Reusing Test Path in White Box Testing

[1]U.Santhosh, [2] K.Vijayakumar
[1]M.Tech Student, [2]Assistant Professor
[1,2]Dept of Software Engineering, SRM University, Kattankulathur, Chennai, India-603203
[1] china.santhosh@gmail.com , [2]kmk.vijay@gmail.com

***Abstract*- Software complexity and the malwares are increasing exponentially since the number of web users is increasing exponentially. Therefore exhaustive and extensive testing of websites has become a necessity today. But testing a website is not 100% exhaustive due to page explosion problem. The basis test paths obtained from the Page-Test-Trees (PTTs) are reused for white box testing of websites. This saves significant amount of time required to generate test paths and hence test cases as compared to the existing approaches of white box testing. The cost and efforts are also minimized. The proposed technique ensures better website testing coverage as white box testing provides better results than black box testing.**

***Keywords* – Page test tree, Page flow diagram, Test path.**

1. INTRODUCTION

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include, but are not limited to the process of executing a program or application with the intent of finding software bugs (errors or other defects).

A primary purpose of testing is to detect software failures so that defects may be discovered and corrected. Testing cannot establish that a product functions properly under all conditions but can only establish that it does not function properly under specific conditions. The scope of software testing often includes examination of code as well as execution of that code in various environments and conditions as well as examining the aspects of code: does it do what it is supposed to do and do what it needs to do. In the current culture of software development, a testing organization may be separate from the development team. There are various roles for testing team members. Information derived from software testing may be used to correct the process by which software is developed.

The box approach

Software testing methods are traditionally divided into white- and black-box testing. These two approaches are used to describe the point of view that a test engineer takes when designing test cases.

White-Box testing

White-box testing (also known as clear box testing, glass box testing, and transparent box testing and structural testing) tests internal structures or workings of a program. In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases [3]. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs. This is analogous to testing nodes in a circuit, e.g. in-circuit testing (ICT).

While white-box testing can be applied at the unit, integration and system levels of the software testing process, it is usually done at the unit level. Techniques used in white-box testing include:

- Code coverage – creating tests to satisfy some criteria of code coverage (e.g., the test designer can create tests to cause all statements in the program to be executed at least once)
- Fault injection methods – intentionally introducing faults to gauge the efficacy of testing strategies

Code coverage tools can evaluate the completeness of a test suite that was created with any method, including black-box testing. This allows the software team to examine parts of a system that are rarely tested and ensures that the most important function points have been tested. Code coverage as a software metric can be reported as a percentage for:

- *Function coverage*, which reports on functions executed
- *Statement coverage*, which reports on the number of lines executed to complete the test

100% statement coverage ensures that all code paths, branches (in terms of control flow) are executed at least once.

Black-box testing

It treats the software as a "black box", examining functionality without any knowledge of internal implementation. The tester is only aware of what the software is supposed to do, not how it does it. Black-box testing methods include: equivalence partitioning, boundary value analysis, all-pairs testing, state transition tables, decision table testing, fuzzy testing, model-based testing, use case testing, exploratory testing and specification-based testing.

Unit testing

Unit testing, also known as component testing refers to tests that verify the functionality of a specific section of code, usually at the function level. In an object-oriented environment, this is usually at the class level, and the minimal unit tests include the constructors and destructors.

Integration testing

Integration testing is any type of software testing that seeks to verify the interfaces between components against a software design.

In general, black box testing is applied to test web components (Graphical User Interfaces) and white-box testing is done at later stages when source code starts creeping in. In this paper, the test-paths (and hence test cases) so generated after black-box testing are reused to do subsequent white-box testing using these paths only. This tests websites extensively. This has many significant benefits- Test data is already available and also the test paths. Hence, we need to test source code at these points (nodes) only. Branching nodes can be pruned; paths can be pruned [4], if required.

Many web applications are developed and tested without any malwares. Earlier, test paths for white box testing and black box testing have been identified separately. Therefore finding the same test paths in black box testing and white box testing is a repeated process. Instead, the test paths found during black box testing can be reused in white box testing and hence cost, effort and time are minimized.

