A Survey of Testing Techniques for Testing Web based Applications

Sangeeta Sabharwal[1], Ritu Sibal[2], Chayanika Sharma[3]
NSIT, Delhi University
Sector 3, Dwarka
Delhi, India
ssab63@gmail.com, ritusib@hotmail.com, chayanika_29a@yahoo.com

ABSTRACT: *Nowadays, the internet has become the primary source of information worldwide. The overwhelming increase in the number of users demands the development of web systems which are compatible with changing user requirements. To ensure high quality web based systems, exhaustive testing of web application is therefore mandatory before they are made live. This paper presents a literature survey of existing techniques used for testing web applications. The main focus of this research paper is on the testing strategies used so far in the area of web application testing. To this end, we performed a search on publications in the selected electronic databases published from 2000 to 2014. Through our careful review, a total of thirteen papers have been selected as primary studies. A comparative analysis of the various techniques, used for testing web based applications is also discussed in the paper.*

Keywords: Survey, Web Application, Testing

Received: 21 February 2015, Revised 19 March 2015, Accepted 25 March 2015

1. Introduction

In today's world, internet is the most powerful tool for providing services and information across the global networks. To address the issues like maintainability, testability, security, performance, correctness and reliability of web applications, testing of web applications is done [2]. Due to the very large number of user interactions, existing techniques for testing traditional software are not sufficient for testing web applications. With the enhancement of the internet, web server technologies, multiprocessors and constantly evolving web architectures, traditional means of testing are not enough to exhaustively test today's complex web applications with multi-tier systems and multiple integration points. Therefore, testing of web applications has become a challenge and the focus of research due to increase in the number of web users.

Web applications are composed of web pages and components [3]. A web page is information viewed on the client side in a single browser window [3]. The interaction between web pages and web components executes web servers, Hyper Text Transfer Protocol (HTTP), browser and networks [3]. A website contains text, images, links and web pages. The web application structure is based on the client-server architecture. There are at least two machines required, i.e. client computer and the server computer. The client sends the request for service to the server computer and the server serves that requested server. The client can be in the form of application, service or software module. The server is a service or a network addressable entity that accepts and executes the requested service. The server can be mainframe system, a component or

some software system [13]. Two tier architecture is used for only two separate computers. Two–tier architecture is not suitable for large website where the security demand is very high. To overcome the limitations of Two- tier architecture, the architecture was expanded to three-tier architecture and then to n-tier architecture. Due to this expansion, web applications are now more scalable and maintainable [3]. The effective testing of a web application depends on many factors. These factors can be testing strategies, test models, algorithms to create test cases from test models and representation of web application components to be tested [5].

The organization of the paper is as follows. In section 2, we discuss various challenges encountered during testing web based applications. Section 3, presents different White Box, Black Box and Gray Box Testing techniques used by the researchers for testing web applications. Finally, section 4 concludes our work by doing a comparative analysis of the various techniques used for testing web based applications as discussed in section 3.

2. Challenges in Testing Web based Applications

Testing of a web application is divided into two categories: - Functional Testing and Non – Functional Testing. Functional Testing is used to test the functional requirements of the web application, whereas Non- Functional Testing is used to test the non functional requirements of the web application.

Functional Testing is used to verify that a web application conforms to the stated requirements. Non- Functional Testing is done to test the conformance to requirements like reliability, maintainability, performance, scalability, usability and security. Testing a web application is one of the challenging tasks in software testing for following reasons:-

♠ Large and diverse users of a website having multiple submit and send requests from the client and server model.

♠ Exposure to security threats like cross–site scripting (XSS), broken authentication, improper error handling [14].

♠ Illegal point of entry into databases and systems containing confidential information may open in web applications.

♠ Testing the performance of a website in terms of server response to the number of client requests.

♠ Reusing the test cases and the reusable components of a web application to develop a website that meet the new requirements of a client in the dynamic environment.

Therefore, to ensure that a web application is working correctly, the factors mentioned above need to be accounted. Test cases covering the functional and non – functional requirements of a web application should be developed covering different aspects of a web application. As the technologies and methodologies to build the web applications and the load on the server changes continuously, the effort should be made to cover these factors so that the web application functions without any failure.

3. Testing Strategies for Web based Applications

The testing of web applications can be done using traditional testing techniques. These techniques are: - White Box Testing, Black Box Testing and Gray Box Testing. In White Box Testing, the internal details of the software are analyzed. The Data Flow Testing and Path Coverage Testing are some of the techniques in White Box Testing. In White Box Testing, the internal architecture of the server and client is analyzed by understanding the technology (AJAX, HTML, etc.) used to implement the models of a web application. In Black Box Testing, the testing is concerned with only input values and the corresponding output values of the software system. Regression Testing is one of the techniques for building the test cases covering the modified component of a software system. Gray Box Testing combines the approach of Black Box Testing and White Box Testing. User Based Session Testing is One of the Gray Box Testing technique to build test cases on the basis of data captured from the user session.

