A Review of Testing Technology in Web Application System

Anirban Bhar[1], Soumya Bhattacharyya[2], Sujata Kundu[3] , Shyamapriya Chatterjee[4]

[1,2,3,4]*Department of Information Technology, Narula Institute of Technology, Kolkata, India.*

Abstract - *Software testing is important activity in Software Development Life Cycle. Software testing is the process of assessing the functionality and correctness of a program through execution or analysis. Software-testing process is expensive and consumes a lot of time through software development life cycle. As software systems grow, manual software testing becomes more and more difficult. Therefore, there was always a need to decrease. Software testing that is not fully exhaustive can only suggest the presence of flaws and cannot prove their absence; moreover it is impossible to completely test an application. This work aims at the process of identifying the bottlenecks that are very common to test in web application systems, bottlenecks that are application specific and testing type specific, analyze the reasons and provide recommendations that will make Software Testing [1] to happen without bottlenecks or with minimum bottlenecks.*

Key Words: SDLC, Testing, Web application systems, Bottlenecks.

INTRODUCTION

Software testing plays a vital role in the software development life-cycle to recognize the difficulties in the process very well [2]. The objective of testing is to find problems and fix them to improve quality (Fig.1). Software testing typically represents 40% of a software development budget.

There are four main objectives of software testing [3]:
1) Demonstration: It demonstrates functions under special conditions and shows that products are ready for integration or use.
2) Detection: It discovers defects, errors and deficiencies. It determines system capabilities and limitations, quality of components, work products and the system.
3) Prevention: It provides information to prevent or reduce the number of errors clarify system specifications and performance. Identify ways to avoid risk and problems in the future.
4) Improving Quality: By doing effective testing, we can minimize errors and hence improve the quality of software [4].

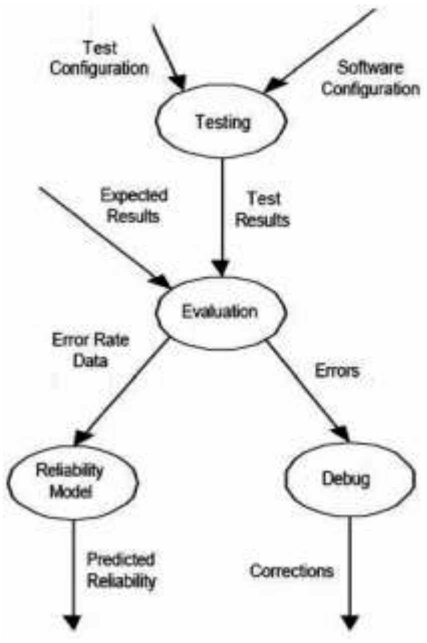

Fig.1

Testing is the primary method that the Organization uses to evaluate software under development. Software quality is nothing but the fulfillment of the end user requirements. Main areas where developers need to focus upon are quality assurance, quality planning and quality control. Software testing plays an important role in these areas by providing the way of reducing the bugs from the application.

Software testing techniques is categorized in two main areas: Manual testing and automated testing.

Automation Testing uses automation tools to execute test cases. In manual testing, test cases are executed by a human tester and software. Automated testing is significantly faster than a manual approach. Automation is one major solution for reducing high testing effort. Automating certain manual tasks from software testing process can save a lot of testing time. It can help in performing repetitive tasks more quickly than manual testing.

Manual Testing vs. Automated Testing

In manual testing (as the name suggests), test cases are executed manually (by a human, that is) without any support from tools or scripts. But with automated testing, test cases are executed with the assistance of tools, scripts, and software.

Testing is an integral part of any successful software project. The type of testing (manual or automated) depends on various factors, including project requirements, budget, timeline, expertise, and suitability. Three vital factors of any project are of course time, cost, and quality – the goal of any successful project is to reduce the cost and time required to complete it successfully while maintaining quality output. When it comes to testing, one type may accomplish this goal better than the other.

In short, manual testing is best suited to the following areas/scenarios:

Exploratory Testing: This type of testing requires the tester's knowledge, experience, analytical/logical skills, creativity, and intuition. The test is characterized here by poorly written specification documentation, and/or a short time for execution. We need the human skills to execute the testing process in this scenario.

Usability Testing: This is an area in which you need to measure how user-friendly, efficient, or convenient the software or product is for the end users. Here, human observation is the most important factor, so a manual approach is preferable.

Ad-hoc Testing: In this scenario, there is no specific approach. It is a totally unplanned method of testing where the understanding and insight of the tester is the only important factor.

Automated testing is the preferred option in the following areas/scenarios:

Regression Testing: Here, automated testing is suitable because of frequent code changes and the ability to run the regressions in a timely manner.

Load Testing: Automated testing is also the best way to complete the testing efficiently when it comes to load testing. Learn more about load testing with our best practices guide here.

Repeated Execution: Testing which requires the repeated execution of a task is best automated.

Performance Testing: Similarly, testing which requires the simulation of thousands of concurrent users requires automation.

Web Application Architecture

Web application architecture defines the interactions between applications, middleware systems and databases to ensure multiple applications can work together. When a user types in a URL and taps "Go," the browser will find the Internet-facing computer the website lives on and requests that particular page.

