MODEL BASED TESTING OF WEBSITE

Sumit Machra[1] and Narendra Khatri[2]

[1]Department of Computer Engineering, Jodhpur National University, Jodhpur, Rajasthan, India-342001
[2]Department of Electronics & Communication Engineering, C.I.T., Abu road, Rajasthan, India

ABSTRACT

Testing is important for Web applications as they grow more complex day by day which affects daily life. Web application testing frameworks have emerged to help satisfy this need. In this a model is developed for the web applications and test cases are generated by combining operational profile and random testing technique. Those test cases are applied to test the actual website. This research work describes a practical oriented approach to engineer automated tests for web applications with reference to a web application development.

KEYWORDS

Sensitivity, Reliability, Website.

1. INTRODUCTION

Aim of testing is to find out the actual behaviour of a system with the intended behaviour whether they match or not so we can say that the goal of testing is to find out the failure and observable differences between them so that the behaviours of implementation and what is expected on the basis of the specification must be the same as stated. Software Testing is the process to improve the overall quality of the product or service under test. Software testing is done with an objective to understand the risks of software implementation. Testing techniques include the process of finding software bugs in a program or application being under test and sometimes other errors or other defects which are not documented [1].

Web Testing is what we applied testing to Web Applications. In nutshell we are applying Software Testing to Web Applications so as to give it a tag called “Web Testing”. While testing it we are using the randomness to test the particular web application using some model. More to say it’s a Model Based Testing of Web Applications using the random statistical function. Hypothesis: According to Pareto principle, in a system, the usage of all subsystems is not homogenous. In a website, being a system same principle shall hold well. By identifying and improving those few important vitals, compared to other many trivial, results overall improvement in website quality [2].

2. WEBSITE TESTING

Web Testing is the name given to software testing process which when done on web applications. Complete testing of a web-based system before going live can help address issues before the

system is revealed to the public. 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 are used in Web Testing [3], [4].

Here are some factors upon which a web application is tested in general:
Functionality Testing
Usability Testing
Interface Testing
Compatibility Testing
Performance Testing
Security Testing

2.1. Automated model based testing

Automated Testing means that testing assisted with software tools or codes which require no input manually.

There are two general approaches to test automation:

[A] Code-Driven Testing: - The public (usually) interfaces to classes, modules or libraries are tested with a variety of input arguments to validate that the results that are returned are correct.

[B] Graphical User Interface Testing: - A testing framework generates user interface events such as keystrokes and mouse clicks, and observes the changes that result in the user interface, to validate that the observable behavior of the program is correct.

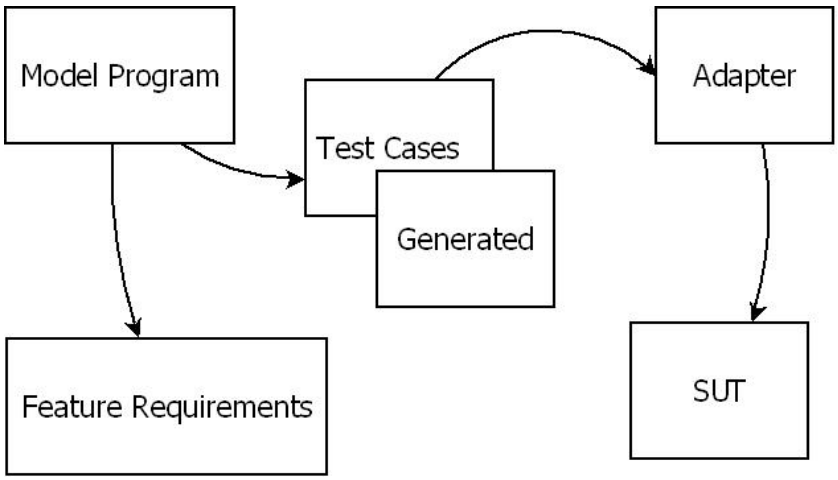

Figure 2.1 Model Based Testing Process

2.2. Random Testing

Random testing quickly generate many tests and easy to implement and reveals system errors put under test. But it sometimes generate many tests which uses the same parts of the code as other tests, as a result its effectiveness is reduced. A Random testing approach is tested by choosing an arbitrary subset of all possible input values. Random testing helps to avoid the problem of only

testing what you know will work. [5], [6], [7].
There are various methods and different techniques for random testing of software applications, web applications and algorithm types that can be used for random testing tools very effectively. Measuring the efficiency and effectiveness of the applied random technique can be done my measurement of testing coverage from one side and failure detection effectiveness from other side.