3.1 White Box Testing

In this section, we will discuss some modeling languages and testing models used so far by the researchers during testing of web–based applications using White Box Testing techniques.

Ceri,S. et al. [8] have proposed a modeling language known as WebML to develop web applications. WebML is a modeling

language that provides modeling abstractions for a website which later can be translated by CASE into a concrete page. The web application is developed by following some patterns and rules. They used WebML to describe these patterns and rules. The website is modeled as pairs of data and hypertext diagrams. These pairs are called as "Skeletons". Using WebML, the pages of the website are expressed by creating Data Model and Hypertext Model. The data model is created using relevant entities and depicting the relationships between them.

The hypertext design of a website shows the navigation and flow of information from one component or unit to another component. The contents used in hypertext diagram are shown in Table 1.

The skeleton helps in creating and understanding the web application using data and hypertext model which is very helpful in:-

♠ **Re-engineering Web Application:** - The web application can be created by using the existing web application.

♠ **Classification:** - Using skeletons, website are classified into five kinds of websites: -

1. Commerce Sites e.g. e-shops, virtual market places.
2. Content sites e.g. Digital Libraries.
3. Service Sites e.g. Order –Tracking sites.
4. Community sites e.g. Chat Rooms, Online admission counseling sites.
5. Context Sites e.g. online directories.

Overall, the approach proposed by Ceri,S. et al. [8] is suitable for creating data intensive websites.

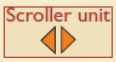
Unit	Visual notation	Description
Data		Shows data about a single entity instance.
Multidata		Shows data about several entity instances.
Index		Shows a list of properties (also called descriptive keys) of a given set of entity instances. A user clicks on an index entry to select and display one instance.
Scroller		Provides commands for scrolling through objects in a list — for example, the sequence of all the instances of an entity. Scrolling commands let the user move to a set's first, last, previous, and next elements and select and display one instance of the set.
Data entry		Shows a form with several fields for collecting user input. Input might be conditions used for searches over entity instances or parameters for operations such as content updates, logins, and generic external operations.

Table 1. Contents Unit of the WebML Composition Model [8]

In the area of Data Flow Testing, Liu,C. et al. [7] analyzed the data flow of HTML using a Web Application Test Model (WATM). The WATM captures the data flow information of web application using two models described below: -

♠ Object Model

♠ Structure Model

Liu,C. et al. defines three types of objects in the Object Model. These objects are Client Pages, Server Pages and Components [7]. In Data Flow Testing, variables in a program are divided into 'c-uses' and 'p-uses' variables [20]. The c-uses variables are those variables which are used in computations in a program, whereas p- uses variables are associated with the edges of the program based flow graph [20]. In order to fulfill the all-uses criteria, the def-clear path from each definition of a variable to each use of that variable need to be determined [20]. A def-clear path is a path containing no new definition of a current variable [20].

In object model, each component in WATM is an object containing attributes and operations. The relationships between them are depicted using association, composition, aggregation and inheritance relationship as used in object-oriented modeling. The data flow information of a web application is analyzed using Control Flow Graph (CFG). CFG is a directed graph used to show the data flow between the objects [17]. The nodes of a CFG represent statements or a group of statements called blocks and edges represents the control flow between the statement blocks. Liu,C. et al. divided CFG into two types of graph: - Inter Procedural Control Flow Graph (ICFG) and Composite Control Flow Graph (CCFG). ICFG is defined as a graph used to represent the data flow between the functions and def/use variable in a web page. CCFG is defined as a graph used to represent the data flow between interacting web pages. CCFG is constructed by connecting the CFG or ICFG of the web pages interaction. CCFG shows the HTTP request between client page and server page.Liu,C. et al. have derived test cases from three different viewpoints: -

♠ Intra–Object Perspective: - Def- use is computed at function, function cluster and object levels.

♠ Inter–Object Perspective: - Def- use is computed among set of objects that passes messages to each other.

♠ Inter–Client Perspective: - Def- use chain is constructed at application level.

The proposed work by Liu,C. et al. [7] shows the object based data flow between the client and the server of the web page. In web testing model, test paths are generated using XML. Few limitations have been encountered in their work. These are:

♠ The requests are sent between client and server in the form of HTML which is very outdated language. This requires WATM to test the current technologies in the web applications like XML and AJAX.

♠ It is a complex task to test all the feasible paths in data – intensive web application using WATM. Therefore, It is difficult to predict the quality of a web application accurately.

