Improving the Software Development Process:
A Dynamic Model Using the Capacity Maturity Model

Ewandro Araujo, Luc Cassivi, Martin Cloutier and Elie Elia

Department of Management and Technology
École des Sciences de la Gestion
Université du Québec à Montréal
P.O. Box 8888, Downtown Station
Montreal, Quebec, H3C 3P8, Canada
Tel.: +1 514 987 3000 ext. 4084
E-mail: cassivi.luc@uqam.ca

Abstract

Regardless of their size, software firms search for better methods to improve the delivery of their projects. The SEI Capability Maturity Model (CMM) is one available framework employed to assist in improving this process. The challenge of identifying the benefits associated with implementation of CMM Level 2 practices for the smaller software development firm is the main focus of this research. The objective is to evaluate the impact of each key process area of CMM 2 on productivity, product quality and ability to meet deadlines. A simulation model is designed to help researchers in software development, and management teams in SMEs, understand the impact of alternative management policies and practices according to CMMD. The results indicate that the CMM's software quality assurance process area has a sizeable impact on productivity and that all CMM process areas impact scheduling activities. The process areas associated with project management (software project planning and software project tracking) have very little impact on product quality as opposed to the other process areas with impacts more substantial on this performance measure. The analysis of scenarios indicates that the adoption of CMM2 practices based on requirements management yields more positive results than policies based on project management.

Keywords: Software development, SME, CMM, performance

1. Introduction

Large firms such as Microsoft, Oracle, SAP or IBM are well-known for large-scale software development endeavors, but small and medium enterprises (SMEs) also supply an array of software to the market. Large or small, companies that develop software are quite similar, notably on their unenviable history of sluggish project success (Keil and Robey, 2001). The SEI « Carnegie Mellon - Software Engineering Institute » pinpoints this fact to the inability of software development firms to manage and better control their software development process (Paulk et al., 1993).

Different solutions have been identified to address this problem, one of which is the use of the Capacity Maturity Model (CMM) to guide companies in meeting their target. However, the complexity of the software development process often obscures the benefits of the CMM approach and the identification of the sources that lead to better performance, this being particularly true for SMEs. Without a proper identification of these sources, the complexity of the development processes will continue to grow, making it difficult for companies to understand the consequence of their decisions over time. Although a CMM certification for an SME is important, it becomes difficult for a manager to fully understand the impact of such a certification, which may lead to the interruption of such a certification, or to a missed CMM implementation.

The main objective of this research is to understand the impact of the CMM method, which improves the software development process, on the performance of an SME through the use of a system dynamics model. Such a model should enable managers to fully understand the behavior of performance measures in software development for projects evolving from a CMM1 level to a CMM 2 level.

The remainder of the paper is organized as follow. The next section presents the theoretical background while section 3 focuses on the research method, namely the principal steps of the study and the model extension. Section 4 then describes the results of the study by comparing different scenarios and thus understanding the influence of a CMM2 certification for small and medium sized software development firms. Finally, research contributions, limits and future research avenues conclude the paper.

2. Theoretical background

Although the software development field has progressed over the last few years with techniques such as software measurement and development methods (methodologies), the success of software development projects remains an important challenge for the industry (Keil and Robey, 2001). The literature is rich with examples portraying the potential causes of failure in software projects: products non compliant to specifications, high maintenance work, important delays according to projection, etc. (Gibbs, 1994; Linberg, 1999).

In the latter part of the 1980s, the SEI identified the principal source of these problems that were afflicting the software industry, namely: the inability for firm to manage the software development process. The institute also noticed that the benefits of using new methods or technologies could not be reached in a undisciplined and chaotic project management context, which is unfortunately often the case in the industry (Paulk et al., 1993).

In 1991, in order to assist organization with this process, the SEI developed a first version of CMM (Paulk et al., 1991). The model included a repository of planning, engineering and management practices that aimed at improving the software development processes. The literature that followed on the subject all praised the benefits of CMM (Humphrey et al., 1991; Wohlwend and Rosenbaum, 1993; Diaz and Sligo, 1997; Herbsleb et al., 1997), which became one of the widespread models to guide firms towards improved software development processes.

