fault exposing capability [8].Future work can be to reduce state space reduction by applying any path seeding algorithm for automatically generating FSM.

In this paper, we have covered resemblance and differences between web application testing and traditional software testing. We considered web testing with respect to various web testing techniques and Web testing tools. This research paper is providing help to get information about existing web testing technique, current scenario of web testing and proposing new research direction in web testing field. The main conclusion is that all testing are fully dependent on implementation technologies and future testing techniques have to adapt heterogeneous and dynamic nature of web application. This finding remarks that, there is a need to generate a test environment to test latest web technology designed web application and exercise each of them. New testing issues can arise for testing web services for improving effectiveness and efficiency of web .

REFERENCES

[1] Bellettini, C., Marchetto, A., Trentini, A.: Testuml: User-metric driver web applications testing. In: ACM Symposium on Applied Computing (SAC), New Mexico, USA (2005).

[2] Benedikt, M., Freire, J., Godefroid, P., VeriWeb: Automatically Testing Dynamic Web Sites, In proceedings of the 11th International Conference on the World Wide Web (Honolulu, Hawaii, May 2002).

[3] Elbaum, S. ,Karre,S., Rothermel,G., Improving web application testing with user session data. In International conference of software Engineering, pages 49-59, 2003.

[4] Fujiwara, S., Bochmann, G., Khendek, F., Amalou, M., Ghedasmi, A. , Test selection based on finite state models, IEEE Transactions on Software Engineering 17(6):591-603, June 1991

[5] Kals, S., Kirda, E. , Kruegel,C., Jovanovic, N., SecuBat: a web vulnerability scanner, Proceedings of the 15th international conference on World Wide Web, May 23-26, 2006, Edinburgh, Scotland. doi>10.1145/1135777.1135817

[6] Larson, J.,Testing ajax applications with selenium. InfoQ magazine, 2006.

[7] Marchetto, A., Tonella, P., Search-based testing of ajax web applications. In: Proc. of IEEE international symposium on search based software engineering (SSBSE). IEEE Computer Society, Windsor 2009 , pp 3-13.

[8] Marchetto, A., Tonella, P., Using search-based algorithms for Ajax event sequence generation during testing, 2010.

[9] Mesbah, A., Bozdag, E., and van Deursen, A. (2008). Crawling Ajax by inferring user interface state changes. In Proc. 8th Int. Conference on Web Engineering (ICWE'08), pages 122-134. IEEE Computer Society.

[10] Mesbah, A.,Van Deursen, A., Invariant-based automatic testing of Ajax user interfaces. In Proceedings of the 31st International Conference on Software Engineering (ICSE'09), IEEE Computer Society, 2009, pages 210-220.

[11] 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: 434-44. 10.1109/SITIS.2007.16

[12] Ricca, F. , Tonella,P. Analysis and testing of web applications, In ICSE '01 Proceedings of the 23rd International Conference on Software Engineering, pages 25–34, 2001.

[13] Andrews, A., Offutt, J., Alexander, R.: Testing web applications by modeling with FSMs. Softw. Syst. Model. 4(3) (2005)

[14] C. Liu, D.C. Kung, P. Hsia, C. Hsu, Object-based data flow testing of Web applications, in: Proceedinds of the First Asia-Pacific Conference on Quality Software, IEEE Computer Society Press, Los Alamitos (CA), 2000, pp. 7–16.

[15] Chow T (1978) Testing software designs modeled by finite state machines. IEEE Transactions on Software Engineering SE-4(3):178-187, May

[16] G. A. D. Lucca and A. R. Fasolino, "Testing Web-based Applications: The State of the Art and Future Trends", Information and Software Technology, vol. 48, 2006, pp. 1172-1186.

[17] Hower, R., Web site Test Tools and Site management Tools. Software QA and testing resource center.http://www.softwareqatest.com/qatweb1.html>2011 (accessed Aug 23, 2011)

[18] Huang,Y.W., Tsai, C.H., Lin,S.-K.Huang, D.T.lee and S.-Y. Kuo. A testing framework for web application Security assessment.Journal of Computer Networks,48(5):736-761, 2005.

[19] Jakarta cactus. http://jakarta.apache.org/cactus/, 2009.

[20] Marchetto, A., Ricca, F., and Tonella, P. (2008a). A case study-based comparison of web testing techniques applied to ajax web applications. Int. Journal on Software Tools for Technology Transfer, 10(6):477-492.

[21] Marchetto, A., Tonella, P., and Ricca, F. (2008b). State-based testing of Ajax web applications. In Proc. 1st IEEE Int. Conference on Sw. Testing Verification and Validation (ICST'08), pages 121-130. IEEE Computer Proceedings of the 23rd International Conference on Software Engineering, pages 25-34,Society.

[22] Park,S., Kwon,G.: Avoidance of State Explosion Using Dependency Analysis in Model Checking Control Flow Model. ICCSA (5) 2006: 905-911

[23] Ricca, F. and Tonella,P., Web Testing: a Roadmap for the Empirical Research in WSE'05 Proceedings of the Seventh IEEE International Symposium on Web Site Evolution , pages , 2005

[24] Sara E. Sprenkle. Strategies for automatically exposing faults in web application.PhD thesis 2007.

[25] Sprenkle,S., Pollock, L., Esquivel, H., Hazelwood,B.,Ecott,S., Automated oracle comparators for testing web applications. In Proc. 18th IEEE Int. Symp. on Sw. Reliability (ISSRE'07), pages 117-126. IEEE Computer Society, 2007.

[26] Van Deursen, A.,Mesbah , A., Research Issues in the Automated Testing of Ajax Applications. In Proceedings 36th International Conference on Current Trend in Theory and Practice of Computer Science (SOFSEM), pp. 16-28. Lecture Notes in Computer Science 5901, Springer-Verlag, 2010.