ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 3, Issue 8, August 2016

Exploiting the Capabilities of Automated Testing In Software Testing

Sidharath Singla
Asst. Prof. A.S. College, Khanna

Abstract: **- Software has infiltrated almost every area of industry and has over the years become more and more widespread as a crucial component of many systems. System failures can be very costly and sometimes means a lot of human lives for critical systems like (flight control, nuclear reactor, medical applications etc.). So there arises a need for some kind of system failure prevention. One way to ensure system's reliability is to extensively test the system. Software testing consumes at least half of the labor expended to produce the software**

Index Terms: — Cloud Testing, Janova, Nessus, Ranorex, Selenium, Software Testing, Test Cases, Testing Stages, Testing Tools, V & V

I. INTRODUCTION

Software testing is an investigation conducted to provide stakeholders with information about the quality of the software under test. It provides an objective, independent view of the software to allow the business to appreciate and understand the risk of software implementation. Test techniques include the process of executing a program or application with the intention of finding software bugs (an error, flaw, failure or fault in a computer program or system that causes it to produce an incorrect or unexpected result or to behave in an abnormal way) made by people in either program's source code or its design. As the number of possible tests for even simpler software components is practically infinite, all software testing uses some strategies to select tests that are feasible for the available time and resources. The job of testing is an interactive process as when one bug is fixed, it can illuminate other, deeper bugs, or can even create new ones. Software testing can be conducted as soon as executable software (even partially complete) exists. The overall approach to software development often determines when and how testing is to be conducted for example in a phased process, most of the testing occurs after the system requirements have been defined and then implemented in testable programs. Also testing should not be a distinct phase in system development but should be applicable throughout the system design, development and maintenance phases. It should be noted that testing cannot establish that a product functions properly under all conditions but can only establish that it does not function properly under specific conditions. Also testing examines the aspect of code: does it do what it is supposed to do and do what it needs to do. Testing has two distinct goals:-

1) To demonstrate to the developer and the customer that the software meets its requirements i.e. Validation Testing.

2) To discover situations in which the behavior of the software is incorrect, undesirable or does not conform to its specifications, which are the consequences of software defects i.e. Defects Testing.

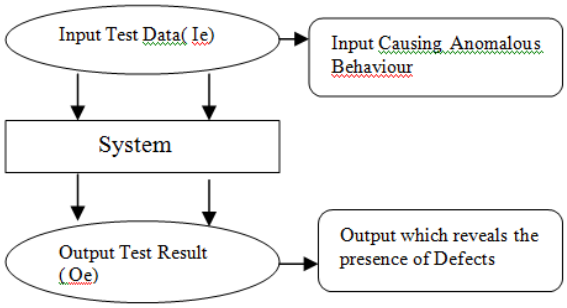

Fig 1:- Input-Output model for program testing

II. VERIFICATION & VALIDATION TECHNIQUE

Testing is a part of broader process of software verification and validation (V & V) often confusing but not same.

Validation: **-** Are we building the right product? (i.e. to check if software meets its customer specifications.)

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 3, Issue 8, August 2016

Verification: **-** Are we building the product right? (i.e. to check if software meets its both functional and non-functional requirements.)

Verification and Validation processes are concerned with checking that software being developed meets its specifications and delivers the functionality expected by the people paying for the software. The ultimate goal of Verification and Validation processes is to establish confidence that the software is 'fit for purpose'. This means that the system must be good enough for the intended use. The level of required confidence depends on the system's purpose, the expectation of the system users, and the current marketing environment for the system. Besides software testing, the Verification & Validation process may involve software inspections, which analyze and check system requirement, design models, the program source code and even proposed system tests. These are so called 'static' V & V techniques in which you don't need to execute the software to verify it. The following figure shows that software inspection and testing support V & V at different stages in software process.

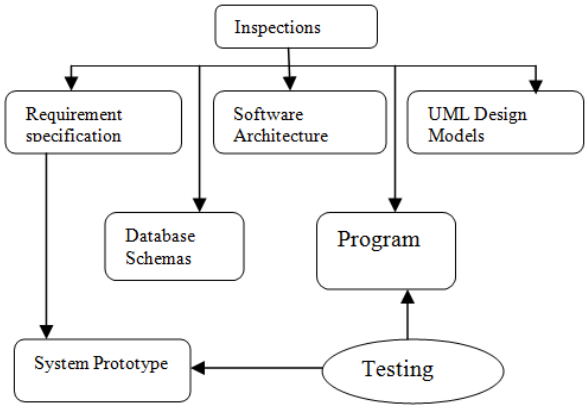

Fig 2:- Inspections & Testing

Program inspections are an old idea and there have been several studies and experiments that have demonstrated that inspections are more effective for defect discovery than program testing. Fagan (1986) reported that more than 60% of the errors in the program can be detected using informal program inspections. In the Cleanroom process (Prowell et al., 1999), it is claimed that more than 90% of the defects can be discovered in program inspections. However program inspections cannot replace software testing as inspections are not good for discovering defects that arise of the unexpected interactions between different parts of the program, timing problems, or problems with system performance. Furthermore, especially in small companies or development groups, it can be difficult and expensive to put together a separate inspection team as well as all potential members of the team that may also be software developers.

