MIS Review Vol. 18, No. 2, March (2013), pp. 19-50

Optimal Software Release Policy Approach Using Test Point Analysis and Module Prioritization

Praveen Ranjan Srivastava[1], Subrahmanyan Sankaran[2], Pushkar Pandey[2]
[1]*Information Technology and System Group, Indian Institute of Management, Rohtak*
[2]*Department of Computer Science & Information Systems, Birla Institute of Technology and Science*

ABSTRACT: *When to stop testing and release the developed software is the one of the most important questions faced by the software industry today. Software testing is a crucial part of the Software Development Life Cycle. The number of faults found and fixed during the testing phase can considerably improve the quality of a software product, thereby increasing its probability of success in the market. Deciding the time of allocation for testing phase is an important activity of quality assurance. Extending or reducing this testing time, depending on the errors uncovered in the software components, can profoundly affect the overall project success. Since testing software incurs considerable project cost, over-testing the project can lead to higher expenditure, while inadequate testing can leave major bugs undetected, thereby risking the project quality. Hence prioritizing the components for testing is essential to achieve the optimal testing performance in the allotted test time. This paper presents, a Test Point Analysis based Module Priority approach to determine the optimal time to stop testing and release the software.*

KEYWORDS: *Module Priority, Expendability, Test Point Analysis (TPA), Normalization, Synchronization Factor, Maintenance Factor, Node Criticality, Reusability, Function Points, Technical Complexity.*

1. Introduction

Software testing is an expensive process. Many software organizations rely on the expertise of technical managers to decide the time to be allocated to test the software. The release time is thus mostly determined by prior experience. Research has shown that, as much as 45% of the total software development cost is spent in testing. In many projects, the time used in debugging can be around 50% of the total development effort (Myers, 1976). Releasing an under-tested software may lead to the risk of latent bugs in the software, leaving the customer dissatisfied and also incurring higher cost of removing the faults post-release. While testing the software beyond a certain limit may lead to an over-priced testing effort, resulting in loss of revenue. Also, a late release of the product in business may result in losing the market foothold. This tradeoff needs to be balanced by finding the optimal time of release and justifying the cost with the stop-test decision.