2. RELATED WORK

Fillipo Ricca and Paolo Tonella stress that Web page is a central entity in any website. Web pages can be static or dynamic. They use white-box testing criteria like page testing, All-uses testing and All-paths testing criteria are also applied on websites. They discuss about path expressions and node-reduction algorithms. They have developed tools names as Reweb and Test Web. The All-path testing criterion is achieved. It is restricted to independent paths. The proposed techniques were applied to several real world Web applications. Results suggest that an automatic support to the verification and validation activities can be extremely beneficial. In fact, it guarantees that all paths in the site which satisfy a selected criterion are properly exercised before delivery. The high level of automation that is achieved in test case generation and execution increases the number of tests that are conducted and simplifies the regression checks. [1].

Eric Y. K. Chan and Y. T. Yu used a classification tree method (CTM) as the black-box criterion and “same path” as white box criterion. CTM are actually the hierarchical trees. However, no web pages are considered. This paper introduces three alternative sampling methods and a way for safely omitting as many test cases as possible from the initial test suite based on the class pairs that are judged to be co path [2]. Empirical study shows that all these sampling methods help to improve the test saving ratio achieved by path-based partial dynamic analysis. Among them, overlapped sampling usually produces the smallest final test suite and, in most of the time, retains the comprehensiveness of the test suite as measured by the path retention percentage. The experiments also show no significant difference in the effectiveness of the independent and disjoint sampling methods. [2].

Lixin Wang uses a method of dividing the program under test (PUT) into a several program segments with small cyclomatic complexity. This paper analyzes the relationship between the input variables and the fragments, gives the approach of processing the split point and illustrates the benefits of the method for generating test data both theoretically and experimentally. For each fragment, the method decreases the complexity of large program while structural testing, increases the efficiency of the test data generation. But for every segment the paths are derived again hence this paper focus only on white box testing alone [3].

Bo Song and Huaikou Miao developed navigation model to generate test. But it is a directed graphs and graphs have problems like cycle. So, EFSM is not easy to use. It will generate a set of test cases with duplicity. So, a FSM test tree (FSM-TT) is proposed. Shorter test sequences can be generated without loss of states. But this sequence of paths has not been utilized further for generation of black box test paths of testing [4].

A PTT is used to establish path expressions [5] where a path expression is defined as an algebraic representation of paths in a tree. Variables in the path expression are links. They can be combined using operators like + and *, that denote selection and loop respectively. Brackets may used to group sub path expressions. For example, k1->k2->k5->(k9+k10) is a path expression [5].

Du Qingfeng, Dong Xiao use basis path testing alone to show how this method can be used for testing codes having switch statements also in addition to the 'if statements'. They also used only white box testing approach [8].

3. PROPOSED METHODOLOGY

Page flow diagram (PFD)

Page flow diagram(PFD)[4] which is used to redirect to the desired pages, these graphs are sometimes problematic and it complicates the testing process and hence PFD is converted to page test tree(PTT)[5].

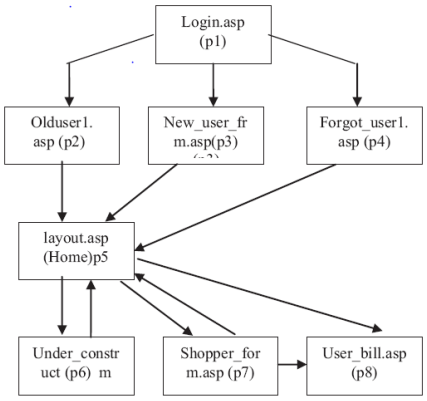

Fig 1: Sample Page Flow Diagram

Page Test Tree (PTT)

From PTT shorter paths are generated by the root link of the tree to the tail link [4]. For smaller web applications the links are given manually and test paths are found out. The main objective of the present paper is, Now–a-days; websites are very complex as they have numerous pages and links. In such a scenario[10], it is difficult to proceed as the PFDs and PTTs will be very complex, large and unmanageable.

