CMMi for Small Business: Initial Tailoring of a Mexican organization.

Francisco Alvarez R, Jaime Muñoz A, [1] ,and Alfredo Weitzenfeld R. [2]
[1]Universidad Autónoma de Aguascalientes, Centro de Ciencias Básicas, Av. Universidad No.940, Aguascalientes, México.
fjalvar@correo.uaa.mx, jmunoz@correo.uaa.mx
[2] Instituto Tecnológico Autónomo de México, Departamento de Ciencias Computacionales, México.
alfredo@itam.mx

Abstract.

The main objective of this work is analyzing the applicability of CMMi (Capability Maturity Model Integration) in developing a detailed software process for small organizations. The implementation of mature methods is hard, requiring highly trained and experienced people. It is usually complicated to apply evolved software processes in smaller organizations due to their lack of mature cultural habits in the use of techniques, tools, metrics, quality patterns, etc., in other words, complete methodologies. Thus, it is essential to incorporate a small but appropriate set of software tools in supporting complete engineering software processes in small organizations.

1. Introduction.

A software development process comprises a group of people, organization structures, rules, politics, activities, software components, methodologies and tools specifically used or created to conceptualize, develop, serve, innovate or extend a software product [10]. Additionally, a software process is a group of activities and associated results leading to the creation of a software product [18]. Software processes have been designed to support a number of development tasks, from the creation of new products to the extension and modification of existing ones. In general software processes are complex, based in particular organization needs and according to the people involved in them; thus the difficulty in standardizing processes and the limitations in many of the tools designed to automate the development process. Historically, a large number of the employed software processes are based on the very classic waterfall and evolutionary development models [17], or the more recent spiral and win-win models [5][6].

One of the main challenges with respect to software processes is how to improve their capabilities, something best achieved through standardization and precise definition. A related problem involves the definition of factors to consider in the construction of a process standard for smaller projects, since these kind of projects have different characteristics from larger ones, including more limited resources and development time [14].

Among the different processes defined to increment the personal productivity, the Personal Software Process (PSP) [12] is a framework helping software engineers measure their personal work in terms of the following aspects: Planning, Design, Coding, Compilation, Tests and Postmortem, in addition to guides that indicate the activities to follow. Since this is a personal process it can be applied to smaller organizations. Team Software Process (TSP) [13], is the solution to the integration of the PSP, however it is at development level, since results of the application of this technique have not been extensively reported. TSP addresses the situation that most software projects are developed by teams. TSP is a software development technology having a framework based on work teams with the following objectives: (1) develop products in several cycles, (2) provide metric for teams, (3) evaluate lists and teams, and (4) offer guides for the solution of problems in teams.

One of the objectives in process development is to improve product quality. For IEEE, quality is defined as the degree to which a system, component or process complies with its specified requirements, in addition to the client or user needs and expectations [2]. In the ISO 9000:2000 norm specification, quality, defined as either poor, good or excellent, is the degree to which a group of inherent characteristics comply with requirements [19].

A well understood and accepted process, based on event measurement and prediction, is better able to control software production [9] thus resulting in quality products.

The elements that affect the quality of a software product include:

1. The client / user, taking into account final product (system) aspects while participating in the software development process.
2. The developer, as responsible for product development and quality assurance.
3. The process (previously defined).
4. The product, taking into account cost, time and effort equilibrium.

These elements are closely related, determining not only product engineering aspects but also the organization, support and administration, defined for almost all existing quality models [11].

To obtain a software quality model with a similar approach to the product and development process it is necessary to establish first a consolidation and then an improvement, in other words the organization needs to establish the way in which the product is produced as well as its standards. Once this is known it can then be improved.

Four aspects can be considered in this improvement: (1) process consolidation in order to achieve a standardization (similar to those offered by ISO), (2) product consolidation evaluated by well known metrics, (3) product improvement moving from product quality to service quality during the complete software life cycle, and (4) process improvement based on separate process maturity models such as CMMi, SPICE, PEMM, etc.[15].

Thus, the importance of a well defined and mature process supporting quality development such as CMMi.

2. Antecedents of CMMi (Capability Maturity Model Integration) in Small Businesses.

A frequent conception about adopting Capability Maturity Model Integration (CMMi) is that it works only for large organizations (its cost and complexity appear to make it impractical for smaller organizations to implement) [3].

Some organizations in USA recently had implemented the CMMi, for example Analytical Service Inc. [20], have three process areas (PA's) of CMMi as part of a pilot study with the Software Engineering Institute (SEI). The organization said CMMi might even be more beneficial to smaller businesses because it allows them to grow more consistently and to make changes less costly, that is, "before growth demands them".

ASI and Cirrus Technology Inc. [20] are two Hunstville, AL, companies that participated in a recent study to develop a business case and technical guidance for small to medium sized enterprises, defined by the study as companies with 25 to 250 employees that wish to adopt CMMi. Initial results from the pilot study look promising: both organizations described significant benefits from using CMMi, especially in the areas of project management and changes management, and both are in the process of documenting and disseminating their finding so others can learn from their experiences.

The pilot, launched in July 2003, is part of joint project between SEI and the U.S. Army Aviation and Missile Research and Development Center (AMRDEC) Software Engineering Directorate in Hunstville. Suz Garcia, [20] a member of the piloting team from the SEI, said that the pilots help support a business case for deploying CMMi in smaller companies.

