International Journal of Computer Applications Technology and Research
Volume 2– Issue 4, 443 - 448, 2013, ISSN: 2319–8656

Priority Based Hybrid Automation Testing Tool

Priyanka
Thapar University
Patiala, India

Sunita Garhwal
Thapar University
Patiala, India

Abstract: Software testing is an important means to ensure software quality and to improve software reliability. As the size and complexity of software applications is continually growing, manual testing becomes infeasible in such arena. Automation of the software testing process saves time and provides better utilization of resources and thus, plays a significant role in the testing activity.

This paper presents a hybrid testing tool which extends the support to automatic testing by providing functionality of both GUI and Code based testing. Moreover, it also prioritizes modules under test to achieve a certain degrees of confidence in the testing of web applications that are under tight project deadlines. The proposed hybrid approach allows to combine the benefits of both approaches while keeping a simple interface and treating the two types of tests in a unified fashion.

Keywords: Testing, Automation, Priority, Hybrid Testing, Code Parser

1. INTRODUCTION

Software organizations are competitive in all the aspects, especially in terms of the quality of the products to be delivered and the context of the time line. With the increasing demands for faster time-to-market and the continuous growth of higher software quality, the software market is becoming more challenging. These challenges are embedded in all areas of software engineering, including verification and validation. Also, increase in the software complexity raises the probability of more defects.

Software tests have to be repeated often during development cycles to ensure quality. Every release of the software may be tested on all supported operating systems and hardware configurations. Manual testing is costly and time consuming whereas, automating the testing process provides the solution for improving the effectiveness of software testing [1]. Thus, automation testing plays an important role in ensuring testing software quality and reliability.

Automation testing is an important aspect in software industry. In order to develop an effective automation tool, a priority based hybrid automation tool has been designed. The existing GUI automation tool is enhanced by adding the functionality of code based testing which is further improved by adding a layer of priority to the automation testing [2]. The tool is used to create completely automated tests for the validation of the functionality and behaviour of the application.

1.1 GUI Testing

GUI testing is a process to test application's user interface to detect whether application is functionally correct or not. GUI tests are performed from the view of the end users of the application and thus, GUI testing is vital for quality assurance[3]. It involves carrying set of tasks and comparing the respective outcomes with the expected output and ability to repeat same set of tasks multiple times with different data inputs and same level of accuracy. GUI testing is used to determine whether the various GUI components

react in the desired manner or not. Implementing GUI testing improves quality, speeds up development process and reduces risks towards the end of the software development life cycle[4]. It can be performed manually with the help of human tester as well as automatically with the help of a software program.

1.2 Manual vs. Automatic Testing

Manual and automation are two different approaches of testing with same goal. Manual testing is conducted by writing test cases and then manually executing them[1]. Automation Testing makes use of software to control the execution of tests and the comparison of actual outcomes with predicted outcomes. Manual testing has advantage of human intuition and common sense which can't be replaced by automation tools but on the other end, it is time consuming, cumbersome and repetitive process and thus, often fails to provide required coverage for the testing process which requires multiple execution cycles. In contrast to manual testing, automated testing requires very less effort on developer's side. It not only automates test case execution, but also test case generation and test result verification. When using automated testing systems, users have to specify what to test and not how. In particular this means that the users have to automatically select relevant manual test cases with regards to the current testing goal [4]. A fully automated testing system is able to test the entire software without any user intervention. Manual testing of software is tedious task and thus, there is a great need for reducing the high costs by means of automated testing [5].

2. PROPOSED TOOL

The proposed tool is a combination of following:

i) Manual testing

ii) Automation testing

iii) Code driven testing

iv) Priority based testing

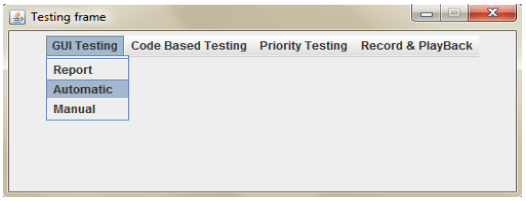

Figure 1. Testing Tool Window

2.1 Manual Testing

It is the process of manually testing the software for defects/bug. Manual testing requires a tester to perform the manual test operations without the help of any automation software.

A manual tester performs the following steps:

i) Select the module to be tested

ii) Enter a test case manually

iii) Execute the test.

iv) Verify the actual result with expected result.

v) Record the results as pass or fail.

vi) Make a summary report of pass and fail test cases.

vii) Publish the report.