There are two benefits of using Random Testing in our application:-

1. The coverage that is applied will provide opportunities for future improvement.
2. In some circumstances, random testing methods are more practical than any types of testing.

3. EXPERIMENTAL SETUP

To test the hypothesis an experiment was conducted, in which the website is modeled as a directed graph. Each node of the graph represents a webpage and the directed edge depicts available hyperlink on that page, as shown in figure:-

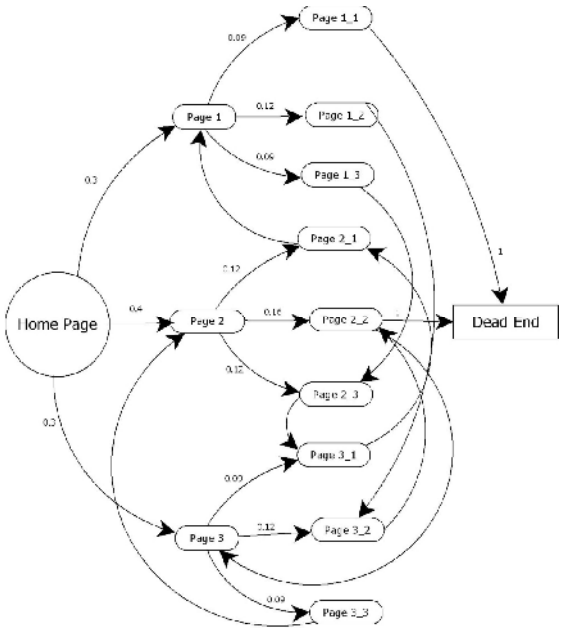

Figure 3.1 Directed Graph of Modeled Website

An initial node is added to represent Home Page i.e. starting node of the graph, all other pages on the site can be reached, as per the hyperlinks available among them. The directed edges are associated with certain probability to represent the chances of traversing that link. When the whole website is visualized in a hierarchical model, it's also called as Site Map, as shown in Figure 3.2 [5] [8].

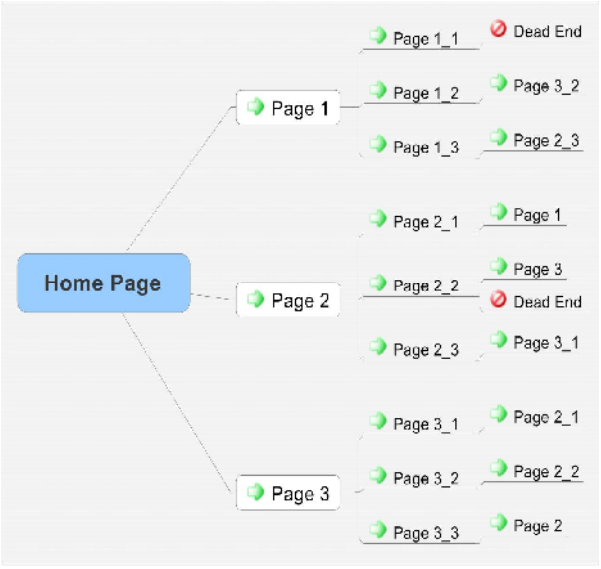

Figure 3.2 Site Map of Website

In the experiment a website consisting of 12 web-pages, spread in three levels, with several back links is developed and used as sample web site. Random walk algorithm guided with transition probabilities were used for traversing the website. Input required for traversing those web pages is automatically generated using random number generator and checking range in which is falls. A random number from the range

between 1 to 100 shall be used for deciding the probability of transition. The traversal is considered to be completed in one of the two situations, either by reaching at last page i.e. Dead-End or by encountering error. In this experiment intentionally an error of missing page is seeded by removing one page which is as shown in the Figure 3.3 below

