Estimating Allocation of Tests Engineers for Software Development Projects within C.E.S.A.R

Presented at the International Conference on Practical Software Quality and Testing (PSQT 2008 North)

Mariana P. Xavier[1], Melissa B. Pontes[1], Luiz C. Silva[1] Felipe S. F. Soares[1,2]

[1]Centro de Estudos e Sistemas Avançados do Recife – (C.E.S.A.R)
Rua Bione, 220 Bairro do Recife – Recife – PE – Brasil

[2]Centro de Informática da Universidade Federal de Pernambuco (UFPE)
Caixa Postal 7851, Cidade Universitária, CEP 50.732-970, Recife - PE - Brasil

{luiz.silva,mariana.xavier,melissa.pontes, felipe.furtado}@cesar.org.br

Abstract

This paper has the purpose to describe a technique for estimating the allocation of tests engineers to be used within a software development project. For elaborating this technique, it was defined that estimating must not have a linear relation to any particular variances of a project, and then, it must take into account a whole set of variances such as the complexity of a project, developers amount, among other ones which will be described as it follows. This experience was created and carried out by C.E.S.A.R – 'Centro de Estudos e Sistemas Avançados do Recife' (Center of Advanced Systems and Studies of Recife), a private institute of research, development and innovation, CMMI level 3, which is part of the technological pool denominated 'Porto Digital' located in the city of Recife-PE – Brazil.

Keywords

software, test, process, tester, developer, ratio, allocation, estimate

Introduction

The C.E.S.A.R – *'Centro de Estudos e Sistemas Avançados do Recife'* (Center of Advanced Systems and Studies of Recife), a private institute of research, development and innovation, CMMI level 3, created in 1996, works focusing on the creation of software products, processes and services, as well as promoting the growth of companies using technology of information and communication (TIC) [1]. In this paper it is presented a practical experience of estimating the amount of tests engineers necessary for the development of software projects within C.E.S.A.R.

The most recent technological advances have been requiring the market to work with even more complex software applications, increasing consequently the risks over the development process, what requires to software a higher quality level [2].

We believe that a well defined and structured process of software testing may help to increase software quality, since the tests procedures are carried out at the right time and in the most adequate way. Dealing with tests as only one of the steps into the process of software development may bring up some unsatisfactory results as well as it may bring up the idea that the tests are not so important, and may be left for the final step. This way, it is reduced the time for tests activities, and in many cases, these activities are carried out by the developers themselves.

Recent studies show that the earlier it is found an error into software development, the smaller will be the costs for its correction [3]. This way, the tests activities must start as soon as it can be, and must be part of a separated development process, even with the correlation of tests and development activities as it is shown on the picture below:

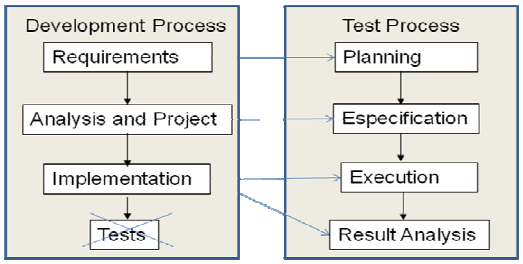

Figure 1 – Parallel Process

In order to cope with this challenge, it is necessary the allocation of tests engineers at the same time it is allocated the developers, the requirements annalists, the quality engineers, among others. However, estimating the amount of tests engineers to be allocated by a software development project is not something so simple. There are several variances into the organization that must be taken into account when quantifying and qualifying this number.

Many organizations estimate this number based only on the amount of developers in a project, maybe because the amount of developers suggest some quantity of produced code, and consequently, suggests the amount of necessary tests, making believe that this relation is something linear [4]. This calculation is uncertain in a way it does not take into account important aspects, like for example, the project complexity, and the time available for its development.

According to Randall W. Rice [5], there is no standard in industry that determines the correct proportion between tests engineers and the amount of developers that may work within any project.

In his book 'Microsoft Secrets' [6], Michael Cusamano states that into Microsoft, the proportion between tests engineers and developers is 1:1, one test engineer for each developer of the project. However, the market indicates that this proportion varies between 1:1,5 and 10:1.

Motivation

In 2005, C.E.S.A.R and Motorola altogether with three other Brazilian institutes of research established a partnership for the creation of BTC - Brazil Test Center, a Motorola test center in Recife, where all its cell phones started to be tested.

Within the implementation of the Brazil Test Center program, it was elaborated a class of professionals highly trained in software testing as well as the generation of advanced processes and the necessary know-how for its implementation. This way, C.E.S.A.R became a worldwide reference in software testing.

Although the organization had already had a high expertise level in software testing, there were some software development projects that did not have any tests engineers in their teams. The tests were carried out by the developers or by third other companies. Due to this fact it came up the necessity of an appropriate allocation of tests engineers within other software development projects into the organization.

