An empirical assessment on cost-effectiveness and re-usability of test cases based on defect severity levels in regrerssion testing

Lalitha Venkatesan.A

Abstract— Regression testing is an important part of the software development life cycle. In this paper an empirical assessment is made between defect priority and defect severity levels. This is assessment is done because, when priorities are assigned to the defects it can dynamically change, but defect severity is absolute and does not change as they reflect the state and quality of the product. Testing the software based on defect severity levels requires a defect bash. When this process is carried out at a given fixed time maximum defects or faults can be revealed. In this approach defect severity levels are assigned using a metric called Total Severity of Faults Detected (TSFD) .In Test Case Prioritization technique the test cases are discarded, those discarded test cases are repaired using the test case repairer and then stored into Test Case Bin(TCB).This TCB acts as a repository to store those repaired test cases. The reason is, instead of writing new test cases, the repaired test cases in TCB are used as well. This TCB not only achieves reusability of test cases but it is highly cost-effective too.

Index Terms— Empirical Assessment, Defect Bash, TSFD (Total Severity of Faults Detected), TCB (Test Case Bin), Reusability and cost-effective.

1 INTRODUCTION

As software ages, the cost of maintaining the software dominates the overall cost of developing the software. Testing can be used to build confidence in the correctness of the software and to increase the software reliability [1]. As engineers maintain software systems, they periodically regression test them to detect whether new faults have been introduced into previously tested code and whether newly added code functions according to specifications. To assist with regression testing, engineers may prioritize their test cases so that those that are more important are run earlier in the regression testing process [2].The failure of regression testing can be found very late in the cycle or found by the customers. Regression testing is important in today's context since software is being released very often to keep up with the competition and increasing customer awareness [3]. Regression testing is necessary when a program has been changed and is usually done during maintenance. If adaptive or perfective maintenance were performed then program specifications were modified. If corrective maintenance was performed then the specifications may not been modified [1].Leung and White in [4] describes two types of regression testing based on whether specifications were changed. The fig.1 shows the Leung and White Cost – Effectiveness Curve determining the cost-effectiveness and cost-ineffectiveness of the software based on regression testing. Corrective regression testing applies when specifications are unmodified and test cases can be reused [4].This sort of testing can be followed to increase the reusability of test case.

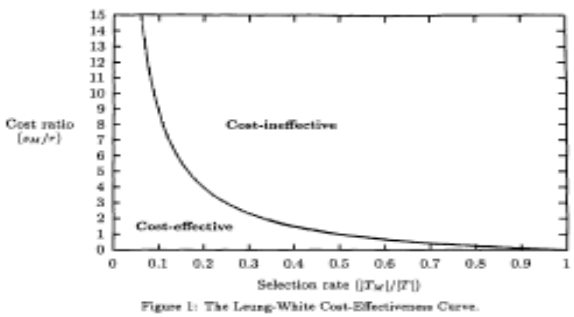

Figure 1: The Leung-White Cost-Effectiveness Curve.

Progressive regression testing applies when specifications are modified and new test cases must be designed [4].This paper

• *Author is currently working as Associate Professor in the department of CSE at Geethanjali College of Engineering and Technology, Hyderabad, India. E-mail:a.lalitha83@gmail.com*

is organized as follows, section II presents the background and related work, section III presents experiment details, section IV presents experiment materials, section V presents experimental design, section VI presents conclusion and future enhancements and finally the references.

2 BACK GROUND AND RELATED WORK

Test case prioritization techniques ([4], [5]), test cases are scheduled based on the priority, according to some criterion (Code Coverage) are executes earlier in the regression testing process. In prioritization techniques, unlike many other techniques (Retest-All, Regression Test Selection, and Test Suite Reduction) this technique discards test cases. Those discarded test cases are repaired and then stored into Test Case Bin (TCB).

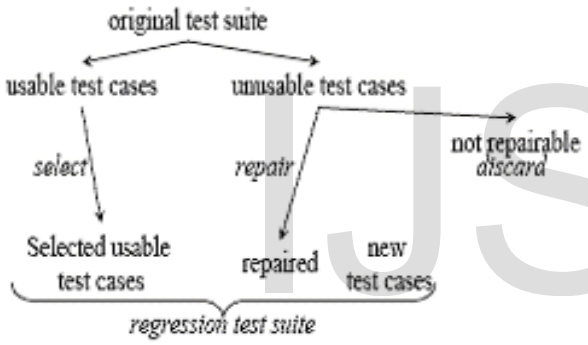

Figure 1.Types of test cases

In this paper, an empirical assessment is made between defect severity and defect priority levels. The severity of defects provides the defect bash team (also called as test team) a perspective of the impact of that defect in product functionality. The table 2.1 shows the defect definition and defect classification which helps in assigning the defect severity levels [3].