Qian,Z. et al. [4] constructed a tree instead of a graph to test pages of a web application. The Page Flow Diagram (PFD) is constructed to determine the dangling hyperlink, broken link or an unreachable link in a website.

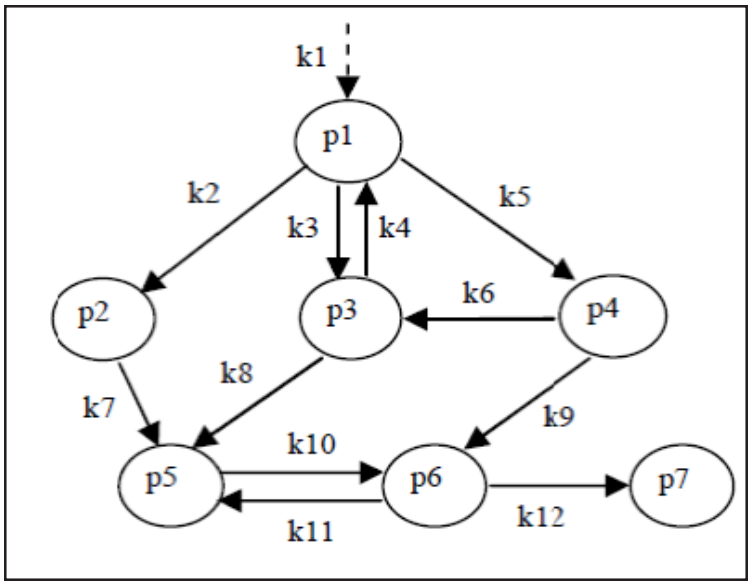


Figure 1. PFD of an Example Web Application [4]

In Figure 1, nodes represent sets of pages p directly linked with each other. The edges represent sets of links *k*1, *k*2, 3,....., kn. Where *i* = 1, 2, 3...., n. The dashed arrow is a first link in the PFD entering default page. To avoid cycles in PFD, a spanning tree called Page Test Tree (PTT) is derived from PFD using the PFD2PTT algorithm as shown in Figure 2. Using this algorithm all possible shorter paths in a web application are generated without any loss of page or link coverage. In this algorithm, two tables FIRST and SECOND are maintained. Initially, the page identifiers are unmarked and are maintained in table FIRST. SECOND is a table containing page identifies marked already. The link and page coverage criteria are used to generate test cases.

Input: A *PFD*

Output: a *PTT* derived from the *PFD*

begin

(1) Add the initial page identifier of the *PFD* into FIRST;

(2) if FIRST is empty, then go to (6);

(3) Select the first page identifier denoted by *pid* from FIRST. If *pid* is within SECOND, then go to (5).

Otherwise, add it into the end of SECOND;

(4) if *pid* is linking to other pages, then

• If some of the other page identifiers are within FIRST or SECOND, then generate their copies;

• Retain the links between *pid* and the other pages (or their copies) of the *PFD*, and

• Add the other page identifiers (or their copies) into the end of FIRST;

(5) delete *pid* from FIRST and then go to (2);

(6) output the derived PTT, which is the *PFD* with only the retained links.

end.

Figure 2. An Algorithm Deriving a PTT from a PFD [4]

PTT shows all the successive relationship between web pages. To derive test cases, using PTT a path expression (k1->k2->k3->.....->knee) is computed by applying Node Reduction Algorithm designed by Beizer, B. [15].

In Node Reduction Algorithm, all serial links are combined by multiplying their path expression, whereas parallel links are combined by adding their path expression. Self loops are replaced by link of the form *X** where *X* is the path expression of the link in that loop. Path expression is converted into test specifications. For validation of test cases, test engine is used which accepts the test specification as input in the form of XML syntax. The test cases are then executed by the test engine and the generated results are matched against the test oracle. The report of test oracle is later produced as the test report. In their work [4], we have found following limitations: -

- PFD is constructed manually, which is quite a complex task and costly.
- Nowadays, web applications are developed using AJAX or JavaScript which is not addressed in the work.

Di Lucca , G.A. et al. [5] used the object–oriented model of web application as a test model. In testing model, the classes or components represent client pages and server pages. The association is used to represent the relationship among these components. Client pages are static pages if their contents are fixed. Client pages are built pages if their content varies over time. The relationships in an object-oriented model of web application are shown as links connecting the page components. To perform functional testing of web application, use case diagram is used to determine the functional requirements of web application. The test cases of client pages are built using decision tables. The decision tables of client and server pages are shown in Table 2 and Table 3 respectively. Using decision table, driver and stub modules are developed. Di Lucca, G.A. et al.

considered driver module as a web page that communicates with the client by sending the input forms and generating events for the test cases. The client pages, server pages or web objects are used to develop a stub module. According to Di Lucca, G.A. et al., the complexity of server page is always higher than client page because of the language used for implementation and technologies used for developing the server pages.