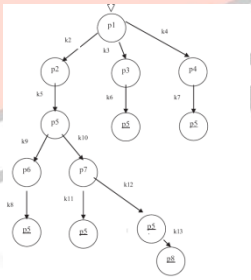

Fig 2: Sample Page Test Tree

CONTRIBUTION

The test paths obtained from black box testing are reused for white box testing of websites. The total time taken to find the test paths when white box testing and black box testing are done separately is compared with the total time taken when the test paths obtained from black box testing are reused in white box testing.

Now–a-days, websites are very complex as they have numerous pages and links. In such a scenario, it is difficult to proceed as the PFDs and PTTs will be very complex, large and unmanageable. Henceforth, divide and- conquer strategy may be followed to divide a website into sub-websites and recursively applying this model of testing. For each sub-website, a corresponding PFD and PTT are found out and later it is integrated together.

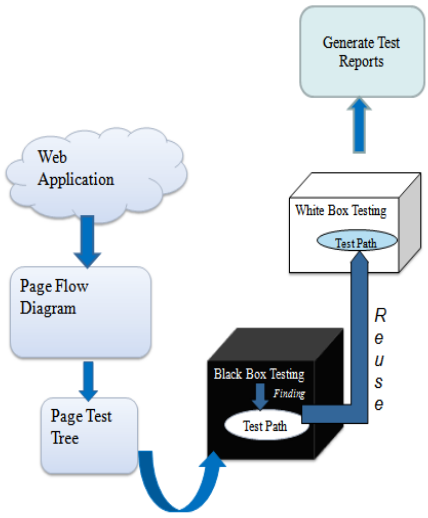

Fig 3: Flow Diagram

5. CONCLUSION

In this paper we have shown that test paths derived from the black box testing of websites can further be utilized to do white box testing of websites. Earlier, test paths for white box testing and black box testing have been identified separately. Therefore finding the same test paths in black box testing and white box testing is a repeated process. Instead, the test paths found during black box testing can be reused in white box testing and hence cost, effort and time are minimized. So in this paper by using divide and conquer strategy may be followed to divide a web site into sub websites and recursively applying this model for testing. By using this methodology test case, cost, time and effort are reduced.

6. FUTURE WORKS

There are several directions for future works. Using this methodology for AJAX based applications, test case generations, test case reduction and test driven development based on the test paths are some of them.

REFERENCES

[1] Fillippo Ricca and Paolo Tonella, "Analysis and Testing of Web Applications", ITC-irst, Centro per la Ricerca Scientificae technologica , Italy, IEEE 2001.
[2] Eric Y. K. Chan and Y. T. Yu, "Evaluating several Path-based Partial Dynamic Analysis Methods for selecting Black-Box generated Test Cases", Fourth International Conference on Quality Software, IEEE 2004.
[3] Lixin Wang, "A Program Segmentation Method for Testing Data Generating Based on Path Coverage", IEEE, 2010.
[4] Bo Song, HuaiKou Miao, "Modelling Web Applications and Generating Tests: A combination and Interactions-guided Approach", 3rd IEEE International Symposium on Theoretical Aspects of Software Engineering, IEEE 2009.
[5] B. Beizer, "Software Testing Techniques", Second Edition, dreamtech Press, 2009.
[6] Zhongsheng Qian, Huaikou Miao, Hongwei Zeng, "Apractical web testing model for web application testing", Third International IEEE conference on Signal-Image Technologies nd Internet-Based System, 2008.
[7] T. Y. Chen and P. L. Poon "A White-Box-Oriented Approach for Selecting Black-Box-Generated Test Cases, IEEE, 2000
[8] Du Quingfeng, Dong Xiao, "An Improved Algorithm for basis Path Testing", IEEE 2011.
[9] Rajiv chopra, "Software Testing-A Practical Approach", Fourth Edition, Katsons, 2013.
[10] Jingxian Gu, Lei Xu, Baowen Xu, Hongji Yang, "An Extended MMpath Approach to Component-based Web Application Testing", 12th IEEE International Workshop on Future Trends of Distributed Computing Systems, IEEE 2008.
[11] N. Mansour and M. Salame, "Data generation for path testing," Software Quality Journal, vol. 12(2), Springer Netherlands, 2004, pp. 121-134