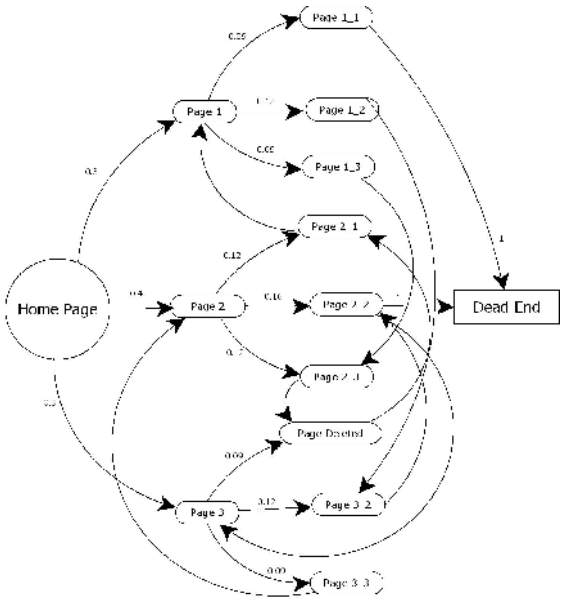

Figure 3.3 Directed Graph of Modeled Website with Injected Error

Process for generating probability based transition:

A random number X is generated in the range of 1 to 100
If ($1 \leq X \leq 30$) then 30% probable event is generated.
If ($30 < X \leq 75$) then 45% probable event is generated.
If ($75 < X \leq 100$) then 25% probable event is generated.

For X with range (1 to N), if ($a < X \leq b$) then (b-a)/N*100 probable even shall be generated.

Using above procedure the model of complete web site is traversed 1000 times and its path of traversals are recoded, further they were used for counting frequency of access of each web page. Same experiment was repeated with error injected web site, and frequency of failure and successful termination is counted. The data generated through traversals were recorded for further analysis.

4. RESULT INTERPRETATION AND ANALYSIS

Statistical Random Testing is done on the website to compare the expected results with the theoretical data. Mathematically it is calculated using the Probability function. Total number of times the test is run is 1000 in each case. Probability of the Page 1 is 0.3%. There is only single link to reach the Page 1 which gives 1000 x 0.3 = 300 times when this page would come. And when run we got 295 times the page which is almost equal. Further taking each of these probabilities we get the following tables.

4.1 Surfing Web Site without Error

The data obtained from experiments was recorded in table 4.1. For each page three columns were populated, describing computed/defined probability, expected number of traversal for each page, and actually obtained number of traversals.

Table 4.1 Coverage without Error

Total Cases	Probability	Expected	Result
Page 1	0.3	300	295
Page 2	0.4	400	395
Page 3	0.3	300	310
Page 1_1	0.09	90	96
Page 1_2	0.12	120	132
Page 1_3	0.09	90	70
Page 2_1	0.12	120	120
Page 2_2	0.16	160	154
Page 2_3	0.12	120	121
Page 3_1	0.09	90	96
Page 3_2	0.12	120	124
Page 3_3	0.09	90	94

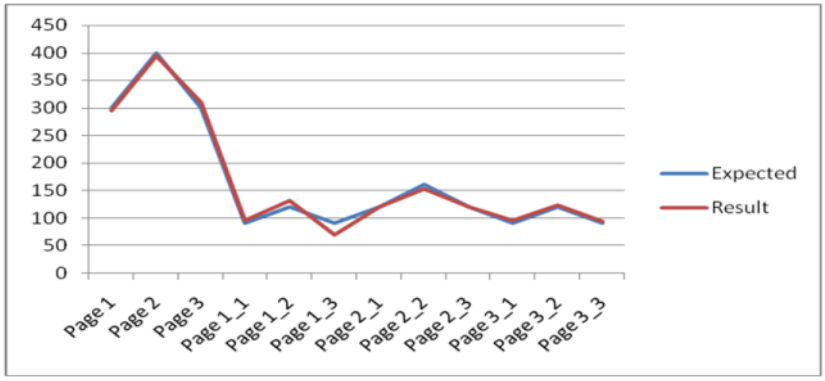

Figure 4.1 Coverage Graph without Error

By observing it is found that count in last two columns are almost same, this denotes that the coverage of website as per expected usage pattern which conforms the coverage criteria. This is visualized in Fig 4.1. The almost overlapping lines denote the degree of matching between expected and obtained coverage.

4.2 Surfing Web Site with Error

The same model was traversed after injecting errors, i.e. removal of one page. To seed an error page 1_3 was removed, since there were two references, this caused two errors practically. Results are recorded and tabulated as shown in table 4.2. The frequency of traversals is as shown below.