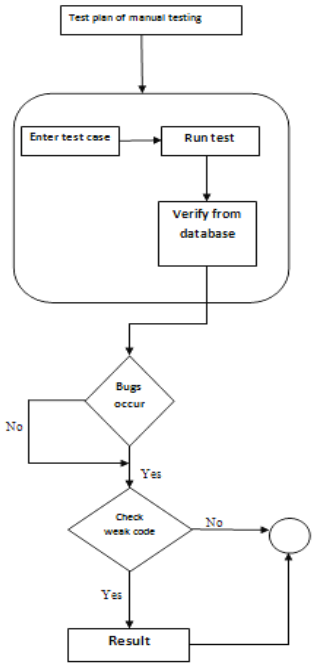

Figure 2. Flow Chart of Manual Testing

2.2 Automation Testing

Automation testing means using automation tool to execute the test case suites. It involves the use of software to control the execution of the test.

The tester would perform the following steps for automation testing.

i) Select the module to be tested.

ii) Browse the test case suite file.

iii) Start the testing process.

The automation tool performs the following:

i) Enter the test data into the system under test.

ii) Compare the actual outcomes with predicted outcomes

iii) Record the test case results as pass or fail

iv) Generate the detailed test report

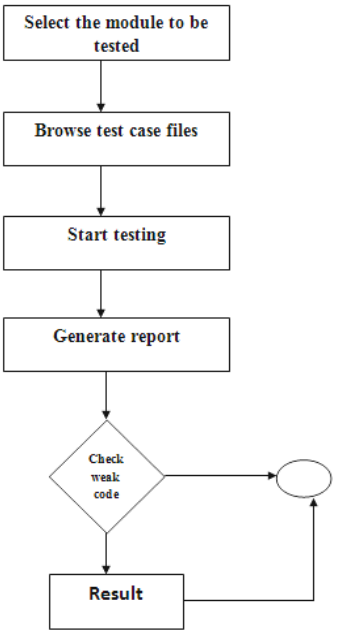

Figure 3. Flow Chart of Automation Testing

2.3 Code Based Testing

The principal of code based testing is to have each and every event/action in the program executed at least once during the test. On the basis of the code based testing, a tester attempts to determine all the reachable elements in the software under the test. The testing process begins by first creating a xmi file for the software under test followed by parsing the xmi to examine the weak code i.e. the code which is not being used during the current execution of the software.

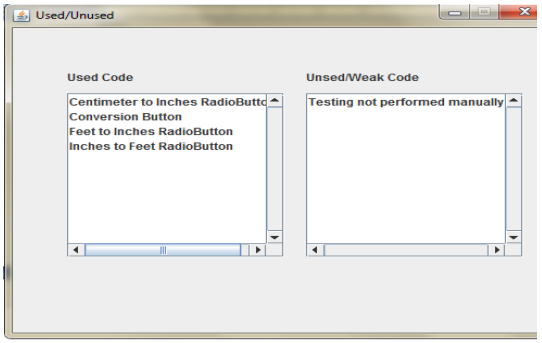

Figure 4. Weak Code Window for Priority Testing

2.4 Priority Based Testing

In this phase of testing, tester defines priority for various modules of application under test. The priorities will be used to execute tests over the specified modules in a specific order. High priority modules will be tested prior to the low priority modules.

Once the testing process for a particular module is over, the test report is generated and the tester is notified accordingly. The testing process continues till the test report for each module under test is generated successfully.