The server then responds by sending files over to the browser. After that action, the browser executes those files to show the requested page to the user. Now, the user gets to interact with the website. Of course, all of these actions are executed within a matter of seconds. Otherwise, users wouldn't bother with websites.

What's important here is the code, which has been parsed by the browser. This very code may or may not have specific instructions telling the browser how to react to a wide swath of inputs. As a result, web application architecture includes all sub-components and external applications interchanges for an entire software application.

Of course, it is designed to function efficiently while meeting its specific needs and goals. Web application architecture is critical since the majority of global network traffic, and every single app and device uses web-based communication. It deals with scale, efficiency, robustness, and security.

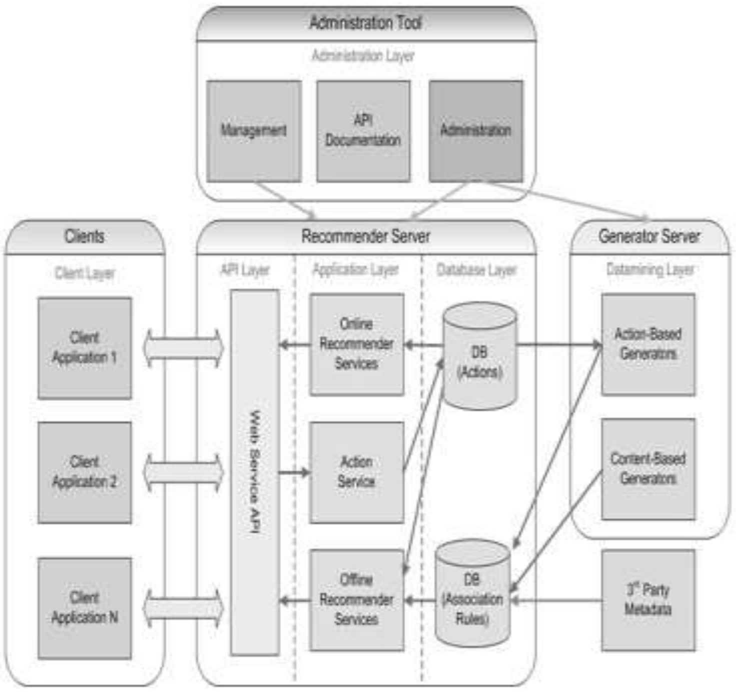

Fig.2

Classes of Threats

Here are the different types of threats which can be used to take advantage of security vulnerability.

Privilege Elevation: Privilege elevation is a class of attack where a hacker has an account on a system and uses it to increase his system privileges to a higher level than he/she was not meant to have. If successful, this type of attack can result in a hacker gaining privileges as high as root on a UNIX system. Once a hacker gains super-user privileges, he is able to run code with this level of privilege and the entire system is effectively compromised.

SQL Injection: SQL injection is the most common application layer attack technique used by hackers, in which malicious SQL statements are inserted into an entry field for execution. SQL injection attacks are very critical as an attacker can get critical information from the server database. It is a type of attack which takes the advantage of loopholes present in the implementation of web applications that allows a hacker to hack the system. To check the SQL injection we have to take care of input fields like text boxes, comments, etc. To prevent injections, special characters should be either properly handled or skipped from the input.

Unauthorized Data Access: One of the more popular types of attacks is gaining unauthorized access to data within an application. Data can be accessed on servers or on a network.

Unauthorized access includes:

- Unauthorized access to data via data-fetching operations
- Unauthorized access to reusable client authentication information by monitoring the access of others
- Unauthorized access to data by monitoring the access of others

URL Manipulation: URL manipulation is the process of manipulating the website URL query strings & capture of the important information by hackers. This happens when the application uses the HTTP GET method to pass information between the client and the server. The information is passed in parameters in the query string. The tester can modify a parameter value in the query string to check if the server accepts it.

Denial of Service: A denial-of-service (DoS) attack is an explicit attempt to make a machine or network resource unavailable to its legitimate users. Applications can also be attacked in ways that render the application, and sometimes the entire machine, unusable.

Data Manipulation: In data manipulation, a hacker changes data used by a website in order to gain some advantage or to embarrass the website's owners. Hackers will often gain access to HTML pages and change them to be satirical or offensive.

Identity Spoofing: Identity spoofing is a technique where a hacker uses the credentials of a legitimate user or device to launch attacks against network hosts, steal data or bypass access controls. Preventing this attack requires IT-infrastructure and network-level mitigations.

Cross-Site Scripting (XSS): Cross-site scripting is a computer security vulnerability found in web applications. XSS enables attackers to inject client-side script into Web pages viewed by other users and trick a user into clicking on that URL. Once executed by the other user's browser, this code could then perform actions such as completely changing the behavior of the website, stealing personal data, or performing actions on behalf of the user.