Variant	Input Section			Output Section		
	Input Variables	Input actions	State before test	Expected Results	Expected output actions	Expected State after test
...	...			...		

Table 2. Decision Table Template for Client Page Testing [5]

A server generates the output in the form of client page. Different input on server generates client pages. So, both the server page and built page of client is unit tested.

Variant	Input Section		Output Section		
	Input Variables	State before test	Expected Results	Expected output actions	Expected State after test
...	...		...		

Table 3. A Decision Table Template for Server Page

In integration testing, the pages linked by hyper textual links, redirect links or submit ones are identified to avoid cycles. A link having smaller number of parameters is deleted.

Zou,Y. et al. [18] has introduced a hybrid coverage criterion for capturing the complex interactions between client side and server side in dynamic web applications. The hybrid coverage criterion counts the code executed and HTML elements exercised during server side and client side interactions. Zou,Y. et. al. defined HTML element as an individual component of HTML document which is the basic block for the front end content of web application. To represent the front end structure of AJAX based web application, a user interface model has been created by the web crawler. Zou,Y. et al. used Crawljax tool as the model generation tool for AJAX based web application. To analyze the effectiveness of proposed hybrid criterion, an experiment has been conducted. Initially, a test pool containing all test cases has been initialized. At every round of procedure zero or one test case has been selected. A test case with high statement coverage and low execution cost is selected in each round. The executing cost of the test case is measured by the cost of initializing the test case and the cost of executing the test commands in the test case. Results show that the hybrid-coverage-based strategy can detect more bugs than the statement-coverage-based and element – coverage - based strategies.

3.2 Black Box Testing

Song,B. et al. [1] used Finite State Machines (FSM) to model the behavior of server side and client side. The client and server message request and message sending behavior is taken into consideration for modeling their behaviour. Message request and responses are used to compose two synchronous FSM's. The test cases are generated by the composition of FSM of both client and server side. Bo Song et al. considered successor state of the current state to match the model under test. The process of finding successor state of the current state is repeated until there is no new state.

Riebisch,M. et al. [9] generated test cases using Unified Modeling Language (UML) use case model. In Figure 3, various modeling domains are shown in a software design. The use case model is transformed into usage model. The usage model is usage oriented and specification based, thus performing Black – Box Testing. Riebisch, M. et al. used usage models to describe how a software system can be used by the users in different ways. Usage model is an input for automated statistical usage testing measuring reliability. Statistical usage testing is a system level testing, which is used to find the code having

a higher frequency of execution than another portion of code based on which test suites are build. Usage models are transformed into markov chain describing the state of usage. In markov chain, next state in sequence depends on the present state. The state of usage is represented by state diagrams.

A state diagram is drawn for each use case and a usage graph is drawn for each state diagram. A usage graph has initial and final states. In usage graph, user action causes transition and generates new state.

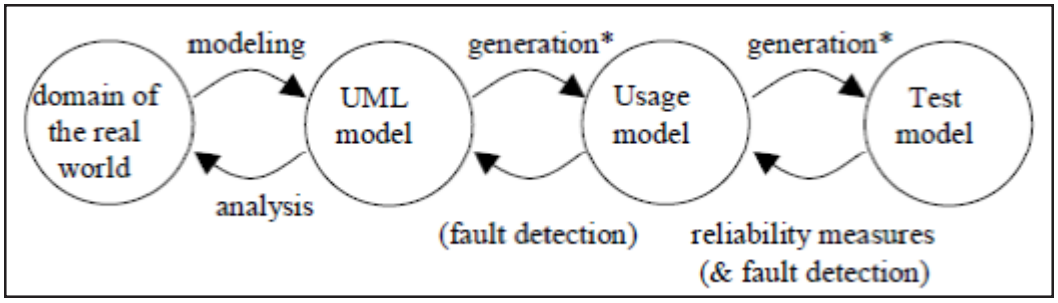


Figure 3. Systematization and Automation of Activities in various Modeling Domains [9]

