International Journal of Modern Trends in Engineering and Research
www.ijmter.com
e-ISSN: 2349-9745
p-ISSN: 2393-8161

Web Application Testing
(Major Challenges and Techniques)

Pooja Sharma[1]
[1]*Dept. of CSE,CBS Group of Institutions, Fatehpuri (Jhajjar), India, sharma.pooja218@gmail.com*

Abstract--Web-based systems represent a young, but rapidly growing technology. As the number of web applications continues to grow, these systems enter a critical role in a multitude of companies. The way web systems impact business aspects, combined with an ever-growing internet user mass, emphasize the importance of developing high-quality products. Thus, proper testing plays a distinctive part in ensuring reliable, robust and high performing operation of web applications. Issues such as the security of the web application, the basic functionality of the site, its accessibility to handicapped users and fully able users, as well as readiness for expected traffic and number of users and the ability to survive a massive spike in user traffic, both of which are related to load testing. 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. 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.
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. Functional requirement testing of web application will be considered in this paper. So after discussing challenges of web testing, focus of the remainder paper will be on compendious of various existing testing techniques for web application. Finally, there is a brief discussion about future trends of testing scopes in web applications and concluding remarks.

Keywords— Web application, Web Testing, testing challenges and techniques.

I. INTRODUCTION

The Web has had a significant impact on all aspects of our society, from business, education, government, entertainment sectors, industry, to our personal lives. The main advantages of adopting the Web for developing software products include (1) No installation costs, (2) Automatic upgrade with new features for all users, and (3) Universal access from any machine connected to the Internet. On the downside, the use of server and browser technologies make web applications particularly error-prone and challenging to test, causing serious dependability threats. In addition to financial costs, errors in web applications result in loss of revenue and credibility. The difficulty in testing web applications is many-fold. First, web applications are distributed through a client/server architecture, with (asynchronous) HTTP request/response calls to synchronize the application state. Second, they are heterogeneous, i.e., web applications are developed using different programming languages, for instance, HTML, CSS, JavaScript on the client-side and PHP, Ruby, Java on the server-side. And third, web applications have a dynamic nature; in many scenarios they also possess non-deterministic characteristics.

e-ISSN: 2349-9745
p-ISSN: 2393-8161

According to Petersen et al. [1], a systematic mapping (SM) is a method to review, classify, and structure papers related to a specific research field in software engineering. The goal is to obtain an overview of existing approaches, outlining the coverage of the research field in different facets of the classification scheme. Identified gaps in the field serve as a valuable basis for future research directions [2, 3]. Results of SM studies can also be valuable resources for new researchers (e.g., PhD students) by providing a detailed overview of a specific research area [4]. In recent years web applications have come to play an increasingly important part in software engineering in general, and also play a vital part in our everyday lives. As these systems offer more and more functionality to the user and enter new markets and new business segments, testing of such systems becomes all the more relevant, important and challenging. Being a relatively young technology there is a risk that existing tools and methods are either imperfect or simply have not been employed by certain software vendors. The lack of experience in testing web applications might be one reason why some companies find it tempting to use ad hoc testing techniques and even make testing the big loser when time is running out. Testing becomes an even more important factor when we consider the numerous properties of web systems that we do not necessarily find in traditional software applications. We will address this, and highlight the differences between traditional systems and web-based systems.

II. TECHNOLOGY

The first thing that probably strikes most of us when given the question of what distinguishes a web application from other software, is that these applications are accessed through a browser. And even though the use of a browser for any type of web system might suggest a standardized and less complex way of dealing with software, the myriad of browser versions among the different vendors complicates testing matters as well as the general quality assurance. Another distinct feature of web pages is the hyperlink, offering the possibility to bring the user to a different site or to a particular web page.
Furthermore web systems do not require any installation in the way we are used to think of that term. With the continuously updated prototype "shipped" to the production site and with all access or user interaction with the application taking place via web browsers, the installation process is practically non-existent.