2.1 The Capacity Maturity Model - CMM

Developed in the United States by the SEI after receiving a request from the DOD (Department of Defence), CMM is built on the concepts of Total Quality and Continuous Improvement, contextualized in software development projects (Paulk et al., 1993). The literature shows that firms engaged in the improvement of their software development process using the CMM framework can drastically enhance the quality and performance of their projects in terms of product quality, customer satisfaction, respected deadlines, etc.

However, a good majority of these articles are anecdotal or introduce case studies on the subject (Humphrey et al., 1991; Wohlwend and Rosenbaum, 1993; Diaz and Sligo, 1997). Goldenson and Herbsleb (1995) were the firsts to present the results of a survey on the implementation of CMM. Their study, which included 167 respondents from 61 firms in North America, led to similar findings (see Figure 1). The potential of CMM is undeniable, but its application in a firm's environment often is difficult and complex.

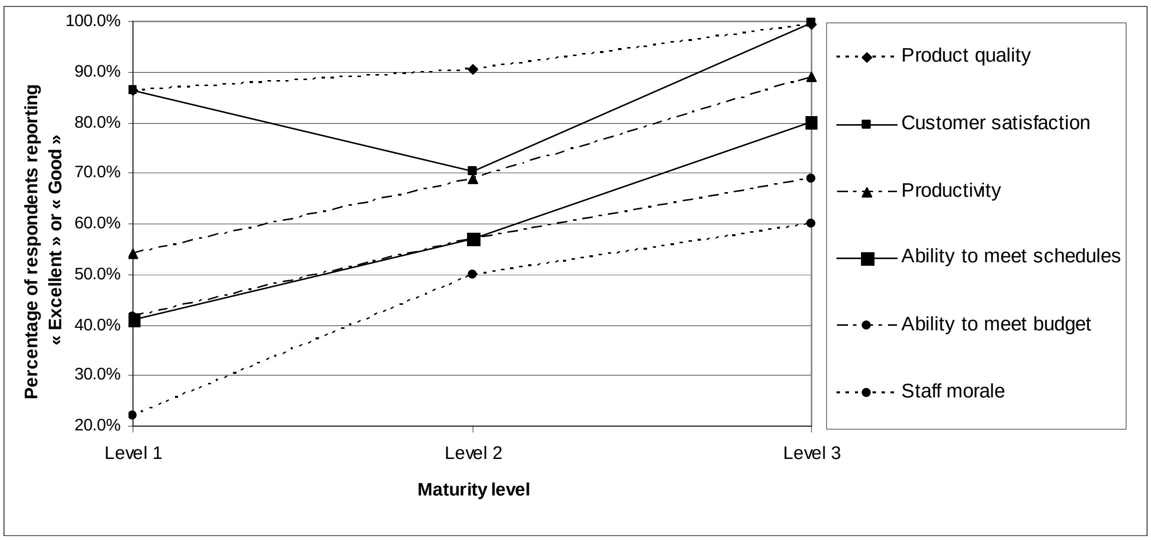
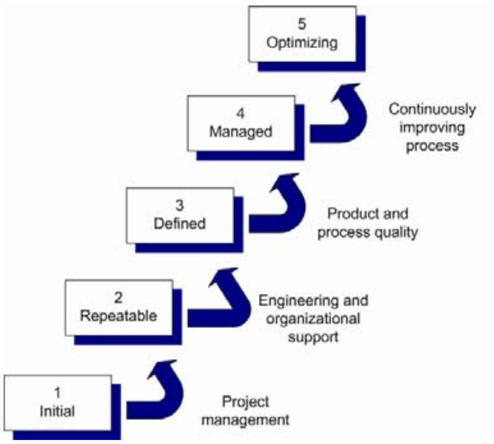


Figure 1 - Quality and performance indicators per maturity level of CMM; Goldenson and Herbsleb (1995)

Process, capacity and maturity

The software development process is defined as an set of technologies, methods and practices used to produce a software (Humphrey, 1990), while the development process' capacity is perceived as a means to anticipate and control the outcome of a new software development project (Paulk et al., 1993). The maturity, on the other hand, involves a potential growth in the development process' capacity, and highlights the richness of the process and the standardization in the application of the process in a firm's portfolio of projects (Paulk et al., 1993).