Table 1 Common Defect classification and Defect Definition

Defect Classification	Defect Definition
Extreme	Product crashes
Critical	Basic functionality is not working
Important	Extended functionality is not working
Minor	Product behaves differently
Cosmetic	Minor irritant

Defect bash is an adhoc testing where people performing different roles in an organization test the product together at the same time [3] with intent of finding maximum faults or errors. When this process is carried out at a given fixed duration maximum faults can be revealed thus helps in developing defect-free software.

3 EXPERIMENT DETAILS

Reasons behind Test Case Bin:

There are some specified reasons to use Test Case Bin (TCB) they are:

There are different types of regression testing techniques namely Retest-All[1], Regression Test Selection[2], Test Suite Reduction[3], then all other techniques the one Test Case Prioritization has an ability to discard the test cases. Every time when the test cases are prioritized it is not meant that they are done exactly as it depends on the different criterion.

1. When P is modified, creating P', test engineers may simply reuse all non-obsolete test cases in T to test P'.

2. Regression test selection use information about P, P' and T to select a subset of T with which to test P.

3. As P evolves, new test case may be added to T to validate new functionality. Over time T grows, and its test cases become redundant.

To improve user-perceived software quality in a cost effective way by considering potential defect severity levels.

1. To increase the rate of detection of severe faults during system levels testing of new code and regression testing of existing code.

2. Failure of regression can be found very late in the cycle or found by customers having a well-defined methodology (Test Case Bin) can prevent such costly issues.

• *Author is currently working as Associate Professor in the department of CSE at Geethanjali College of Engineering and Technology, Hyderabad, India. E-mail:a.lalitha83@gmail.com*

3.For the above stated reason, the tester's can develop their own TCB's to store the discarded test case which does not require any test cases to be written very formally.

Structure of Test Case Bin (TCB):

Test Case Bin acts as a repository where the repaired test cases are stored for later use. Fig.3 mentions the various use cases that are involved in processing the Test Case Bin (TCB).The test cases from the original test suite is classified into usable and unusable test cases, those unusable test cases are repaired using the test case repairer and then stored into TCB.

Priority and Defect Severity Levels

The main criteria which very often used are code coverage criterion and it also depends on the perception of the tester. Any criteria can be used depending on the tester's choice ([2], [8], and [9]). During this case, there is a change in priorities assignment. In order, to overcome this issue defect severity levels are assigned which are absolute and as they reflect the quality of the product [3].This TCB makes use of defect bash as well because it does not require for us to write any test cases as it depends on tester's choice and creativity. Test Suite contains various test cases that are formed using Test Case Prioritization technique. When this technique is performed some test cases are discarded, these test cases are stored into a repository like structure called as Test Case Bin (TCB). This TCB is used to achieve both re usability of test cases and also helps in saving the testing cost and time.

Defect Bash:

Defect bashes are planned, short-term and resource intensive activities. Testing by all the participants during defect bash is not based on written test cases; whatever is tested by defect bash is left to an individual's decision and creativity. Even though it is said that defect bash is ad-hoc testing, not all activities of defect bash are unplanned. All the activities in the defect bash are planned activities, except for what to be tested [3].There is several essential practices that are followed in defect bash, one of them is "Cross boundaries and test beyond assigned areas. This practice means that, not only functional defects are found but also certain other non-functional defects such as memory leak, long turnaround time, missed requests, high impact and utilization of system's resources. As defect bash is a unique testing method which can bring out both functional defects and non-functional defects [3]. However, if the defect bash lab is not set up or not monitored properly, there is a chance that some of the non-functional defects may not get noticed at all.

4 EXPERIMENT MATERIALS

Defect Bash Team:

Whenever software is created, right from the requirement phase till the maintenance phase the software keeps on evolving or undergoes changes or enhancements. These enhancements are necessary to increase the quality of the software. As this defect bash is not based on written test cases it can very effectively make use of TCB. During this assessment, severity levels are assigned not only for functional defects but also for the non-functional defects to keep track all types of defects. Finally, the defect bash team has to generate test report once the defects are properly classified.

Additional Instrumentation:

Based on the test report by defect bash the values of defect severities are plotted using the tool called PRISM. This tool namely the "PRISM" is a probabilistic model checker used for formal modelling, it analysis the systems which exhibit random or probabilistic behaviour. PRISM includes a simulator; a tool which can be used to generate sample paths through a PRISM model. It is useful for debugging models during development and for running sanity checks (smoke test) on completed models. There are three models that are used in this tool are DTMC[4], CTMC[5], MDP[6].Using this tool the present state of the various test cases are studied well and their respective behaviours are plotted. Using this tool the Test Case Bin (TCB) is modelled.

• *Author is currently working as Associate Professor in the department of CSE at Geethanjali College of Engineering and Technology, Hyderabad, India. E-mail:a.lalitha83@gmail.com*

[4] Discrete Time Markov Chains [5] Continuous Time Markov Chains [6] Markov Decision Processes [77]TSFD (Total Severity of Faults Detected) is the summation of severity values of all faults identified for a

release.