III. TESTING STAGES

Typically, a commercial software system has to go through three stages of testing:-

1. ***Development Testing***: where the system is tested during development to discover bugs and defects. System designers and programmers are likely to be involved in the testing process. It is carried out three levels of granularity: Unit Testing, Component Testing, and System Testing.

2. ***Release Testing***: where a separate testing team tests a complete version of the system before it is released to the users. The main aim of the release testing is to check that the system meets the requirements of system stakeholders. It includes Requirement-based Testing, Scenario Testing, and Performance Testing.

3. ***User Testing***: where users or potential users of the system testes the system in their own environment. For software products, the 'user' may be an internal marketing group who decide if the software can be marketed, released, and sold. It includes Alpha Testing, Beta Testing, and Acceptance Testing.

The testing process usually involves a mixture of **manual** and **automated** testing. In manual testing, a tester runs the program with some test data and compares the results to their expectations. They note and report discrepancies to the program developers. In automated testing, the tests are encoded in a program that is run each time the system under development is to be tested. This is usually faster than the manual testing, especially when it involves regression testing — re-running previous tests to check that the changes to the program have not introduced new bugs. The use of automated testing has increased during the past few years. However testing can't be fully automated as automated tests only check that a program does what it is supposed to do. It is practically impossible

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 3, Issue 8, August 2016

to use automated testing to tests the systems that depends on how the things look (e.g., a graphical user interface), or to test that a program does not have unwanted side effects.

IV. CHOOSING TEST CASES

Testing is an expensive and time consuming process, so it is important that you choose effective test cases. Effectiveness means two things:

1) The test cases should show that, when used as expected, the system that you are testing does what it is supposed to do.

2) If there are defects in the system, these should be revealed by test cases. For this two kind of test cases can be written. The first of these should reflect normal operations of a system and should show that the system works, it should use normal inputs to check that the system works properly and initializes all relevant components. The other kind of test case should be based on testing experience of where common problem arise, it should use abnormal inputs to check that these are properly processed and do not crash the system.

V. CLOUD TESTING

Cloud Testing is a form of testing in which applications use cloud computing environment and cloud infrastructure. The cloud infrastructure includes hardware and bandwidth that closely simulates real world conditions and parameters. Testing a cloud includes availability, security, performance, interoperability, disaster recovery and multi-tenancy testing. Testing is performed in three areas of cloud that includes infrastructure, platform and service. Organizations and enterprises face a lot of issues with testing such as limited test budget, lack of meeting the prescribed timelines in addition to high cost per test. Cloud infra offers unlimited storage, quick availability and can be scaled as desired. With no upfront investment, cloud is turning out to be the really cost-effective solution offering multiple choices for software developers. Cloud-based Testing has emerged as the best way of testing with a compelling combination of lower costs, pay-per-use models, scalability and flexibility it offers. It can also address the ramping demand for sophisticated test environments.

Cloud Testing is a new approach to quickly and affordably prepare web and mobile sites for daily and peak traffic that tests using cloud-based resources, capitalizing on the low costs, high scalability, and rapid setup and tear-down of the cloud. With Cloud Testing, web teams are empowered with a FREE testing platform to test for products.

VI. AUTOMATED TESTING TOOLS

1. Ranorex

Ranorex is easy-to-use test automation software for developing and managing projects in teams made up of both testers and developers. This is a simple, comprehensive and cost effective tool used for automatic testing. It is a better alternative as testing tools because it tests applications from a user's perspective, using standard language and common programming techniques like C# and VB.net. It does not require understanding a scripting language, because it is coded in pure .net code. Any one of the three languages, VB.net, C# and Iron Python can be used. It is used by a lot of commercial software companies and enterprises around the globe. A step-by-step wizard helps you to set up the test environment and quickly get started. Non-programmers can use the script-free drag & drop functionality, whereas professional programmers can use an API for C# and VB.NET to enhance their test suites and recordings. Ranorex is both easy-to-use and affordable, making it the best tool for small and large teams. One license includes all tools, technologies and updates for your automated testing. The friendly learning curve allows you to get started quickly and become more productive instantly. Sign up for live webinars and take part in basic or advanced training sessions offering a possibility for live questions and answers.

Ranorex helps to increase the test coverage through the automation of different types of tests as follows:

- Data Driven Test
- Keyword Driven Test
- Regression Testing
- Automated Functional Testing
- Cross Platform & Device Testing
- Automated GUI Testing
- Cross Browser Testing

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 3, Issue 8, August 2016

Services Provided by Ranorex

1. Helps to Manage and develop test automation projects with Ranorex Studio.
2. Helps to Record, replay and edit user actions with Ranorex Recorder.
3. Helps to minimize your maintenance effort with Ranorex Object Repository.
4. It Keeps Your Tests Running With Ranorex Repositories.
5. Helps in Element Browser within Ranorex Spy.
6. Ready to Use GUI Objects.
7. Reproduce bugs and maintain tests with Ranorex Click & Go Test Reports.
8. Explore and analyze your software application with Ranorex Spy.
9. Explore and Analyze Applications.
10. Ranorex Snapshot File.