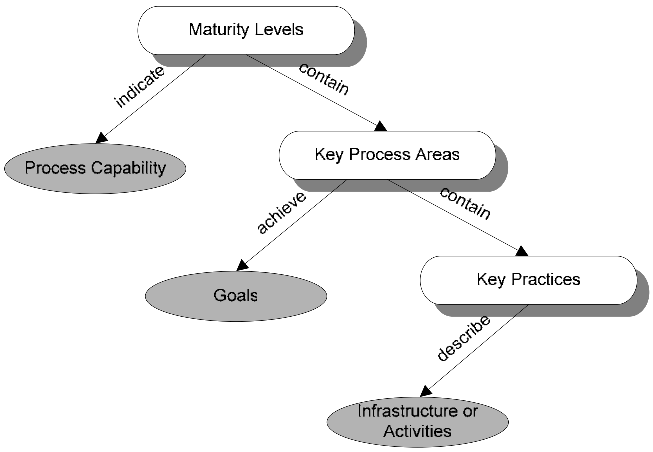
In CMM terms, an immature software development firm basically improvises its software development process each time. Structured project plans and formal project tracking are basically inexistent, which leaves the project's outcome to the effort and competence of particular individuals involved in the project. Mature firms, on the other hand, master and manage effectively the software development process. CMM is comprised of five maturity levels, which is a hierarchy of the competencies required by a software development firm, and forms the successive basis to continuous improvement of the process (see Figure 2).



Source: Paulk et al. (1993)

Figure 2 - The Five Levels of Software Process Maturity

At level 1 (Initial), the development process rests on improvisation where a few sections of the global process are defined. The success of the process lies on the motivation and competencies of key individuals in the firm. At the second level (Repeatable), the procedures are put in place in order to execute project planning, costs evaluation and the identification of functionalities. The process is hence based on the repetition of previous processes in successful and similar projects. At level 3 (Defined), project management and software engineering activities are documented, standardized and integrated in a coherent manner. At level 4 (Managed), measures of both the development process and product quality are identified and gathered, which enables a firm to forecast potential tendencies of the process, and then react accordingly. Finally, at level 5 (Optimizing), the firm focuses on the development process' continuous improvement. The firm possesses the means required to identify and solve the weaknesses of the software development process.



Source: Paulk et al. (1993)

Figure 3 - The CMM structure

For each maturity level, except the initial level, key process areas indicate where firms should concentrate their efforts to improve the software development process (Figure 3). These key process areas identify activities that, when executed simultaneously, should allow objectives to be reached.

The SEI document (*Capability Maturity Model, Version 1.1* (Paulk et al., 1993)) presents the key process areas for the 4 levels of CMM, which are summarized in Table 1.

Table 1 - Key process areas for each level of CMM

CMM 2	**CMM 3**
Requirements Management Software project planning Software project tracking and oversight Software subcontract management Software quality assurance Software configuration management	Organization process focus Organization process definition Training program Integrated software management Software product engineering Intergroup coordination Peer reviews
CMM 4	**CMM 5**
Quantitative project management Software quality management	Defect prevention Technology change management Process change management

Each key process area identifies a cluster of related activities that achieve a set of goals when carried out collectively (figure 2). The key practices depict the infrastructure and activities that contribute to the implementation in the area. The objectives summarize the main activities of a process area, and also enable the assessment of the realizations through the analysis of the activities encountered during the project.

2.2 Software development in SMEs

The software development industry occupies an important place in Canada’s economy. In 2001, firms in this industry had more than 128 000 employees with revenues of 18.6 Billion $CAN, most of which were SMEs (99,5%)

The literature on software development SMEs is abundant, covering diverse themes such as improvisation and its role in small software organizations (Dybå, 2000). Other authors such as Kamsties et al. (1998) and Nikula et al. (2000) studied software requirement engineering practices while others have focused on software development process improvement (Kelly and Culleton, 1999; Otoya and Cerpa, 1999; Villalón et al., 2002).

SMEs and CMM

Software development SMEs have been hit by the tough reality of the sector. High costs and delays, non compliant products and other problems are frequent in SMEs of this sector. Consequently, their interest for the improvement of the development process and for CMM is growing as smaller companies are often sub-contrators (Brodman and Johnson, 1994), a position that often requires a CMM certification to obtain contracts (Baker, 1996). International competition, for instance in India and Russia where several firms are certified at high levels of CMM, has also pushed North American SMEs (and in other regions) to adopt CMM (Scott et al., 2004).