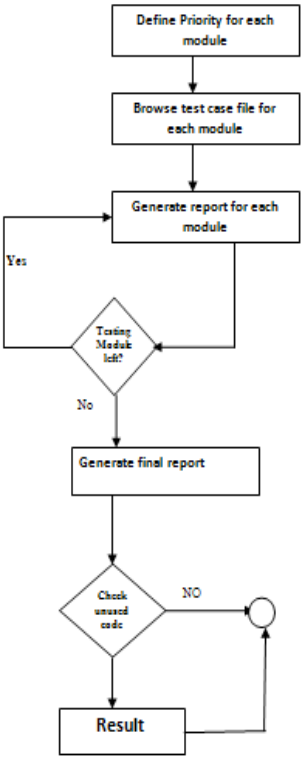

Figure 5. Flow Chart of Priority based Testing

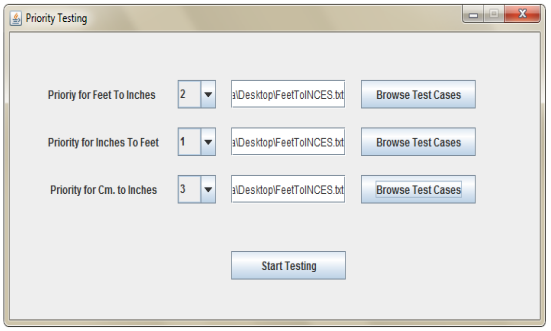

Figure 6. Input Frame for Priority based Testing

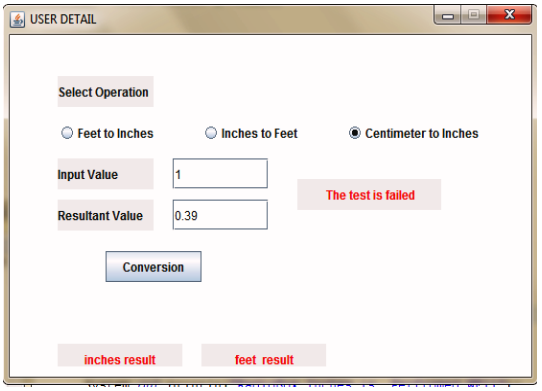

Figure 7. Notifications on Testing Window

2.5 Responsibilities of Proposed Tool

The proposed tool has the following capabilities:

i) *Efficiency in recording*: It can capture all the events and actions that were performed on the application during the automation and manual testing which ever being performed at that particular instant of time.

ii) *Reporting of test results*: The test results can be reported passed/failed statement at the time of manual, automation and priority testing.

iii) *Parsing the code*: It can parse i.e. analyze the source code of any program written in java to examine the various fields, methods, constructors and the interfaces of the java program being examined.

iv) *On camera testing*: The user has a privilege to have the snapshots of the various events involved during the whole testing process.

v) *Prioritizing the testing of modules*: The user can prioritize various modules under test to define the flow of testing in the required manner.

vi) Finding the weak code: The tool can help the user to determine the weak code i.e. code not being used during the testing of the software under observation.

2.6 Comparative Analysis of Testing Types

The following graph shows the efficiency of various types of testing in terms of time. From the results, it can be concluded that priority based testing is more efficient in terms of time in comparison to automatic and manual testing.

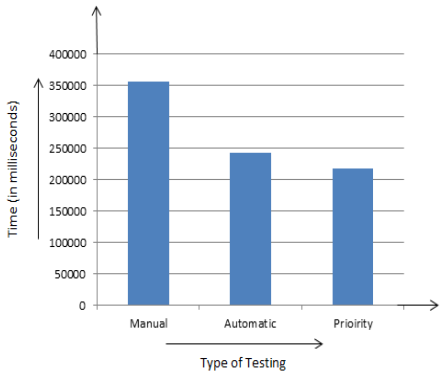

Figure 8: Testing Vs. Time

3. CONCLUSION

GUI testing is vital for quality assurance because the GUI tests are performed from the view of the end users of the application. Many times all the functionality of the application can be invoked through the GUI and therefore GUI tests can cover the entire application.

As the size and complexity of software is continually growing, manual testing becomes very difficult and tedious task. But on the contrary, automated testing can help to improve efficiency of the testing process in order to identify areas of a program that are prone to failure.

The proposed model allows the user to define priority for each of the modules under test. After each of the modules is tested effectively, the corresponding testing report is generated automatically and is available for the review while other modules are being tested. The priority based automation tool helps to minimize the user intervention to a large extent, which makes it highly efficient and less time consuming.

The code based testing helps the user to identify the structural components which are not being used in that particular phase of the testing process, hence enables user to debug the entire application efficiently. The capability of record and playback and on-camera testing further improves the efficiency of tool.

Hence, priority based hybrid automation tool proves to be highly efficient in terms of quality and time.

4. FUTURE SCOPE

Various future opportunities that can be explored are as follows:

i) The code parsing feature of the proposed tool is able to parse only the java source code files which can be extended by making the it language independent.
ii) The design of tool can be generalized to test any GUI java application.
iii) More features of the code based testing can be explored so that the tool can serve as a total solution for testing.

5. REFRENCES

[1] A. Leitner, I. Ciupa and B. Meyer, "Reconciling Manual and Automated Testing: the AutoTest Experience", Proc. 40th Hawaii International Conference on System Sciences, pp: 261-271, January 2007.

[2] M. Ali, S. Mushtaq and N. Arshad., "Priority-Based Automated Testing of Web Application", Proc. IEEE Conference on Communication and Information Technology, 2012.

[3] P .Nagarani and R. VenkataRamanaChary "A Tool Based Approach for Automation of GUI Applications", Proc. 3rd IEEE International Conference of Computing Communication &Networking Technologies, pp: 1-6, July26-28, 2012.

[4] P. Aho, N. Menz and T. Räty "Enhancing Generated Java GUI Models with Valid Test Data", Proc. IEEE Conference on Open System, September 2011.

[5] J.Prabhu and N.Malmurugan "A Model for GUI Automated Testing Framework in Software System", *International Journal of Computer Applications*, vol. 64, no.15, February 2013.