5 EXPERIMENTAL DESIGNS

This experimental design depends on the variables these variables can be of two types they are dependent variables and independent variables.

Dependent Variables:

Time allotted for conducting defect bash:

Defect bash is very popular among application development companies, to increase the rate of fault detection and correction. Defect bash is an activity involving a large amount of effort so it is very important to choose the frequency and duration of defect bash well in advance depending on which the success rate of the project is decided.

Costs in fault-detection effectiveness:

Costs in fault detection effectiveness can be measured by assigning proper defect severity levels depending on table 1.After assigning those levels we check for re-usability of test cases. Then the test case repairer is used to make the unusable test cases into usable one.

Savings in test execution time:

As defect bash purely depends on tester view and creativity it is not required to write test cases [3]. Because of this reason, not much cost is involved in making the test cases as well. As it uses the technique Test Case Bin it can very effectively reduce the cost of execution time and validation time.

Independent Variables:

Test Case Bin Technique:

Processing techniques of TCB are stated as follows:

(1). Test cases that are thrown by test case prioritization techniques are stored into TCB initially.

(2).Then those discarded test cases are checked using the "Test Case Checker" to differentiate the usable and unusable test cases.

(3).After classification, the unusable test cases are sent to the "Test Case Repairer" to make the unusable test case into usable one.

(4).Those reusable test cases are stored into Test Case Bin (TCB).

(5).Whenever a test case is retrieved from TCB it is assigned with proper defect severity levels using TSFD7 metric.

(6) Assign Hypothesis levels for the test cases based on certain criteria in order to calculate the reusability levels.

(7).Comparative analysis is made between original test suite and Test Case Bin (TCB) in terms of time, cost and reusability levels.

VI.CONCULSION AND FUTURE ENHANCEMENTS

In this approach the unusable test cases are made into usable test cases using the technique of Test Case Bin (TCB).The above process is done using the test case repairer and defect bash along with the assignment of defect severity levels. Using this technique Test Case Bin (TCB) the test cases are repaired and made use. Re-usability levels of original test suite and Test Case Bin are compared to make this process very effective. This TCB can be more efficient if every organization develops their own TCB's to store the unusable test cases. This TCB not only achieves re-usability of test cases but it is highly cost-effective too thus by reducing the time of writing test cases. Future work is carried in achieving reusability of test cases for the distributed environment as this work is carried for a standalone application.

REFERENCES

[1] Yuejian Li, Nancy J. Wahl, An Overview of Regression Testing, ACM SIGSOFT, Software Engineering Notes Vol. 24, No.1, page 69, Jan 1999.

[2] Hyunsook Do, Gregg Rothermel, On the Use of Mutation Faults in Empirical Assessments of Test Case Prioritization Techniques, IEEE Transactions on Software Engineering, Vol. 2,No.9,September 2006.

[3] Ramesh Desikan, Gopalaswamy Ramesh, Software Testing Principles and Practices, Pearson Education, ISBN 81-7758-121-X, 2006.

[4] Leung, H.K.N. and L. White, A Study of Integration Testing and Software Regression at the Integration Level, Proc. Conf. Software Maintenance, San Diego,pp.290-1.

[5] S.Elbaum, A.G.Malishevsky S. Elbaum, A. Malishevsky, and G. Rothermel. Test case prioritization: a family of empirical studies. IEEE Transactions on Software Engineering, 28(2):159–182, 2002.

- *Author is currently working as Associate Professor in the department of CSE at Geethanjali College of Engineering and Technology, Hyderabad, India. E-mail:a.lalitha83@gmail.com*

[6] T. Hérault, R. Lassaigne, F. Magniette, andS. Peyronnet. Approximate probabilistic model checking. In Proc. 5th International Conference on Verification, Model Checking and Abstract Interpretation (VMCAI'04), volume 2937 of LNCS. Springer, 2004.

[7]http://www.cs.bham.ac.uk/~d xp/prism link to use PRISM tool.

[8] G .Rothermel, R. Untch, C. Chu, and M.J. Harrold,"Prioritizing Test Cases for regression Testing",IEEE Trans. Software Eng., vol.27,no.10,pp.929-948,Oct.2001.

[9] G.Rothermel, S. Elbaum, "On Test Suite Composition and Cost-Effective Regression Testing", vol.13, no.3, pp.227-331 July 2004.

- *Author is currently working as Associate Professor in the department of CSE at Geethanjali College of Engineering and Technology, Hyderabad, India. E-mail:a.lalitha83@gmail.com*

• *Author is currently working as Associate Professor in the department of CSE at Geethanjali College of Engineering and Technology, Hyderabad, India. E-mail:a.lalitha83@gmail.com*