In C.E.S.A.R, the team allocation is determined even in a previous phase, before the start of Project Planning, the so called Pre-Sale. It was realized improvement opportunities during this step for the determination of the most adequate amount of tests engineers within C.E.S.A.R projects.

In 2006, during the Pre-Sale of a project for Dell, the C.E.S.A.R suggested the allocation of tests engineers within the project. Firstly, it was estimated the proportion 1:3 (one engineer for each three system engineers), which has already been used by Dell. Although the project was successful, we realized that the amount of tests engineers was not the adequate one, what made tests teams get really overloaded.

Kathy Iberle and Sue Bartlett (2001) [7] carried out a study in which they managed to determine the amount of tests engineers necessary for a development project, but there was the previous condition of existing at least one similar project as reference basis, in which the relation between tests engineers and developers was known, but that was not applied to C.E.S.A.R reality.

Some studies were carried out by GrIT (C.E.S.A.R Independent Test Group) [8] in order to get better resources for deciding the most adequate amount of tests engineers to be allocated within the projects of the organization.

Estimating Technique Proposal

After the projects developed with Dell in 2006, C.E.S.A.R started planning the allocation of

tests engineers for new projects during the Pre-Sale phase. In 2007, the company negotiated with Dell three more contracts for development projects. During these projects, some studies were carried out using C.E.S.A.R history basis, not only including Dell projects, but also several other projects of the organization.

Firstly it was identified the criteria which have had real impact over the definition about the amount of tests engineers for the development projects of the organization. They were listed as it follows:

- System Size: it may increase the amount of tests cases and the amount of time spent on its development. It was taken into account the *small size* (projects until 160 UCP[1]), *medium size* (project with more than 160 UCP until 750 UCP), and *large size* (projects with more than 750 UCP), in accordance to the criteria based on the history basis of the organization for classifying system size.
- Project Duration: it interferes over the testing strategy to be adopted with the use, for example, of exploring, tracking or automatic tests. They were classified into *short* (projects until four months of duration), *medium* (projects over four months and until eight months) and *long* (over eight months), in accordance to the criteria based on the history basis of the organization for classifying the duration of a project.
- Number of releases produced per month: it may require the execution of more regression cycles in a shorter time frame. The values of the criteria are *Zero* (time longer than a month to produce a release), *One* (production of one or two releases per month) and *Two* (production of three or more releases per month).
- System Complexity: it may increase the planning duration as well as the requirements for most complex testing strategies. According to system complexity, the projects were considered as *simple* ones (e.g., a system of standard information for web), *medium* ones (e.g., a system of information using two databases), and *complex* ones (e.g., real time systems).
- Amount of developers: the greater the amount of developers, the bigger amount of produced code, and the bigger possibility of errors into code integration. It was considered projects with until five developers, projects with six to fifteen developers, and projects with more than fifteen developers.
- Types of tests: some projects need several types of tests (functional tests, interface tests, performance tests, stress, usability, among others), and it may require different kinds of expertise and/or specific training. For these criteria it was considered the types of tests used by the organization classifying the projects as with only one type of test, projects with two to four types of tests, projects with five to seven types of tests, and projects with eight or nine types of tests.
- Types of projects: there are two types of projects within the organization: consulting and development ones. Within the consulting projects there is much more research, resulting in more consistent processes as well as in a smaller amount of errors.
- Time limit restrictions: this aspect works as a risk factor when estimating delivery dead lines. The projects were classified into those which will not get into production, and those which will get into production immediately after delivery.

After listing these criteria, they were put in order according to the impact they caused within the organization. For each one, it was given grades in accordance to its level of relevance, varying from 0,1 (less relevant) to 0,5 (more relevant). These grades are used in a pondered sum of criteria, and its result is mapped by the amount of tests engineers (see Table 1).

[1] *Use Case Points*

Results of pondered sum	Amount of Test Engineer
< 3,3	1
≥ 3,3 and < 4,9	2 to 3
≥ 4,9 and < 7,1	4 to 5
≥ 7,1	6 or more

Table 1. Indication of tests engineers total amount

All data was consolidated and it was set an estimating spreadsheet (see Table 2) for tests engineers which was validated within seven (07) projects into the company, including ongoing projects as well as concluded ones (see Table 3 below).