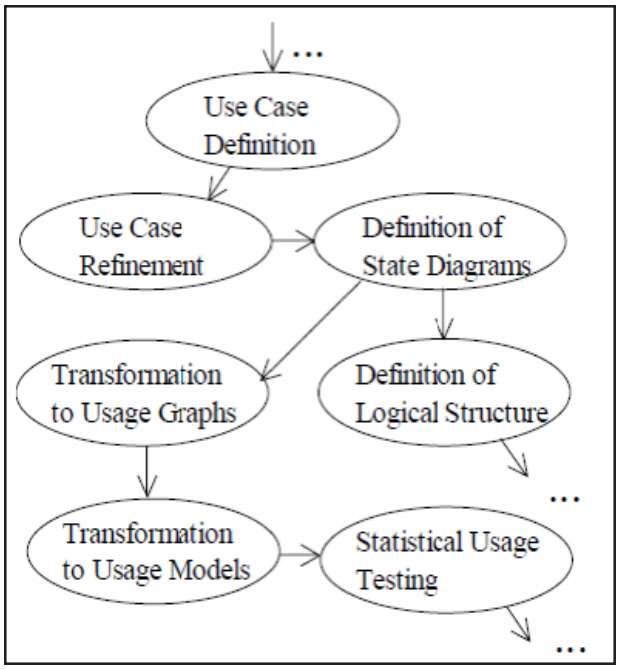


Figure 4. Various Activities in Riebisch's Proposed Approach [9]

In Figure 4, different activities of the Reibisch's proposed approach are shown. The usage graphs are converted into usage models by assigning transitions probabilities. For test case generation, the random testing is done on the usage model. The minimal arc coverage criterion has been used to develop the test suite. The sequence of transitions traversing the usage model is less in the minimal arc coverage criterion.

To Conclude: -

♦ The proposed approach by Reibisch et al. supports iterative development process, thereby reducing the test efforts.

♦ Flattening and removing hierarchical structures in a state diagram is a complex process and need to be automated.

A UML based approach has been proposed by Li, L. et al. [10] to test web applications using the use case diagram and sequence diagram. The web application is modeled as Use Case Transition Model (UCTM). The use cases are considered as components in the test model. The structural relationships are shown between use cases and stereotype <<communicate>>

is used to show the transition between the components (use cases). Sequence diagrams are used to show the dynamic behavior of a use case. The alternative or exceptional scenarios in use case diagram are depicted via branches and loops in sequence diagram. For loop testing, a predicate in the loop is tested for two alternative paths. The test scenarios are derived by transforming sequence diagram into Restricted Message On Vertex Graph (RMOVG). The vertices in RMOVG represent messages in sequence diagram and the edges represent the transitions between the messages. The bottom up approach is used to build RMOVG from use cases. Li,L. et al. proposed the Constraint Message Coverage criterion while constructing RMOVG for UCTM. According to this criterion, each message must visit the sequence diagram at least once.

Tarhini,A. et al. [6] have applied Regression Testing technique on the web applications. The web application is viewed as an event driven environment. Regression Testing is done to ensure that the changes made in the software system do not affect the functionality or behavior of the software system. The test suite is selected to cover the changes in the system. The web application is transformed into Event Dependency Graph (EDG) which is used to model three types of dependencies: -

· **Link Dependency: -** Two pages are link dependent if first page requests the second page via an event. For example, text, image or hyperlink.

· **Visible Effect Dependency: -** If the requesting page modifies the other page via an event and the page opens with modified content, then these two pages have Visible Effect Dependency.

· **Invisible Effect Dependency:** If the requesting page modifies another page via an event and the page open without any modified content, then these two pages have Invisible Effect Dependency.

The following steps have been followed by Tarhini,A. et al. [6] for Regression Testing technique [6]: -

1. The EDG of web application and modified web application is constructed.

2. The nodes from both the graphs are compared and changed nodes are identified.

3. The potential nodes affected by the changes are identified.

4. The test cases are selected that covers the changed nodes and the nodes affected by the changes.

In step 1, each node in the original EDG G and modified EDG G'are checked. If the node is present in G' then the neighbor nodes in G are checked for any changes made in G'. If the corresponding node is not found, it means that changes have been made in the modified EDG G'. In step 2, the potential affected nodes directly or indirectly dependent on the modified nodes are identified. Each dependency in the EDG is checked to identify the affected nodes. The neighbors of affected nodes are taken into consideration and are added to the affected node list. In step 3, the test cases are selected that covers the node in the affected node list.

Lebeau,F. et al. [19] presented an approach to overcome Multi-Step XSS vulnerability having multiple variants and automate Model Based Vulnerability Testing (MBVT) or web applications. To overcome XSS vulnerability, Lebeau,F. et al. applied Model Based Testing approach to generate test cases for web application. In their proposed MBVT, various activities like test purpose definition, model design, test generation, test concretization, test execution and verdict assignment have been included.

