An Experience Factory to Improve Software Development Effort Estimates

Ricardo Kosloski and Káthia Marçal de Oliveira

Universidade Católica de Brasília,
SGAN 916 - Módulo B - Pós Graduação - Campus II,
Brasília - DF – Brasil, CEP: 70790-060
rikoslos@brturbo.com, kathia@ucb.br

Abstract. It is well known that effort estimate is an important issue for software project management. Software development effort can be obtained from the size of the software and the productivity of its development process. Nowadays the Function Point Analysis stands out as an approach largely used for software size estimate, while productivity values are extracted from international historical databases. Some databases show the median value of various productivity projects while others present all data according to some specific characterization of the projects. We argue that defining a framework of characteristics that impact on software project productivity can improve comparison between finished projects and the new ones that need an effort estimate. This article presents an approach to effort estimate with continuous improvement of the estimates using some characterization of the projects. Continuous improvement is based on the use of an experience factory.

1 Introduction

Planning and controlling software development projects demand a lot of attention from the managers. Appropriate effort estimate is an important part of this task. It is particularly important to the organizations, because too high an estimate may result in loosing a contract to a competitor, whereas too low an estimate could result in a loss for the organization [2].

One way of defining effort estimates consists in relating the software size, to the effort required to produce it, by means of a value of productivity [18] (i.e., effort = productivity x software size). The effort is defined in man/hours and is the quantity of work required to realize the software development project [9]. The software size can be defined by its functional content using methods such as the well estabilished Function Point Analysis [17, 18]. The productivity can be computed as the relationship between the effort required for the software construction and the functional size of this software [18].

Productivity values can be found, organized according to some characteristics, in historical databases [13, 19]. However, each organization should set up its own historical productivity database, so as to better reflect its own relationship with its customers in software development [8]. The similarity of old projects with new ones

F. Bomarius and S. Komi-Sirviö (Eds.): PROFES 2005, LNCS 3547, pp. 560–573, 2005.

could thus result in better productivity value itself resulting in better effort estimate of the future projects. Another important part of this work is the definition of appropriate characteristics to compare the similarity of projects.

This article presents an approach to effort estimate based on appropriate characterization of the projects. The characterization of the projects is continuously improved so as to reach a better productivity definition for each new project based on past projects with similar characterization. Continuous improvement approaches for software organizations have been defended by different authors [2,14,20]. In this context, Basili *et al.* proposed the concept of experience factory to institutionalize the collective learning of the organization that is at the root of continual improvement and competitive advantage. Since the main idea of experience factory is to use past experiences and since we believe that the effort estimate of old projects can help in a more accurate definition of new estimates, we decided to base our approach on the experience factory.

In the following sections we present a brief review of the use of productivity for effort estimates (section 2) and experience factory (section 3). Next, we present the definition (section 4) and use (section 5) of the experience factory for software development effort estimate. Section 6 presents our conclusions and ongoing work.

2 Using Productivities in Software Effort Estimates

The productivity can be defined as the ratio between the quantity of work required to develop a software and the size of this software [9,21]. In this case, the effort is accounted in man/hours while the size can be measured by different kinds of metrics (e.g., Function Point Analysis–FPA [12], NESMA [22] and COSMIC [6]). Benchmarking studies concluded that the productivity obtained by sizing the software in function points is more consistent and allow better comparisons [17,18]. Also FPA is a functional software size metric that can have its elements extracted from the requirements of the system, available early in the developing process [7], and FPA is independent of the technology used for implementation [12] and can be used to determine whether a tool, an environment, a language or a process is more productive than an other within an organization or among organizations [2,10].

We found different factors, in the relevant literature, which can impact the software development productivity. Some authors [5,8,21] show that the experience of the software development team in the technology used or in the system's business area, is an important factor for the correct definition of productivity. Others defend that schedule restrictions imposed to the software development [2,23] and software complexity [19,23] impact the productivity. Software size is also a relevant factor in effort estimate and impacts the productivity [2,5,17,21,23]. Maxwell [19] say that the use of CASE tools in software development projects can improve the productivity. Finally, productivity increases with reuse because consumers of reusable work products do less work in the software construction [15].

Two main historical databases for software productivity are proposed by the Software Productivity Research (SPR) group [29] and the Institute of Software