Requirements-Based Test Case Prioritization

Hema Srikanth
Department of Computer Science,
North Carolina State University,
NC 27695
hlsrikan@ncsu.edu

Laurie Williams
Department of Computer Science,
North Carolina State University,
NC 27695
lawilli3@ncsu.edu

ABSTRACT
Test case prioritization techniques have been shown to improve regression-testing activities by increasing the rate of fault detection, thus allowing testers to fix faults earlier. The current techniques, mostly code coverage-based, treat all faults equally. We build upon this work to develop a prioritization scheme with two main goals: identifying the severe faults earlier and minimizing the cost of test case prioritization.

Keywords: Software Testing, Reliability, and Quality

1. INTRODUCTION
Software testing is a strenuous and expensive process. Research has shown that at least 50% of the total software cost is comprised of testing activities [7]. Companies are often faced with lack of resources, which limits their ability to effectively complete testing efforts. Test case prioritization (TCP) that involves execution of test cases in a pre-defined order can be cost effective [2, 3, 6]. As a result, the rate of fault detection early in the development cycle is improved, allowing the developers to rectify the faults earlier while improving the system delivery speed [3, 6]. We build upon previous work and apply TCP at system level.

2. PROBLEM STATEMENT
The current prioritization techniques [3, 6], mostly coverage-based (statement, branch or other coverage), can be expensive to employ and assume that all faults are equally severe in nature. In reality, some faults can be critical in nature (high risk failures), such as causing the customer significant monetary loss, delaying a product shipment, or having safety or security implications. Current TCP schemes can be enhanced by incorporating additional knowledge gained through requirements engineering research: (a) requirements with high complexity tend to have a higher number of faults [1], (b) requirements volatility, which results in re-design, addition or deletion of existing requirements, tend to increase project risk [5], and fault density [4], thus often times causing project failures [5], (c) roughly 20% of the system is responsible for about 80% of the faults [8].
Therefore, there is a need to maximize the effectiveness of the testing resources via a scheme which: (1) is economical (does not add significant overhead to team); (2) improves perceived software quality; and (3) identifies the more severe faults earlier.

3. METHODOLOGY
We build upon the current TCP techniques [3, 6] and propose a multi-faceted prioritization strategy called Prioritization of Requirements for Testing (PORT Version 1.1) by exploring three prioritization factors (PFs): (1) customer-assigned priority on requirements, (2) requirement complexity, (3) requirements volatility. Our preliminary set of research goals is listed below.
G1: To identify the most severe faults/failures earlier in system test.
G2: To improve the software field quality.
G3: To devise the minimal set of PORT PFs that can be used to effectively for TCP.
The PFs, customer-assigned priority (CP), requirements complexity (RC) and requirements volatility (RV) are assigned values. CP is the value (1 to 10) assigned by the customer based on the importance of the requirement. RC is the value (1 to 10) assigned by the developer based on the perceived implementation difficulties of the requirement. RV is the number of times a requirement has changed. Higher factor values indicate a need for prioritization of test case related to that requirement.
Based on the project and customer needs, the development team assigns weight to the PFs such that the assigned total weight (1.0) is divided amongst the PFs. For every requirement, Equation 1 is used to calculate a weighted prioritization (WP) factor that measures the importance of testing a requirement earlier. Test cases are then ordered such that the test cases for requirements with high WP are executed before others.

$$WP = \sum_{PF=1}^{n} (PFvalue * PFweight) \quad (1)$$

4. CASE STUDY

A postmortem analysis was conducted on an industrial project that comprised of 152 KLOC. The project planning started in February 2003, progressed into beta test in August, and delivered in October 2003. A total of 1030 faults were identified in system test: 16 of them were classified as Severity 1, 259 as Severity 2, 608 Severity 3, and 157 Severity 4 faults. We analyzed the fault detection rates of the Severity 1 and 2 faults. Our goal was to determine the characteristics of the severe faults and their actual detection rates.