The Certify It tool provided by Smar testing company has been used for test generation supporting all the activities of MBVT. The UML4MBT has been used for capturing the behavior of Software Under Test (SUT) which is given as an input to CertifyIt tool. UML4MBT have three types of diagrams namely UML class diagram capturing the static view of SUT, UML object diagram capturing the relationship between different objects of SUT and state diagram representing a dynamic view of the SUT. To analyze the effectiveness of their proposed approach, the experiment is conducted on PHP/ MySQL based web application. The test cases are computed and executed to expose Reflected Cross Site Scripting (RXSS) vulnerabilities. Lebeau et al. defined RXSS attack as an attack containing the malicious data which is immediately sent back to the user [19]. Vulnerability Test Patterns (vTP) are formally constructed for expressing testing needs and to highlight the breach in the web application. The characteristics of vTP includes name, its description, its testing objective, its prerequisites, its procedure, its observation and its oracle, its variants, its known issues, its affiliated vTPs and its references. The formally defined vTP which are entry points of MBVT are then converted into machine readable language. The results of experiments show proposed MBVT as an accurate and precise technique.

Youmi, M.E. [21] proposed Fuzzy All Pairs (FAP) Testing approach that uses fuzzy testing and all-pairs testing for increasing the effectiveness of web application testing. The proposed approach includes four steps outlined below:-

1. **Partitions Identification :**- All possible combinations of valid and invalid classes or partitions are identified.

2. **Pairs Identification :** All combinations of valid classes identified in step 1 are identified. Invalid classes are tested independently.

3. **Test Cases Definition :** By using randomness of fuzz testing, a random value is selected. A random test case is executed inside the boundaries of the partitions.

4. **Random Values Selection :**Additional random test cases are executed to ensure that no errors exist.

The empirical results of Youmi, M. E. et al. proposed approach shows better results than Equivalence Partioning technique and Boundary Value Analysis technique.

To Conclude :-
Since their approach [21] is based on the fuzzy testing, a random value chosen for selecting test cases to be executed does not ensure that all test cases are executed. The selection criteria does not ensure that all bugs are discovered in a software.
.

3.3 Gray Box Testing
Elbaum, S. et al. [12] tested web application using the user session data to produce the test suite effectively at minimum expense. The technique used in their work uses the hybrid approach combining the Ricca and Tonella's [16] White Box Testing and User Session based Testing. The few problems addressed by them in the web
applications are: -

• **Server Load:** - Suppose a website is built with some users in mind, but the load on the server exceeds suddenly.

• **Maintenance:** - The maintenance of web application is done at a faster rate than other software systems. Therefore, test suites that cover the changes made in the web application must be developed automatically.

• **Web Architecture:** - The test approaches must be able to handle the complex, multi tiered and heterogeneous architecture of web applications.

The following assumptions have been made in the Elbaum, S. et al. [12] procedure: -

• Only independent paths are tested.

• As many inputs generate many pages, so for testing search fields only one input value is taken into consideration.

• Circular links and textual differences in dynamic pages are not considered.

The nodes in the model represent web pages, frames or web objects and edges represent relationships between them. In their work, Ricca and Tonella [16] approach has been followed and quasi test cases are generated. The quasi test cases are in the form of e1e4, e2e6 .....e5e9 where e1, e2,....,en are the edges connecting the web objects of the web application. The quasi test cases provide the target URL paths. The test cases are generated semi-automatically from the static structure of the web application by using White Box Testing criteria. The user boundary values are used as input values for White Box Testing. For test case generation, the client requests in the form of URL's and name-value pairs are stored. The two user based session testing techniques applied by Elbaum,S. et al. are: -

1. US –1

a) Each user session is transformed into test cases.

b) The test case corresponding to each user session containing request r_1, r_2,.....,r_n is formatted into HTTP requests that can be sent to the web server.

c) The resulting test suite has m test cases, one for each user session.

2. US –2
a) Based on the collected data, user sessions are generated.

Their approach captures the user events on the server site. To conclude,

- The effort in capturing the url and name-value pairs is less as these are processed by the web applications.
- The user intervention for capturing the user behaviour is less.

Peng,X. et al. [11] also used the user session data in their Request Dependence Graph (RDG) to generate test cases by applying Genetic Algorithm (GA). GA is a search algorithm that is inspired by the way nature evolves species using a natural selection of the fittest individuals [20]. The possible solutions to the problem being solved are represented by a population of chromosomes [20]. A chromosome is a string of binary digits and each digit that makes up a chromosome is called a gene.

The structural analysis of web application is done using RDG construction. The request dependence is shown as the relationship between the components or nodes of the web application. The edge represents the request dependence between two pages. The request labeled on the graph is formatted as: - "GET/POST PAGE < P1, P2,.....,Pn > where P1, P2,....., Pn are the possible parameters in the request. All the parameter- value pair of requests is enumerated without values. The RDG and their corresponding labeling are shown in Figure 5.