Table 4.2 Coverage with Error

Total Cases	Probability	Expected	Result
Page 1	0.3	300	310
Page 2	0.4	400	409
Page 3	0.3	300	281
Page 1_1	0.09	90	90
Page 1_2	0.12	120	127
Page 1_3	0.09	90	96
Page 2_1	0.12	120	125
Page 2_2	0.16	160	158
Page 2_3	0.12	120	127
Page 3_1	0.09	90	X
Page 3_2	0.12	120	108
Page 3_3	0.09	90	83

Total Runs:1000 X=Deleted Page 3_1

The last column contains X to denote that this page could not be reached. Total number of time each page traversed is stored and number of times the removed page was also recorded. It can be observed that the error could not go undetected by random traversals. Moreover, number of times the error occurred was matching with the frequency of that removed page access. This show that the degree of easiness of detecting the error depends on type of error i.e. chances of encountering those errors.

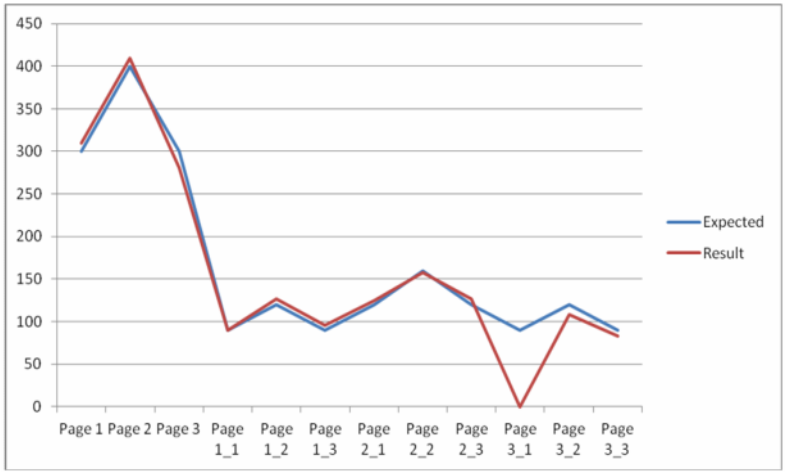

Figure 4.2 Coverage Graph with Error

4.3 Reliability Estimation

Further analyzing the data it is found that webpage is terminated 602 times due to errors and 398 times successfully. In these 602 times, 350 errors are due to Page 2_3 and remaining 252 is due to Page 3.So further contribution of various faulty pages on the failure of the website is accounted as below:-

Table 4.3 Reliability Estimation

Total Cases	Run	Page 2_3	Page 3
DeadEnd	398		
Halt	602	350	252

Total Runs:1000

Figure 4.3 Error Distribution

By visualization the pie chart the distribution of error contribution are as follows:

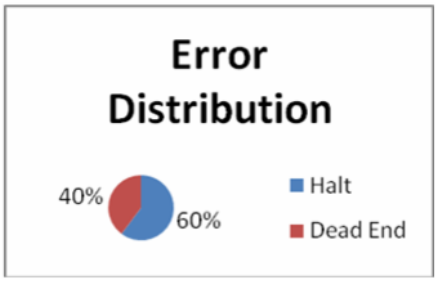

Figure 4.4 Error Contribution

4.4 Sensitivity

It is seen clearly that it is advisable to correct the page or pages which contributes a larger area or larger part of the failure. After correcting one of those faulty pages (Page 2_3) the reliability increases from 40% to 75% which is shown in the figure:

Table 4.4 Sensitivity

Total Cases	Run	Page 3
DeadEnd	748	
Halt	252	252

Total Runs: 1000

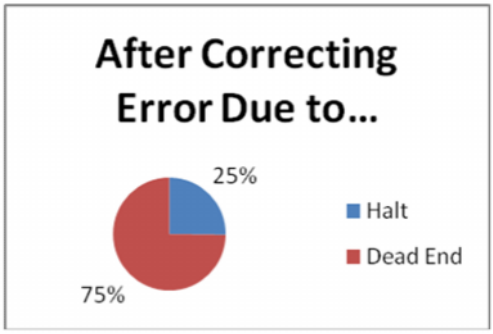

Figure 4.5 Error Correction

From the above diagram it's clear that some of Web Pages are accessed more than others which give us the "Sensitivity" of the Website. Along with those sensitivity of various pages can be used to improvise the overall performance of the website.