The implementation of CMM in large firms is complex, but the challenge for SMEs is even more important as they have to face specific constraints (financial, technological and organizational) that add to the complexity of the implementation. The culture of small firms is often refractory to the implementation of formal processes (structured project plan, documentation, etc.) as the work process of employees lead to the situation that they touch just about every aspects of software development in an SME. Employees see the introduction of these procedures as limiting creativity and innovation, which often are key elements to the survival of SME (Kelly and Culleton, 1999).

The lack of time and human resources often is linked to an SME's difficulties to follow the CMM recommendations and requirements. Due to these constraints, SMEs often deal with external consultants to identify and assess the appropriate CMM level. At this stage, SMEs require strong support to evolve towards higher levels in the CMM model; and this is especially true for the implementation of modalities specified in the model (Goldenson and Herbsleb, 1995).

2.3 Systems dynamics and software engineering

Systems dynamics is applied to a number of different problems in software engineering. Christie (1999) presents a few examples such as the assessment of project costs, the impact of adopting new software development policies (practices), as well as a project's post-mortem analysis.

Abdel-Hamid and Madnick (1991) developed a software development project simulation model that aimed at formalizing procedures, management practices and policies involved during the development process. The model includes four sub-sections: software production, control, planning and human resources (Figure 4).

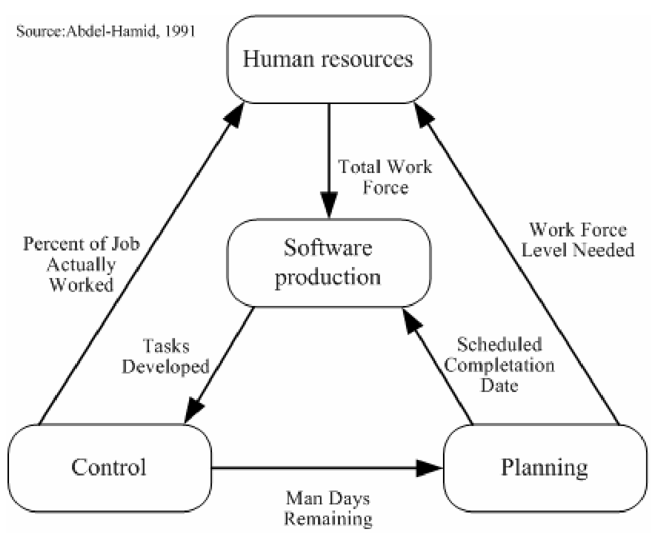


Figure 4 - The four sub-systems of Adbel-Hamid and Madnick's model (1991)

Due to its complexity, the « software production » sub-system was further decomposed into four sub-sectors by Adbel-Hamid and Madnick: Software Development Productivity Sector, Quality Assurance Sector, Testing Sector and Manpower Allocation Sector (Figure 5).