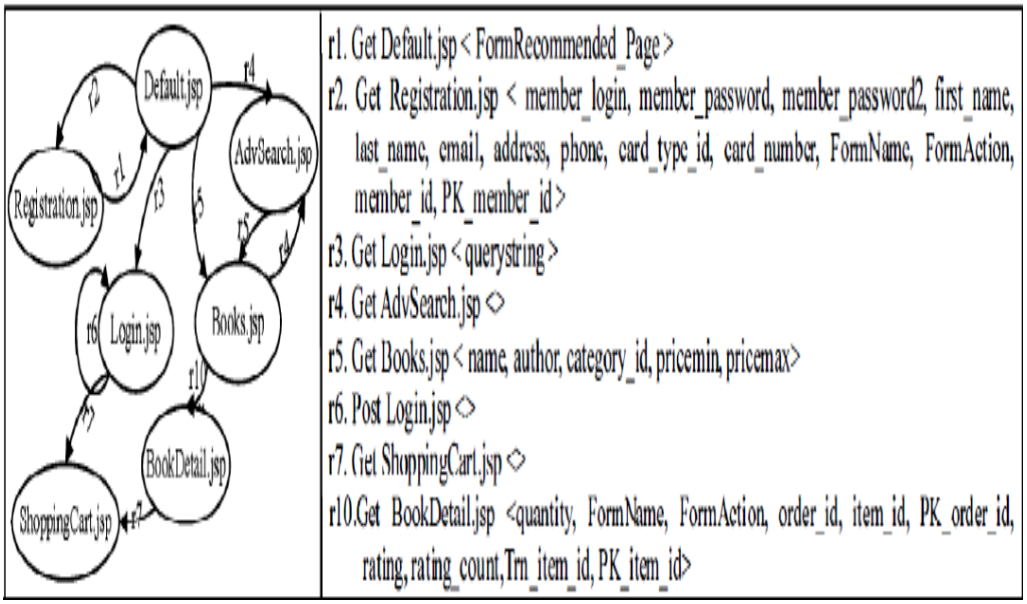


Figure 5. Partial Request Dependence Graph and the Labelled Requests [11]

The test cases should cover as many relationships as possible in RDG. The user session data is taken as the initial population in GA. A gene is encoded as combination of requests and pages. A user session is identified as a sequence of requests and parameter values to describe the user's requests for web services. Each session is represented as transition relation. e.g. "Request -> Page -> Request -> .....->Page -> Request". (A chromosome is encoded as a transition relationship between page and request). The fitness value is computed as: -

$$\text{fitness} = (\propto * |\text{CLTR}| + |\text{CLTR}|) / (\propto * |\text{DTR}| + |\text{LTR}|)$$

Where CDTR is the number of data dependence transitions covered in the chromosome, CLTR is the number of link dependence transitions covered in the chromosome. DTR is the number of data dependence and LTR is the number of link dependence transition relations in the web application.

4. Conclusion

In this paper, we have discussed various testing strategies used for testing web applications. The existing technologies and strategies encountered during testing of web based application are highlighted in our work. Table 4 gives a comparative analysis of the various techniques that are prevalent for testing web based applications. The contribution of each strategy has been highlighted in this table along with the approach that has been adopted for testing web applications.

S. No.	Testing Technique	Author	Approach/Technique	Contribution
1.	White Box Testing	Ceri, S. et al., 2002	Web Modeling Language	Suitable for data intensive websites.
		Liu, C. et al., 2000	Captures data flow information of the web page. Test case derived from intra- object perspective, inter-object perspective, and inter-client perspective of the web page.	The Graph is constructed which shows the object based data flow between the client and server of the web page.
		Qian, Z. et al., 2007	PFD is constructed that show relationship between pages of a web application.	Optimization of the number of test cases using PFD2PTT algorithm is proposed; this algorithm shows all possible shorter paths in a web application.
		Di Lucca, G. et al., 2002	The client and Server pages are unit tested.	Model of web application is constructed using object oriented approach. The Class diagram is used to model client pages and server pages. Use case diagram is used to determine functional requirements of web application.
2.	Black Box Testing	Zou,Y. et al., 2013	Hybrid Coverage Criterion	Suitable for Dynamic Web Applications. Hybrid coverage based strategy can detect more bugs than statement coverage based and element coverage based strategy.
		Song,B. et al., 2011	Finite State Machine	Finite State Machines to model message requests and responses of sever side and client side.
		Riebisch,M. et al.	Random Testing on Usage Models.	Test Effort Reduction.