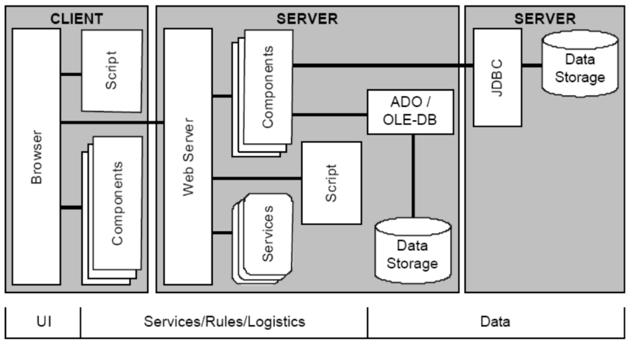

Figure 1. Web System Architecture

Not only do web applications involve several third party products, but they also have several physical points where potential slow-downs might occur. A typical three-tiered web system is shown in figure above. Hence the performance of most web-based systems nowadays relies heavily on web servers, application servers and database servers.

III. PROBLEM DEFFINATION

e-ISSN: 2349-9745
p-ISSN: 2393-8161

The problems and challenges that come in the wake of testing web systems are many and diverse. First of all there seems to be inadequate technical expertise in the field and also a lack of mature testing tools for these applications. The main cause for this is most likely web systems, and the testing of these, being a relatively "juvenile" and rapidly changing technology. The temptation for software vendors to ad hoc software engineering processes might also lead to a somewhat informal version management, thus complicating testing matters by not having a properly working version to revert to. This makes analyzing defects even harder since one has to deal with an environment that is becoming increasingly complex. There are also testing concerns from a more technical point of view. Web systems are built on partly separate development technologies related to programming language, user interface and user interaction. They depend on COTS products – Commercial Off-The-Shelf – that cannot be fully controlled, such as Flash, Java Applet, Acrobat Reader and ActiveX to name a few. Multiple users sharing the same resources, like a web server, increases the possibility of the user experiencing some kind of trouble, especially when the number of users wanting to use a particular service exceeds the threshold for what the system can handle. Another complicating factor is the pressure of making time-to-market as short as possible. This has led the web industry into shorter iterations, implying that test passes ought to or must be performed in step with the development cycles. This in turn means that testing activities will take place more frequently than what is the case for traditional software engineering. The tests to be performed will have to focus on detecting errors that may cause unacceptable levels of robustness. The latter rises the challenges of finding methods tailored to testing of web systems and assess their suitability for the robustness of such applications. All of the issues mentioned above, combined with the multitude of different browser versions among different vendors, make testing and quality assurance in the context of web-based systems a challenging prospect on more than one level.

IV. WEB APPLICATION TESTING CHALLENGES

Testing of web application employing new technologies (like AJAX, Flash, Active X plug-in component, Ruby on rails, etc.) is an area that has not been investigated so far. In this paper 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 state full client, asynchronous communication, delta updates, unwritten JavaScript, client side DOM manipulation, event handling, timing, back / forward button and browser dependence.
A web application faces various challenges during testing and should be able to conduct tests for:

- Browser compatibility
- Operating System compatibility
- Windows application compatibility where required (especially for backend testing)

Web application testing allows a user to specify how virtual users are involved in the testing environment i.e. either increasing users or constant users or periodic users load.

- Increasing user load, step by step is called Ramp [5] where virtual users are increased from 0 to hundreds, where, Ramp test is the test which uses escalating numbers of users over a given time frame to determine the maximum number of users the web server can accommodate before producing error messages.
- Constant user load maintains specified user load at all time.
- Periodic user load tends to increase and decrease the user load from time to time.

Websites offer new challenges to developers, as well as testers. Scalability and performance are two areas that are significant in the e-commerce and web applications space. However, when new

e-ISSN: 2349-9745
p-ISSN: 2393-8161

technology is used to make web applications perform well and scalable, new testing methodologies have to be created along with those technologies. Performance is critical, and based on a study from the Newport Group, more than half the recently deployed transaction-based web applications did not meet expectations for how many simultaneous users their applications could handle[6].

Challenges of web testing because of embedded features of current web technologies are as follows:

A. Asynchronous behavior

Web application nondeterministic behavior because of asynchronous behavior is also a great testing challenge. This non-deterministic behavior 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 behavior 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 application testing also. This asynchronous challenge may reach to an incorrect final state or some output values may be different from the expected ones [7].

B. 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 [7] 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 [8, 7]. 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 behavior 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 behavior makes testing problematic.