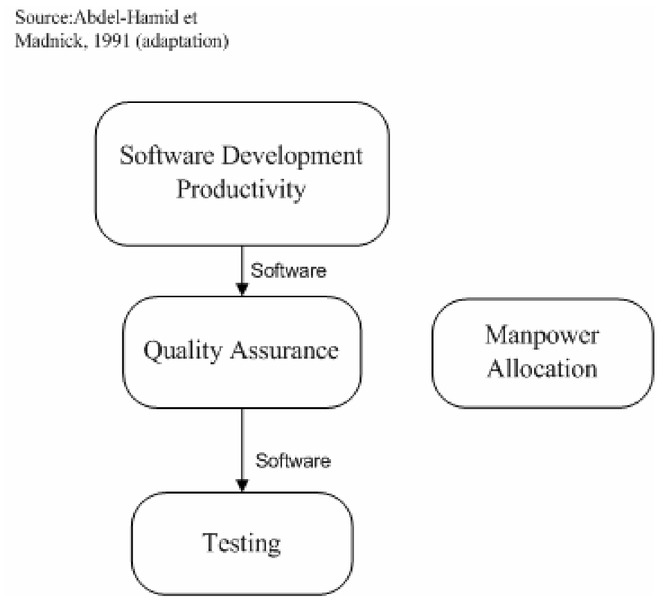
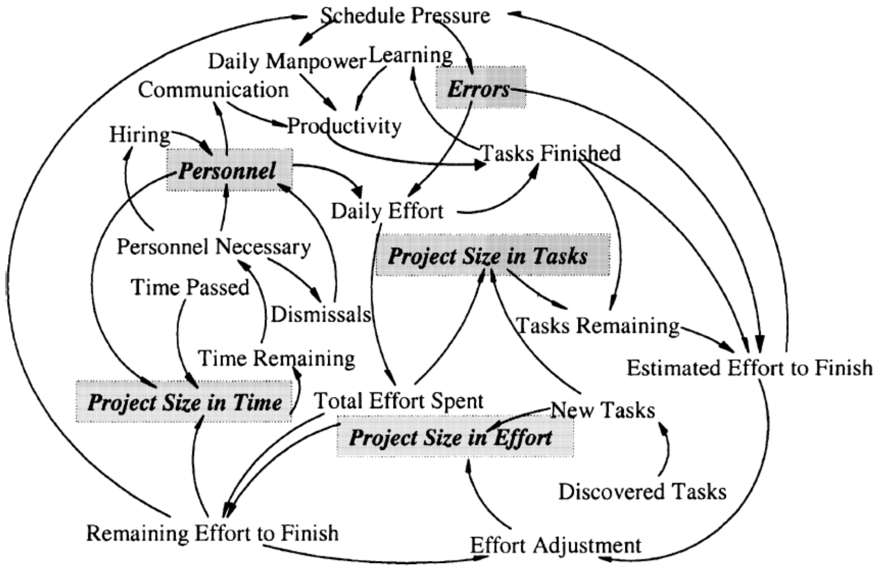


Figure 5 - Sectors of the Software Production sub-system

Abdel-Hamid and Madnick's model (1991) is a key reference in the field. A number of dynamic models are based or were inspired by their model. These models are mainly focused on the expansion of capacities and of applications of Abdel-Hamid and Madnick original model. Rus *et al.* (1999) simulation model aimed at testing software reliability according to the planning and management of a software development project. Pfahl and Lebsanft (1999) combined SD, static modeling techniques and quantitative modeling in order to develop a model that analyzed the effects of unexpected change requests on the software projects performance. Kahen et al. (2001) also developed a system dynamic model that investigated the software evolution process. Finally, Ruiz *et al.* (2001) in an attempt to simplify Abdel-Hamid and Madnick's work, developed a model of software project dynamics (RDM) in the initial phases of a project where the availability of information is often reduced. Figure 6 presents Ruiz *et al.*'s (2001) simplified causal loop diagram in which three main software development process variables are put forth: personnel required for the development process, number of errors in the developed software and software development project size. Three different means are used to determine the project size variable: number of tasks, time and effort required to execute the tasks.



Source: Ruiz et al. (2001)

Figure 6 - Simplified causal loop diagram of the RDM

3. Research Methods

The complexity of a software development process makes it difficult for decision makers to establish a reliable mental model that would anticipate the outcome of modifications carried to the process. The high level of feedback in software development, like many other management processes, is the cause of these challenges (Christie, 1999). Project planning methods such as the Gantt charts offers a static view of the context and doesn't take into account the effects of possible retroactions. Methods based on costs and efforts, such as COCOMO, only examine a few aspects of the software development process, and are not equipped to take on the analysis of an entire process. Finally, the traditional experimental approach, that is of implementing changes to the process (in respect to CMM) and then analyzing the consequences of these changes during the project, is costly for a SME.

Hence, a simulation method was chosen to conduct this research. Systems dynamics is an approach that handles the weaknesses of the other methods mentioned above. It also enhances problem solving and understanding in a software development process (Kellner et al., 1999). Christie (1999) also adds to the pertinence of this method by declaring using simulation approaches is of utmost importance for managers trying to understand CMM.

3.1 Research steps

Five steps were required to conduct this research.

i- At first, different dynamic models pertaining to software development projects (Abdel-Hamid and Madnick, 1991; Rus et al., 1999; Pfahl and Lebsanft, 1999; Donzelli and Iazeolla, 2001;

Kahen et al., 2001; Ruiz et al., 2001) were analyzed in detail to identify a model that could fit de software development process in SMEs. The analysis focused on the constraints of SMEs during software development such as the absence of historical data on similar projects, which led to the identification of a model that answered our needs. Ruiz et al.'s (2001) RDM model was selected in part due to its ability to integrate information availability.