Features Of Ranorex

1. GUI object Recognition.
2. Reusable Code Modules.
3. Early Bug Finding.
4. Seamless Integration.
5. Record. Play & Pause.

Janova

Janova is an automated software testing tool that speaks in plain English. Using web based application testing, our software performance testing works behind the scenes through cloud-based system of workers. It is a web-based, software testing tool that turns plain English business rules in to tests themselves. A web application can be created in seconds in English vernacular, eliminating the need to know complicated code and the user acceptance criteria becomes the automated test itself. This compliments the standard software development methodology where business analysts create requirements in English before developing applications and the quality assurance analysts then use those requirements for acceptance testing. Janova stores your tests, executes the tests against your web application, and clearly communicates the results in an easy-to-read report. Since Janova does all of this in the cloud, it's easier, faster, and far less expensive than other testing solutions. Running tests securely in the cloud, Janova removes the need to keep disparate files and software up to date, or spend thousands of dollars to maintain cumbersome software testing equipment.

Selenium

Selenium is one of the popular automated website testing tools available online and is used to test GUI and functionality of the website. Selenium is easy to get started with and any novice tester can handle it with minimum difficulties. Selenium is unarguably the most widely used open source solution to meet the needs of your testing project. Selenium can be successfully used with cloud services to test mobile apps across multiple browsers and platforms. Selenium interacts with web browsers to test actions, inputs, and expected outcomes to improve device and platform coverage. Sauce Labs is one of the good solutions which lets you execute tests recorded in Selenium on a cloud-based emulation system across multiple devices, browsers, and platform configurations. It is important to develop mobile application tests with extensive coverage—and minimal test script maintenance which is what cloud ensures. Sauce labs can be optimized for testing in a continuous integration workflow with a focus on reliability and scale. By running tests concurrently on Cloud, you can keep your build quick without sacrificing coverage. Selenium comes as a proper fit for cloud based testing because of its open source nature. You can either setup your own selenium grid in cloud or use something like sauce which is an off the shelf solution. Selenium can be easily setup on Amazon EC2. You need to create Amazon AWS and EC2 accounts and create web service access keys and setup environment and install Selenium Grid. Following are the few things to be kept in mind while you decide to go with Selenium:-

- Clear understanding of Vision and Scope of Testing.
- Select An Appropriate Programming Language for Test Automation.
- Pick an Appropriate Text Editor.
- Features of a Good Acceptance Test.
- Make it Robot and Human Friendly.
- Scaling.

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 3, Issue 8, August 2016

Nessus

Nessus is a remote security scanning **tool**, which scans a computer and raises an alert if it discovers any vulnerabilities that malicious hackers could use to gain access to any computer you have connected to a network. This free tool offers a surprisingly robust feature set and is widely supported by the information security community. It doesn't take long between the discovery of a new vulnerability and the posting of an updated script for Nessus to detect it. In fact, Nessus takes advantage of the Common Vulnerabilities and Exposures (CVE) architecture that facilitates easy cross-linking between compliant security tools.

Features of Nessus:-

1. Flexible Deployment Options.
2. Multiple Assessment Types.
3. Rich Assessment Capabilities.
4. Ongoing Management.
5. Connect With Core Systems.
6. Reporting.

VII. CONCLUSION

Every software development group tests its products, yet delivered software always has defects. Test engineers strive to catch them before the product is released but they always creep in and they often reappear, even with the best manual testing process. Cloud-based automated testing is the best way to increase the efficiency and coverage of your software testing. Cloud-based automated testing tools provides the way to playback pre-recorded and predefined actions compare the results to the expected behavior and report the success and failure of these tests to the engineers. Cloud-based automated testing offers a best way to test cross platforms and goes without slowing time-to-market or breaking the bank. Cloud-based automated test will ensure the capacity of the software even in the most extreme scenarios. Cloud-based automated testing eliminates the effort and cost related to extending the on-premise test infrastructure. The Cloud-based automated testing provides the assured performance, worldwide readiness, cost control, and enterprise application coverage benefits to their customers.

REFERENCES

1. Andrea, J. (2007). 'Envisioning the Next Generation of Functional Testing Tools'. IEEE Software, 24(3),58-65.
2. Bezeir, B. (1990). Software Testing Techniques, 2nd edition. New York: Van Nostrand Rheinhold.
3. www.janova.us
4. www.tenable.com/products/nessus-vulnerability-scanner.
5. www.gallop.net/blog/succeeding-with-selenium-testing.
6. www.ranorex.com.
7. Pressman, R.S. & Associates, Inc. (2005). Software Engineering: A Practitioner's Approach, 6/e; Chapter 14: Software Testing Techniques.
8. IEEE(1990), IEEE Standard Glossary of Software Engineering Terminology, Los Alamitos, CA: IEEE Computer Society Press.
9. Chauhan, R.S. , Singh, I. (2014). 'Latest Research and Development on Software Testing Techniques and Tools'. INPRESSCO,4(4),2368-2372.