Web Based Application Testing Methodologies

Different types of testing techniques like coverage testing, structural testing, statistical testing, combinatorial interaction testing, penetration testing, Search based software engineering testing, Unique Input/ Output method using Genetic Algorithms [5], Web application Slicing [6], [7], Hierarchical testing [8], Bypass testing, Cross Browser compatibility testing [9], Leveraging User session Data Testing have been presented by various researchers in the context of web application testing [10].

Structural Testing - Data flow analysis on web applications is performed and model for testing the application is built dynamically.
Statistical Testing – Input Sequence is generated to test the interactions with web applications based on the profile use of the web application.
Mutation Testing – The technique of introducing faulty code (called mutants) into the source code deliberately at predetermined points and testing the software to uncover any unknown errors. It is one of the effective coverage criterion techniques for testing of web applications [11].
Combinatorial Interaction Testing - Using a combination of different techniques by first designing a unique input space matrix for the web application
Penetration Testing - Automated tests which are run simulating the active attacks to expose the susceptibilities of the web applications.
Search Based Software Engineering Testing - Exploration of solutions within a state space and calculating a fitness function to the solution iteratively until we arrive at a most optimal solution. The technique is employed for branch coverage of web applications.
Using UIO and Genetic Algorithms - Path selection is done based on a unique input/output (UIO) algorithm and

automatic test case generation using Genetic Algorithms which results in the best test sequences.
GUI Interaction Testing - GUI widgets events sequences are performed and the web application tested for correctness by observing the state of the GUI widgets.
Web Application Slicing - Reduced web application which behaves completely as the original one with respect to some criterion and performing the testing.
Cross Browser Compatibility testing - Subjecting web applications to deployment across different browsers for adherence to expected results.
Hierarchical Strategy - High level operational profile is developed enumerating frequency of use of operations and a high level function group to thoroughly test such an operation or related components is done.
Bypass Testing - Bypass client side checking by providing invalid inputs to web application to check correctness and security of the web application.
Leveraging User Session Data - Test cases are generated by applying strategies to collected user interactions in the form of URL's and name-value pairs.
Browser Fuzzing By Scheduled Mutation - Browsers are validated by using the static and dynamic ways, the former based on the input format while the latter randomly executing instructions giving one input at a time.
Invariant Based Technique - Testing the web application by crawling the web pages, and formally designing a state flow graph with all the possible user interaction sequences resulting in the possible user interface states.
Model Based Testing Technique - Web application is reduced to a state transition graph and navigation through links is tested to ascertain correct behaviour of the web application.

CONCLUSION

In this paper we introduced a set of related search based testing algorithms, adapted for web application testing and augmented the approach with static and dynamic seeding. This paper also shows, there are many issues raised by web application testing, such as dynamic type binding and user interface inference that create novel challenges for search based testing that have not previously been addressed. Dynamically mined value seeding approach can significantly reduce effort and increase effectiveness for web application testing.

REFERENCES

[1] Brain Marick, Classic Testing Mistakes, RSTAR97 conference
[2] Introduction to software testing available at http://www.onestopsoftwaretesting.com/introductionand-importance-of-software-testing-in-sdlc/
[3] Anju Bansal, International Journal of Computer Science and Mobile Computing, Vol.3 Issue.6, June- 2014, pg. 579-584
[4] F. Saglietti, N. Oster, and F. Pinte, "White and grey-box verification and validation approaches for safetyand security-critical software systems," Information Security Technical Report, vol. 13, no. 1, pp. 10–16, 2008.
[5] Poonam Soni, Dr. Sanjay Tyagi, ―Testing Web Applications Using UIO with GA‖, The International Journal of Soft Computing and Software Engineering Volume 3, Issue 5, May 2013 ISSN: 2277 128X.
[6] Lei Xu, Baowen Xu, Zhenqiang Chen, Jixiang Jiang, Huowang Chen, "Regression Testing for Web Applications Based on Slicing", Proceedings of the 27th Annual International Computer Software and Applications Conference (COMPSAC'03) 0730-3157/03 2003 IEEE.
[7] Filippo Ricca, "Analysis, Testing and Re-structuring of Web Applications", Proceedings of the 20th IEEE International Conference on Software Maintenance (ICSM'04) 1063-6773/04 2004 IEEE.
[8] Jeff Tian, Li Ma, Zhao Li and A. G¨unes¸ Koru, "A Hierarchical Strategy for Testing Web-Based Applications and Ensuring Their Reliability", Proceedings of the 27th Annual International Computer Software and Applications Conference (COMPSAC'03) 0730-3157/03 2003 IEEE.
[9] Ali Mesbah, Mukul R. Prasad, ―Automated Cross-Browser Compatibility Testing‖, ICSE '11, May 21–28, 2011, Waikiki, Honolulu, HI, USA.
[10] Sebastian Elbaum, Member, Gregg Rothermel, Srikanth Karre, Marc Fisher II. ―Leveraging User-Session Data to Support Web Application Testing‖, IEEE Transactions on Software Engineering, Vol. 31, No. 3, March 2005.
[11] Hanh Le Thi My, Nguyen Thanh, Khuat Thanh, ―Survey on Mutation-based Test Data Generation‖, International Journal of Electrical and Computer Engineering (IJECE), Vol 5, No 5, 2015.