Our results show that the (1) 13% of Severity 1 and 23% of the Severity 2 faults were found during beta testing, and (2) 78% of the Severity 1 and 2 faults were found in Modules 1 and 2. The test team indicated that Modules 1 and 2 were the two most volatile and complex modules. Of the total number of Severity 1 and 2 faults detected, 12% of the Module 1 and 21% of Module 2 faults were found in beta test. The company's goal for future releases is to reduce the detection of severe faults during beta testing to less than 5%. Identification of severe faults later in the process cycle could result in a lack of time to thoroughly fix severe bugs, thus compromising software quality. These results motivate the development of the PORT TCP scheme.

5. POSTER LAYOUT

Our poster presentation would comprise of (1) Introduction, and (2) Problem Statement, and (3) Example demonstrating PORT, and (4) Graphical Representation of PORT Benefits, and (5) PORT Architecture, and (6) Summary, and (7) Future Work. Table 1 shows the poster layout.

Table 1: Poster Layout of PORT

<table>
<tr><th colspan="3">Prioritization of Requirements for Testing (PORT)</th></tr>
<tr><td>Introduction</td><td rowspan="3">Graphical Representation of PORT Benefits</td><td>PORT Architecture</td></tr>
<tr><td>Problem Statement</td><td>Summary</td></tr>
<tr><td>PORT Example</td><td>Future Work</td></tr>
<tr><td colspan="3">Collaborators and Sponsors</td></tr>
</table>

6. SUMMARY

Our preliminary results show that requirements complexity and volatility impact fault density. These factors were identified as initial set of PFs for PORT. We plan to further analyze the industrial data to determine the effectiveness of customer priority in our prioritization scheme. We believe that PORT could improve the effectiveness of testing activities as it (1) reduces the effort utilized for TCP in comparison to coverage-based techniques that prioritize based on the number of statements or branches covered, and (2) focuses on functionalities that are of highest value to the customer, and (3) improves the rate of detection of severe faults. Rectifying severe faults earlier is believed to improve perceived software quality. We will validate PORT with industrial partners. IBM and Tekelec have expressed interest in participating in this research.

ACKNOWLEDGEMENTS

This work has been funded by the North Carolina State University Center for Advanced Computing and Communication (CACC).

REFERENCES

[1] S. Amland, Risk Based Testing and Metrics, 5th Intl. Conf. EuroSTAR '99, Barcelona, Spain, 1999, pp. 1-20.

[2] F. Basanieri, A. Betolino, and E. Marchetti. CoWTeSt: A Cost Weighed Test Strategy, Escom-Scope 2001, London.

[3] S. Elbaum, A. Malishevsky, and G. Rothermel, "Test Case Prioritization: A Family of Empirical Studies," IEEE Trans. on Software Engineering, vol. 28, February 2002.

[4] Y. K. Malaiya and J. Denton, "Requirements volatility and defect density," Proc. of 10th Intl' Symposium on Software Reliability Engineering, 1999.

[5] J. O'Neal and D. Carver, "Analyzing the Impact of Changing Requirements," Proc. IEEE Int'l Conference on Software Maintenance, Florence, Italy, 2001.

[6] G. Rothermel, R. Untch, C. Chu, and M. Harrold, "Test Case Prioritization," IEEE Transactions on Software Engineering, vol. 27, pp. 929-948, October 2001.

[7] L. Tahat, B. Vaysburg, B. Korel, and A. Bader, "Requirement-Based Automated Black-Box Test Generation," 25th Annual Int'l Computer Software and Applications Conference, Chicago, Illinois, 2001.

[8] E. Wong, J. Horgan, M. Syring, W. Zage, and D. Zage, Applying design metrics to predict fault-proneness: a case study on a large-scale software system, Software Practice and Experience, 2000, 30, pp 1587-1608.

The primary author, Hema Srikanth, of this paper is a Ph. D candidate at North Carolina State University, Raleigh, NC. She is in the second year of her Ph. D program and is a new investigator.