ABSTRACT

It is now widely accepted that software projects utilizing the Web (e-projects) face many of the same problems and risks experienced with more traditional software projects, only to a greater degree. Further, their characteristics of rapid development cycles combined with high frequency of software releases and adaptations make many of the traditional tools and techniques for modeling defects unsuitable. This article proposes a simple model to explain and quantify the interaction between generic defect injection and removal processes in e-projects. The model is based upon long standing and highly regarded work from the field of quantitative ecological population modeling. This basic modeling approach is then subsequently tailored to fit the software production process within an e-project context.

Keywords: e-commerce models; e-projects; e-commerce software; reliability modeling; predator prey systems; Web based applications

INTRODUCTION

The Internet now forms a major platform supporting a wide range of commercial, educational, and social applications. While many of the early developments on the Web were based on client-server technology, the position has changed dramatically in the last few years with new technologies forming the basis of current applications. Today, many of the software industry’s current and new developments are destined for use either directly or indirectly, in sophisticated Web-based applications. Unfortunately, these developments have not come without cost, and industry is now faced with problems associated with verifying and maintaining Web-based applications. Current approaches to verification in the software industry are not succeeding. While systems grow evermore complex, the percentage of total costs consumed by verification continues to grow as defect rates increase (Tassey, 2002).

The characteristics of e-projects differ from traditional information technology projects in a number of ways. E-projects are characterized as any project which evolves software that is deployed via the
E-projects are typically much smaller than many traditional projects; correspondingly, their production period is much shorter with projects lasting from several weeks to a few months not uncommon. They also tend to evolve at a much faster rate (typically the “shelf life” of an e-project system may be only 3 months) and hence their production is nearly always highly iterative in nature. Starting from a largely descriptive overview of the required functionality they often move through a quick build and test cycle, to be immediately faced with perhaps another quick build and test cycle. This situation can become even more extreme when using modern development methodologies; for example, when using the SCRUM methodology (Schwaber, 2004), the project manager often aims for iteration cycles as short as every 2 weeks.

Many traditional production processes, such as the waterfall or spiral models, defer verification and validation activities until late in the life cycle, resulting in these components often being considered as an afterthought to the production component. Further, the testing activities often get disassembled into artificial sub-activities such as unit testing, integration testing and system testing. The popular “V model” of an integrated testing process is a good example of this type of structuring; This model results in the production team being forced into conducting these subactivities, regardless of the relative cost-benefit issues associated with each of these subactivities. Clearly, in any arbitrary project the costs and benefits associated with any verification and validation activity will be highly dependent on the domain of operation and the product under development. Since testing now accounts for more than half of the costs on many projects, any production methodology that fails to actively consider the costs and benefits of the testing activities is potentially wasting an extremely large amount of resources and failing to perfect the product for the marketplace. In contrast, e-projects systems with their short cycles tend to have a simpler, more time focused structure, typically comprising intertwined build and test components. For example, the recent rise of popularity of agile methodologies can be directly linked to the increasing number of e-projects in production. These methodologies often view the production processes in a different light from the traditional production methodologies; for example, in an XP production environment, the use of test-first programming (Beck, 2003) is extremely common.

Unfortunately, these differences in the production process of e-projects cause difficulties for many quantificational models of artifacts within the software processes or products, including approaches to modeling defects. Traditional views of modeling defects tend to assume that we are using explicit quality targets, which is less likely to be the case within these new types of systems where managing the risks to the project and product are a more common approach. In addition, traditional models often assume that existence of a production model such as the “V,” where we have a long production period followed by a significant period of verification and validation. Commonly, we only model and analyze this verification and validation period. Again, this type of model will not translate effectively into the e-project domain. What is required is an alternative type of defect model which attempts to directly describe these frequent cycles of build and test processes. The principal aim
Related Content

Multi-Level Adaptation in End-User Development of 3D Virtual Chemistry Experiments
www.igi-global.com/chapter/multi-level-adaptation-in-end-user-development-of-3d-virtual-chemistry-experiments/137360?camid=4v1a

Integration of Libre Software Applications to Create a Collaborative Work Platform for Researchers at GET
www.igi-global.com/article/integration-libre-software-applications-create/2609?camid=4v1a
Improving Online Readability in a Web 2.0 Context
www.igi-global.com/chapter/improving-online-readability-web-context/37696?camid=4v1a

Autonomic Healing for Service Specific Overlay Networks
www.igi-global.com/article/autonomic-healing-service-specific-overlay/70385?camid=4v1a