		Li,L. et al. , 2008 Youmi, M.E. et al., 2014	Use Case Diagram and Sequence Diagram are used to model Web Application. Fuzzy All-Pairs Testing	Proposed Constraint Message Coverage criterion reduces the number of test cases. Proposed approach shows better results than Equivalence portioning and boundary value analysis technique.
3.	G ray Box Testing	Tarhini,A. et al., 2008 Lebeau,F. et al., 2013 Elbaum ,S.et al., 2003 Peng ,X.et al. , 2011	Event Dependency Graph Model based Vulnerability Testing Hybrid approach combining the Ricca and Tonella's White Box Testing and user session based testing. Genetic Algorithm	Regression Testing of web Application Approach is used to expose Reflected Cross Site Scripting vulnerabilities. Less effort in capturing the sequence of user events. Structural analysis of web application using Request Dependence Graph

Table 4. Comparative Analysis of Various Techniques Adopted for Testing Web based Applications

References

[1] Song,B., Gong,S. Chen, S.(2011). Model Composition and Generating Tests for Web Applications. In: *Seventh International Conference on Computational Intelligence and Security*, p 568-572, IEEE , December.

[2] Di Lucca,G. A., Fasolino, A. R. (2006). Testing Web–Applications: The State of Art and Future Trends. *Information and Software Technology*, 1172-1186, 48 (12), Elsevier.

[3] Naresh Chauhan. (2010). Software Testing: Principles and Practices. Oxford University Press.

[4] Qian, Z., Miao, H., Zeng, H. (2007). A Practical Web Testing Model for Web Application Testing. In: Third International Conference on Signal–*Image Technologies and Internet based System*, 434-441, IEEE , December.

[5] Di Lucca, G. A., Fasolino, A. R., Faralli, F., De Carlini, U. (2002). Testing Web Applications. *In*: *International Conference on Software Maintenance*, 310-319. , IEEE.

[6] Tarhini, A., Ismail, Z., Mansour, N. (2008). Regression Testing Web Applications. *In: International Conference on Advanced Computer Theory and Engineering*. 902-906, IEEE, Decmber.

[7] Liu, C., Kung , D., Hsia, P., Hsu, C. (2000). Object based Data Flow Testing of Web Applications. *In* : *First Asia Pacific Conference on Quality Software*, 7-16, IEEE, October.

[8] Ceri, S., Fraternali, P., Matera, M. (2002). Conceptual Modeling of Data–Intensive Web Applications. *Internet Computing*, 6 (4), 20- 30, IEEE, August.

[9] Riebisch, M., Philippow, I. Gotze. UML–Based Statistical Test Case Generation, http://imamu.edu.sa/DContent/IT_Topics/UML-Based%20Statistical%20Test%20Case%20Generation.pdf.

[10] Li, L., Miao, H., Qian, Z. (2008). A UML–Based Approach to Testing Web Applications. In: International Symposium on Computer Science and Computational Technology, 397- 401, IEEE.

[11] Peng, X. Lu, L. (2011). A new approach for Session–Based Test Case Generation by GA. In: 3rd *International Conference on Communication Softwares and Network*. 91- 96. IEEE, May 2011.

[12] Elbaum, S., Karre, S., Rothermel, G. (2003). Improving Web Application Testing with User Session Data. *In*: 25th *International conference on software Engineering*, 49-59, IEEE.

[13] Service- Oriented Architecture, http:// java.sun.com/developer/Books/j2ee/jwsa/JWSA_CH02.pdf\

[14] Web Applications Security Threats and Counter Measures, http://207.235.13.154/docs/WebSecurityThreats.pdf

[15] Beizer, B.(1990). Software Testing Techniques. Second Edition. Van Nostrand Reinhold Company Limited.

[16] Ricca, F., Tonella, P. (2001). Analysis and Testing of Web applications. *In: Proceedings of the International conference on Software Engineering*, 25-34, IEEE.

[17] Sharma, C., Sabharwal, S., Sibal, R. (2011). Applying Genetic Algorithm for Prioritization of Test case Scenarios Derived from UML Diagrams. *In: International journal of Computer Science Issues*, 8 (3) (2) 433-444.

[18] Zou,Y., Fang,C., Chen, Z., Zhang, X., Zhao, Z. (2013). Regression Testing Web Applications. *In*: SEKE. p. 1-6.

[19] Lebeau, F., Legeard, B., Peureux, F., Vernotte, A. (2013). Model based Vulnerability Testing for Web Applications. *In: Sixth international conference on software testing, verification and validation Workshops*, 445- 452.

[20] Sharma, C., Sabharwal, S., Sibal, R. (2013). A Survey on Software Testing Techniques using Genetic Algorithm. *International journal of Computer science issues*, 10 (1) 381- 393, January.

[21] Youmi, M. E., Falah, B. (2014). Testing Web Applications by Unifying Fuzzy and All-Pairs Techniques. *In: International conference on Multimedia computing and systems*, 547-551, IEEE.