Criteria	Grade	Values	Sub-total
System size: 1 - small 2 - medium 3 – large	0,4	2	0,8
Project duration: 1 - short (until 4 months) 2 - medium (> 4 to 8 months) 3 - long (> 8 months)	0,3	3	0,6
Number of releases produced per month: 1 - zero 2 - between 1 and 2 2 – more than 2	0,5	1	0,5
System complexity: 1 - simple (e.g., a system of standard information for web) 2 - medium (e.g., a system of information using two databases) 3 - complex ones (e.g., real time systems)	0,4	2	0,8
Amount of developers: 1 - until 5 2 – from 6 to 15 3 – more than 15	0,4	2	0,8
Types of tests (functional tests, default recovery, usability, security, performance etc.): 1 - Only one type of test 2 – Between two to four types of tests 3 - Between five to seven types of tests 4 – Eight or nine types of tests	0,4	2	0,8
Types of project: 1 – consulting or research 2 - development	0,1	2	0,2
Time limit restrictions: 0 - the project has a fixed time limit but it may have some delay 1 – there is no possibility of delay. For example, the system will get into production right after delivery.	0,3	0	0
Total			4,4
Amount of Tests Engineers			2 a 3

Table 2. Estimating Spreadsheet

Criteria/Projects	P1	P2	P3	P4	P5	P6	P7
System size	1	2	2	2	3	2	2
Project duration	1	3	3	2	3	2	2
Number of releases produced per month	1	1	2	1	3	2	2
System complexity	2	2	2	2	2	2	2
Amount of developers	1	2	2	2	2	1	2
Types of tests	1	2	2	2	3	1	1
Types of project	2	2	2	2	2	2	2
Amount suggested by the spreadsheet	2 a 3	2 a 3	4 a 5	2 a 3	4 a 5	2 a 3	4 a 5
Amount of tests engineers allocated for the project	2	3	4	3	5	3	4

Table 3. Projects used for the validation of estimating spreadsheet

By the use history basis for the determination of relevance grades, the estimating spreadsheet demonstrated to be adequate for all the projects in which the validation was done without any kind of adjustments.

The document was approved by SEPG (*Software Engineering Process Group*) and published into company process to be used as a support during the estimating done on the Pre-Sale phase of all projects into the organization.

Conclusions and Future Work

The estimating spreadsheet was used within some projects of the organization during the Pre-Sale phase and it has demonstrated to be quite useful for the determination of the amount of tests engineers to be allocated for projects.

For getting these results it was essential the fact of having experience from different projects of the company, making it possible to determine the correct relevance aspect (grade) given for each criteria.

The case study still needs improvement and studies continue to be carried out in order to have a most precise diagnosis. It has not taken into account yet some cognitive and qualitative aspects related to the experience of tests engineers to be allocated for a project. Before the number suggested, it is the Pre-Sale team that defines about the experience level of those people who will be part of the tests team.

Another point of study is about the rump up and rump down of projects that, depending on the tests phase, it may need more or less tests engineers according to the way the tests team is organized into the company. Further details will also be included into the study about applications complexity based on the estimating technique COCOMO II [9]. As new variances are being added to the study, it will be necessary a new calibration.

This experience works into the context of a CMMI quality level 3 organization which has a well defined testing process, and has been allocating tests engineers within its development projects, gradually building a history basis.

By the fact of all organizations have their own particularities, the estimating spreadsheet used by C.E.S.A.R may not be adequate to the context of each one individually, what makes it necessary an analysis over variances and relevance grades.

In cases of companies which are starting to have software testing activities, and consequently have no history basis, the recommendations suggested by the estimating spreadsheet may be a good start with new possible adjustments and calibration.

References

[1] Chagas, U., Gouveia, C. and Didier, A. (2007) "Resultados práticos de V&V dos projetos do C.E.S.A.R.", In: REVVIS, Brasil.

[2] Gouveia, C., Oliveira, J. and Quidute, R. (2006) "A Way of Improving Test Automation Cost-Effectiveness", In: CAST, Indianapolis, USA.

[3] Myers, G., (2004) The art of software testing, John Wiley & Sons, Inc, 2nd Edition.

[4] Rothman, J. (2000) "It Depends: Deciding on the Correct Ratio of Developers to Testers",

http://www.jrothman.com/Papers/ItDepends.html, Março.

[5] Randall, W. (2000) “The Elusive Tester to Developer Ratio”, http://www.riceconsulting.com/articles/tester-developer-ratio.htm, Março.

[6] Cusamano, A. and Richard W. (1995) “Microsoft Secrets” , Simon and Schuster, Inc., New York.

[7] Iberle, K. and Bartlett, S. (2001) “Estimating Tester/Developer Ratios (or Not)”, In: Proceedings of the Pacific Northwest Software Quality Conference, USA.

[8] Gomes, K., Pontes, M. and Silva, L. (2008) “GRIT: A Practical Experience of the Independent Software Test Group at C.E.S.A.R.”, In: PSQT Conference, USA.

[9] Boehm, B., Abts, A., Brown, A., Chulani, S., Clark, B., Horowitz, E., Madachy, R., Reifer, D. and Steece, B., (2000) Software Cost Estimation with Cocomo II, Prentice Hall, 1fs edition.