Project Cost Overrun Simulation in Software Product Line Development

Makoto Nonaka[1], Liming Zhu[2], Muhammad Ali Babar[3], and Mark Staples[2]

[1] Faculty of Business Administration, Toyo University, Japan
nonaka-m@toyonet.toyo.ac.jp
[2] National ICT Australia
{liming.zhu,mark.staples}@nicta.com.au
[3] Lero, University of Limerick, Ireland
Muhammad.alibabar@ul.ie

Abstract. The cost of a Software Product Line (SPL) development project sometimes exceeds the initially planned cost, because of requirements volatility and poor quality. In this paper, we propose a cost overrun simulation model for time-boxed SPL development. The model is an enhancement of a previous model, specifically now including: consideration of requirements volatility, consideration of unplanned work for defect correction during product projects, and nominal project cost overrun estimation. The model has been validated through stochastic simulations with fictional SPL project data, by comparing generated unplanned work effort to actual change effort, and by sensitivity analysis. The result shows that the proposed model has reasonable validity to estimate nominal project cost overruns and its variability. Analysis indicates that poor management of requirements and quality will almost double estimation error, for the studied simulation settings.

Keywords: process simulation, cost overrun estimation, software product line development.

1 Introduction

Software Product Line (SPL) development can shorten the total cycle time, the duration from the beginning of core asset development to the end of product development, by applying large-scale reuse [1]. However, effort estimation, planning, and development management for SPL are more complex and difficult than those for sequential development, because of inter-connected relationships between core assets and products, concurrency of their projects, and multiple deadline management [2]. In addition, there are still general problems with software effort estimation because of unplanned work [3] and requirements volatility [4]. The total cycle time can sometimes be longer than initially planned because of these problems.

Requirements volatility is the tendency of requirements to change over time. High requirements volatility has a large impact on cost and effort overruns [5]. Some unexpected critical requirements changes are in practice unavoidable, and

J. Münch and P. Abrahamsson (Eds.): PROFES 2007, LNCS 4589, pp. 330–344, 2007.

can for example be caused by faults on external components to be compensated by software, or by marketing issues requiring new functionality to catch up with other competitive products.

Quality problems are also important factors for cost overruns. A certain number of defects will inevitably remain in released software products, as software testing can not demonstrate the absence of defects [6]. When residual defects in core assets are detected after their release to product projects (not to customers), corrective maintenance is usually performed to modify the core assets. When multiple product projects are undertaken simultaneously during core asset maintenance phase, corrective maintenance in core assets sometimes brings associated rework to all ongoing product projects that depend on the core assets, to adapt the products to the changed core assets. We have called this type of rework "adaptive rework" [7].[1] In addition, each product project will also be delayed caused by defects injected during the project.

Though the reasons why software effort estimation error appears have been shared among software professionals [3,9], it is still difficult to predict the amount of overrun and its variability, or the level of risk, in specific situations. The variability of effort estimation gives us more useful information than traditional minimum-maximum intervals to indicate its uncertainty [10]. This can be estimated by a simulation approach. With regard to the quality problem in core assets, we previously proposed a simulation model for estimating project delay and its variability in SPL development [7]. Though the model was validated to be capable of estimating reasonable project delay and its variability, it did not consider requirements volatility and rework caused by defects injected during product projects as sources of project delay.

Literature shows that avoiding project delay is sometimes considered to be a high priority for project success [11]. For such projects, delay will be avoided even though much additional cost is required. Cost overrun estimation for time-boxed projects therefore should be studied. However, in our previous model, any piece of adaptive rework causes project delay, which in practice may be resolved without delay but with additional cost.

Even in the literature concerning effort estimation and simulation in SPL development, these problems have not been sufficiently considered [2,12,13,14,15]. In this paper, we propose a project cost overrun simulation model for time-boxed SPL development by enhancing our previous model. The major enhancements from the previous model include: consideration of requirements volatility, consideration of unplanned work for defect correction during product projects, and nominal project cost overrun estimation. A homogeneous Poisson process is introduced to represent requirements volatility. We use a similar technical approach as in our previous model to represent effort of defect correction during product projects. We estimate nominal cost overruns in month scale instead of

[1] The meaning of "adaptive rework" in this paper and that of "adaptive maintenance" in an IEEE standard [8] are somewhat different. Adaptive maintenance is defined in [8] as "modification of a software product performed after delivery to keep a computer program usable in a changed or changing environment."