Task-Resource Capability Alignment:
Discerning Staffing and Service Issues in Software Maintenance

Rafay Ishfaq, Department of Aviation and Supply Chain Management, Auburn University, Auburn, AL, USA

Uzma Raja, Department of Information Systems, Statistics, and Management Science, University of Alabama, Tuscaloosa, AL, USA

ABSTRACT

The effective management of software maintenance processes involves decisions about workforce levels, skill and expertise mix of developers, assignment of defect resolution tasks, and monitoring key system performance measures. This research uses a queuing based simulation approach to study these managerial issues. Using the data archives of a large global software organization, an empirical study of the historical defect reports and management decisions is conducted. A task-resource capability alignment scheme is developed that captures the defect complexity and skill/experience capabilities of software maintainers. The results of the empirical-computational study show that the defect arrival/reporting process affects the resource utilization and the time a defect spends in the system. The results also highlight the role of dedicated and shared resources on the system performance and indicate that replacing an experienced and skilled developer requires a significant order of magnitude increase in the maintenance workforce.

Keywords: Defect Resolution, Resource Allocation, Simulation, Software Maintenance, Task Assignment

INTRODUCTION

In any software system, maintenance is an inevitable process (Lehman & Belady, 1985). Software maintenance refers to the process of managing changes in a software product after it is released for use. The maintenance activities are generally classified as: (a) corrective, which relates to repairing a fault, (b) adaptive in terms of handling new requirements, and (c) perfective, which is focused on improving the system performance (Lientz & Swanson, 1980). System users, software project managers and system developers can initiate maintenance requests. Software product managers evaluate these requests from a cost and impact perspective. Once approved, requests for change can result in several activities, e.g., system analysis, re-design, prototyping and validation of the system components. Once all the needed

DOI: 10.4018/irmj.2012100101

changes have been implemented and verified, the new version (containing all the changes) is released to the users (Arther, 1988; Banker & Slaughter, 1997).

Managing the software maintenance process and defect resolution activities in an organization is a challenging management function (Swanson & Beath, 1979). Efficient management of the software maintenance process requires matching the capacity and capability of software developers (referred to as resources) with the defects of varying levels of complexity (Ngo-The & Ruhe, 2009). The defects are reported to a team of software developers, which handles these defects through a software defect resolution (SDR) process. A defect identifier, a defect category, the skill requirements of maintenance resources and a due date for defect resolution, parameterize each software defect. The actual time to resolve a defect depends on the skill and experience level of the assigned developer and the inherent complexity of the defect. The total time a defect spends in the SDR process (referred to as system) consists of the time spent to resolve the defect and the queue wait time, when the assigned team member is busy working on a defect assigned earlier.

There are generally two perspectives regarding the software defect resolution process: the perception of the customer, and the internal perception of the software company (April, Hayes, Abran, & Dumke, 2005). This paper studies the defect resolution process from the latter perspective. In a software company, the activities of a defect resolution process have typical characteristics (Abran & Nguyenkim, 1993). The defects arrive in a random manner and are managed more in line with a queue-management approach rather than as managing a project. The defects are reviewed and managed according to their assigned priorities, which may change at any time.

The SDR considered in this paper is described in Figure 1. The defects of a particular software product are reported to the manager of the software maintenance team. The team manger classifies the defect by assigning it a priority based on the type and its significance. The team manager uses allocation rules to assign a software developer to resolve the defect. This allocation depends on the developer(s) capability, defect resolution requirements, availability of the developer(s) and their assigned workload. Software developers manage their individual work queues according to some sequencing rule. A defect is handled and resolved before the next defect is taken up. The resolved defects exit the system.

The performance of the SDR process is evaluated based on the system level performance targets. The performance measures most often used are; completion time, timeliness and throughput (Leung, 2004). The completion time measure relates to the sequencing and scheduling of activities which will ensure that either a single or a batch of defects is resolved in the shortest possible time, given the limitations of the available resources. The timeliness measure refers to the ability of the SDR process to conform to some required time-line associated with resolving a specific software defect, say by a given deadline for a new product version release or meeting the operational targets such as the average queue waiting time and the average time spent by a defect in the system. The third measure, throughput, relates to the performance of the resources in terms of how many defects are resolved by a resource in a fixed period of time or in terms of the resource utilization, which measures the percentage of available time a resource uses to fix the defects.

The purpose of this research is to evaluate the managerial issues related to the SDR process focused on defect-resource allocation policies, staffing levels and resource skill requirements. The specific objectives of this research are: (a) study the role of the dedicated and shared resources (software developers) in improving the efficiency of the SDR process, and (b) study the marginal impact of the changes in the mix of resources and their skill levels on the system performance. The rest of this paper is organized as follows: First, we discuss the prior research in the area of software maintenance and the defect resolution process. The next section presents