C. 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 behavior for testing. Problems may arise in detecting all dynamically generated and updated states.

D. State full behavior

e-ISSN: 2349-9745
p-ISSN: 2393-8161

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 state full, 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 [9]. State full behavior is a challenging task because states itself contains many information and state behavior 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.

E. Delta server message

Delta-Server messages [9] 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 [10].

V. WEB APPLICATION TESTING TECHNIQUE

Marchetto [11] 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.

A. White Box Testing

White box testing designs 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 [12, 13, 14, 15]. 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 behavior 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 [8] follows primarily two testing methods. Object based data flow testing and Control flow model based testing. Object based data flow testing [16] client tier interaction behavior not server tier interaction behavior 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 modeled 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, inter procedural 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

e-ISSN: 2349-9745
p-ISSN: 2393-8161

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 behavior of web application nowadays. These days' web applications are designed using DOM element during execution and add 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 [17, 18].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 re-quests to the server and lack of complete state information required in AJAX testing.

B. 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 behavior 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 behavior of web application in consideration and derive test case from them [14]. Andrew proposed a system-level testing technique that combines Test Generation based on Finite State Machines with constraints [14]. 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 [19, 14, 20, 21]. 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 behavior 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 dependency analysis in model checking control flow graph [22]. The FSM is a model for describing the control flow aspects. FSM, like a State charts, 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[23] that exploits decision table as a combinatorial model for representing the behavior of web application and generating test cases. Second solution proposed by Andrew that model state ma-chine using state dependent behavior of web application and generates test cases.
Second approach is user session based testing approach suggested by Elbaum [17]. 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 an approach in which he is integrating white box and user

e-ISSN: 2349-9745
p-ISSN: 2393-8161

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.

VI. 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 behavior 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 behavior as per event fired by user.

A. State Based Testing

Marchetto proposed a state based testing technique [7, 8]. Idea is that the states of client side components of an AJAX application need to be taken into account during testing phase [7]. 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 [7]. 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. 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[7,8,24,25].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

e-ISSN: 2349-9745
p-ISSN: 2393-8161

AJAX features. He accepted in his work [24] that FSM recovery needs improvement, in order to automatically infer proper abstraction function.

B. Invariant Based Testing

Static analysis techniques are not able to reveal faults due to dynamic behavior 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[9,27].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 [10]. Out of all research issues one issue is to automatic invariant detection. His invariant based testing was de-pendent 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 de-rived GUI state machine [25]. 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.

VII. FUTURE WORK AND CONCLUSION

In this research paper, the importance of website testing have been highlighted which is one of the new breed of testing for the past few years. A survey on web testing methods and challenges described some issues and challenges and ways to avoid same issues. 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 experiments 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 [25]. 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.

e-ISSN: 2349-9745
p-ISSN: 2393-8161

REFERENCES

[1] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson. Systematic mapping studies in software engineering. In 12th International Conference on Evaluation and Assessment in Software Engineering (EASE), pages 71– 80, 2008.

[2] B. Kitchenham, D. Budgen, and P. Brereton. The value of mapping studies - a participant-observer case study. In PProceedings of Evaluation and Assessment in Software Engineering, 2010.

[3] B. A. Kitchenham, D. Budgen, and O. P. Brereton. Using mapping studies as the basis for further research: A participant-observer case study. Journal of Information and Software Technology, 53:638–651, 2011.

[4] D. Budgen, M. Turner, P. Brereton, and B. Kitchenham. Using Mapping Studies in Software Engineering. In Proceedings of PPIG 2008, pages 195–204. Lancaster University, 2008.

[5] Retrieved from the World Wide Web: http://www.paessler.com/webstress/features

[6] Shea, B. (2000, May/June). Avoiding Scalability Shock. Software Testing & Quality Engineering Magazine, 42-46.

[7] 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.

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

[9] 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.

[10] 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. Lec-ture Notes in Computer Science 5901, Springer-Verlag, 2010.

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

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

[13] 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.

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

[15] 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

[16] 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.

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

e-ISSN: 2349-9745
p-ISSN: 2393-8161

[18] 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.

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

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

[21] 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)

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

[23] 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.

[24] 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.

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

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