3. Initial Tailoring with CMMi in a Mexican organization.

A large number of the organizations that develop software around the world are small in their size (between 10 and 100 employees). For example, 87% of the companies devoted to software development in Mexico have between 7 and 60 employees [1]. Many of the smaller companies oppose the CMMi model due to the expensive compliance effort, both in time and money. Some of the shortcomings are [8]:

1. Produces excessive documentation.
2. Defines an extensive number of Specific Practices (SP).
3. Requires extensive resources.
4. Involves high training costs.
5. Defines practices independent of project type.
6. Lacks guidance in satisfying project and development team needs.

Among the shortcomings encountered in applying CMMi to software process improvement in smaller organization, it was identified that many of the SP's do not apply, such as Supplier Software Subcontract Management. Almost all small projects do not require external services [14], while other ones require tailoring [7]. These difficulties are closely related to the underlying philosophy of CMMi.

In small teams the quality of the development group is very important for favorable results, high levels of abilities and experience generates quality in the products [4]. The team cannot dedicate a lot of time to administrative procedures or documentation. The time is largely dedicated to the design, programming and tests of the product (construction of the software).

Some research with teams of similar size (smaller to 10 people) that adjusts the CMMi in small projects through a tailoring of the process [16]. The results are reported in documentation reduction and invested effort, maintaining the quality of the software, reducing the cycle through control formats and documentation in short projects.

The case study analyzed involves a small organization (10 people), where 4 stages (group of activities to implant CMMi) were followed during 12 months.

The first stage involves tailoring of existing software processes. The number of activities in phases is reduced and the effort invested is also decreased in terms of project management.

The second stage is the interpretation and adaptation of Specifics Practices, the steps are: (1) reduction of roles: Software Engineer, Software Quality Assurance, Project manager, (2) development of an activity diagram adjusted to small organizations, and (3) design of artifacts or products for each SP analyzed.

The third stage involves the design of a training program and reorganization of roles. Since human resources are usually limited in small organizations, different people can be involved simultaneously in several projects in a part time basis, by playing multiple roles in the same project. There should be well-defined rules in order to avoid any conflicts due to multiple roles. There are restrictions on SQA roles and members of test teams, for example, the SQA team should be different from that involved in software development.

4. Conclusions

Evidences exist on the assertive application of CMMi in projects and small teams, although a complete solution is not yet presented to a needed adjustment for the Model, the factors that consider the reduction of the process are:

- Reduce the phases considering the necessities of a short project.
- The consideration of priority of activities in small team development and construction of software.
- Reduce the control formats and documentation to facilitate the administration of the project.

The process development organization has been initially applied in a software development company obtaining to the date the following results:

- The application of CMMi to small organizations is possible and contributes to the improvement of their software process.
- The efficiency of the software process was measured for Specific Practices achieving.

As can be seen from this work, it is necessary not only to have a CMMi model tailored to small organizations but also a corresponding evaluation criteria.

5. References.

[1] Amiti / Bancomext. Outline of government support to the software industry. Secretaría de Economía, Mexico. (In spanish). 2001.

[2] An American National Standard, IEEE Standard fos Software Quality Assurance Plans. IEEE Computer Society. 1984.

[3] Batista J. and Dias de Figueiredo A. "SPI in a Very Small Team: a Case with CMM". Software Process Improvement and Practice 2000; 5: 243-250.

[4] Basili, V.R. and Rombach, H.D. "The TAME project: towards improvement-oriented software environments". IEEE Trans. On Software Engineering, 1998, 14(6), 758-773. (Caps. 24, 25).

[5] Boehm, Barry, "A spiral model for software development and enhancement", Computer, Vol. 21, No. 05, May 1988.

[6] Boehm, Barry., "Using the Win Win Spiral Model: A case Study", Computer, Vol. 31, No. 07, July 1998.

[7] Brodman JG and Jonson DL. 1992. "Software process rigors yield stress, efficiency". Signal Magazine, 55, August.

[8] Brodman JG and Jonson DL. 1995. "Tailoring the CMM for small business, small organizations, and small projects". Proceeding of the 7th Annual Software Technology Conference, April.

[9] DeMarco, T., *Controlling Software Projects*, Prentice Hall. 1982.

[10] Fugetta, A. *Processo Software*, Aspetti strategici e organizzativi, il Cardo editore in Venezia. 1994.

[11] Huff, K. *Software Process Modelling, in Software Proces*, Eds. A. Fuggetta and A. Wolf, J. Wiley & Sons Ltd. 1996.

[12] Humphrey, W. *Introduction to the Personal Software Process*. Pearson Education. USA. 1997.

[13] Humphrey, W. *The Team Software Process (TSP).* Software Engineering Institute, 2000.

[14] Leung H and Yuen, T., 2001. "A Process Framework for Small Projects." Software Process Improvement and Practice, 6:67-83.

[15] Printzell, Ch. and Conradi, R. "A taxonomy to compare SPI Frameworks". EWSPT 2001, LNCS 2077, pp. 217-235. Springer – Verlag. 2001.

[16] Richardson, I. "Software Process Matrix: A Small Company SPI Model". Software Process Improvement and Practice. 6: 157-165. 2001.

[17] Royce, W. "Managing the development of large software systems: concepts and techniques". *Proc IEEE WESTCON*, Los Angeles, CA. (Cap. 3). 1970.

[18] Sommerville, I. *Software Engineering*, Pearson Education Limited, Sixth Edition, England. 2001.

[19] Tingey, M., *Comparing ISO 9000, Malcom Bladrige, and the SEI CMM for software: a reference and selection guide*. Prentice Hall, 2001.

[20] Lauren, Heinz, *CMMi for Small Businesses: Initial Results of the Pilot Study*, Software Engineering Institute, 2004.