4.5 Frequency Distribution

It is clearly visible from the diagram the number of times a web page is accessed that is Frequency Distribution of Web Pages:

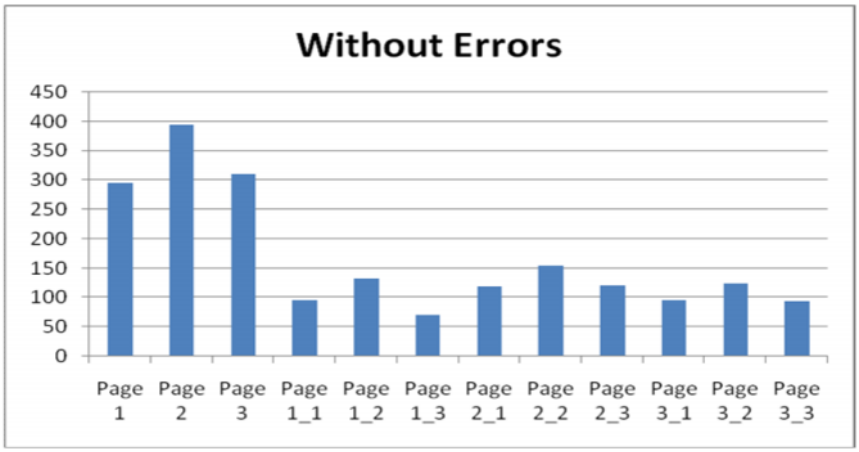

Figure 4.6 Frequency Distribution without Errors

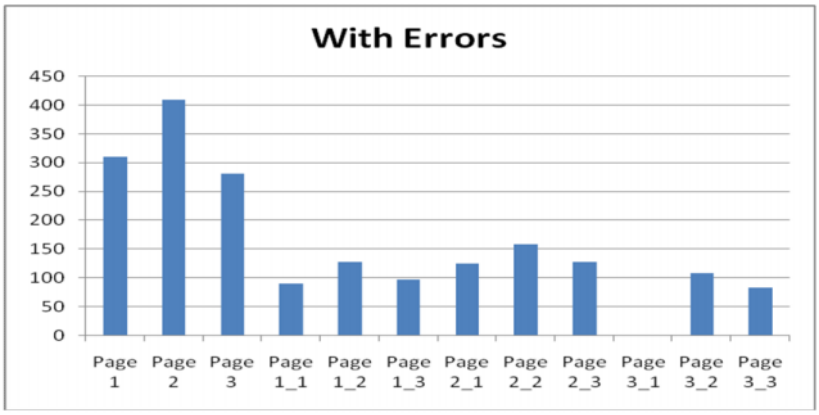

Figure 4.7 Frequency Distribution with Errors

5. CONCLUSIONS

By the comparison of all parameters the testing efforts can be scheduled and prioritize to achieve maximum quality with minimum efforts and schedule.

ACKNOWLEDGEMENTS

We are very thankful for our kind hearted Guruji Kaushal Narayan Joshi Ji, without his motivation this paper will not complete. We especially thanks to Dr. Rajesh Purohit Sir for guiding us.

REFERENCES

[1] B. Beizer. Software Testing Techniques, 2nd Edition, International Thomson Computer Press, 1990.
[2] George Muzea “The Vital Few Versus The Trivial Many:Invest with the insiders, Not the masses” John Wiley Publication Canada ISBN 0-471-68195-4
[3] F. Ricca and P. Tonella “Testing Processes of Web Applications” Annals of Software engineering, (14):93–114, 2002

[4] S. Artzi, A. Kie˙zun, J. Dolby, F. Tip, D. Dig, A. Paradkar and M. D. Ernst “Finding Bugs in Dynamic Web Applications” in International Symposium on Software Testing and Analysis, July 2008

[5] Manar Alalfi, James R.Cordy and Thomas R. Dean “A Survey of Analysis Models in Website Verification and Testing” in Technical Report 2007

[6] P.R. Menon ”FSM-Based Test Generation Methods - A Survey” in international proceedings Euro ASIC-92 at France ISBN 0-8186-2845-6

[7] S.R.Dalal ,A.Jain,N.Karunaithi and B.M. Horowitz “Model Based Testing in Practice” to appear in proceedings of ICSE,99 (ACM Press)

[8] Nan Liu and Christopher C. Yang “A Link Classification Based Approach to Website Topic Hierarchy Generation” in proceedings of the International World Wide Web Conference (WWW'07) 2007

Authors

Sumit Machra, M.Tech (Computer Science), Jodhpur National University, Jodhpur, Associated with teaching since 2007, His area of interest is Software Engineering and Cryptography & network security.

Narendra Khatri, Asst. Prof. in Department of Electronics and Communication Engg., C.I.T., Aburoad, Completed M.Tech. (Communication Engineering) from CTAE, Udaipur and teaching Since2008.His area of interest is Communication, Signal and Image Processing.