To evaluate the potential use of RDM in an SME context, it was decided to calibrate the model using data obtained from a Brazilian SME specializing in software development. This firm had recently evolved from the CMM1 level to CMM2. The test confirmed the selection of RDM as the reference model in this research project.

ii- The second step consisted of creating and recalibrating the RDM model using Powersim (simulation software). In order to do so, the RDM equations published in Ruiz et al. (2001) "Modelo Dinàmico Reducido" were inserted into Powersim. Some equations were missing and some rebuilding was required. This effort was undertaken using Adbel-Hamid and Madnick's model (1991), which is the base model of Ruiz et al.'s (2001) work.

iii-The next step involved the creation of the extended model that allowed the simulation of the software development process of a firm at the CMM2 level. The literature of the key process areas of CMM2, along with the more generic literature on software engineering and software development improvement methods were the main sources for the extension of the model. To resolve certain incomplete links in the model, Ford and Sterman's (1997) "Expert Knowledge Elicitation" method, which proposes three sequential phases (positioning, description and discussion) were followed. Four experts in software project management, each with more than ten years of experience in the industry (all of which had more than five years of experience in SMEs) participated in the study. This step is detailed in section 3.2.

iv- A fourth step was then required to test the extended model. A series of eight tests proposed by Sterman (2000) were conducted. The procedures adopted in this step are: 1) Structure Assessment Test, 2) Dimensional Consistency Test, 3) Parameter Assessment Test, 4) Extreme Conditions Test, 5) Integration Error Test, 6) Surprise Behavior Test, 7) Sensitivity Analysis Test and 8) Behavior Reproduction Test. Overall, the tests executed on the extended model confirmed that the model had a behavior that respected the limits and constraints of a software development SME involved in CMM certification. The results also demonstrate a reasonable level of confidence of the model.

The data required to run this last test (Behavior Reproduction) was obtained once again by the same Brazilian SME, which enabled two different behavior reproduction tests, one for Level 1 of CMM and the other for level 2 of CMM. The other important test (Sensitivity Analysis) allowed us to analyze the impact of each of CMM2's key process area (in terms of level of effort) on software project performance measures. In order to identify the performance measures to be analyzed in this study, the same adopted approach for the extension of the RDM was undertaken, which lead to the following measures identified by Goldenson and Herbsleb (1995): Productivity, Product quality and Ability to meet schedules. The sensitivity analyses also helped identify two interesting scenarios used in the next step.

v- Finally, in the fifth and final step, the extended model was run to analyze the management policies and practices of a CMM2 certification. Using different CMM2 management policies (once again based on the Brazilian SME), project performance outputs were examined. Simulation involved two different scenarios composed of specific combinations of level of efforts for each key process areas of CMM2. These scenarios were then compared to a status quo (that is, without a CMM 2 certification (CMM1 initial)).

In the next section further details are given regarding step 3: extending the reference model to CMM2.

3.2 Extension of the reference model to CMM2

When Ruiz et al. (2001) used Abdel-Hamid and Madnick's (1991) model on a Spanish company, they quickly sensed the complexity or difficulties related to determining the initial values of the parameters and functions of the model. The absence of historical data and the important number of parameters and functions makes it very impractical to use, which is why Ruiz et al. (2001) developed a simplified model (MDS) that could be amenable for use in these particular situations.

As in Abdel-Hamid and Madnick's (1991) model, RDM is composed of four sub-systems: software production, control, planning and human resources (see Figure 4 presented earlier). With the exception of the planning sub-system, all other sub-systems were simplified. Table 2 presents a comparison of both models (RDM; Abdel-Hamid and Madnick) in numbers, and specifies the simplification (as a percentage of reduction).

Table 2 - Comparing the simplified RDM model to Abdel-Hamid and Madnick's

	RDM (Ruiz et al.)	**Abdel-Hamid and Madnick**	**Percentage of reduction**
Number of variables	67	138	48,6%
Number of parameters	19	37	51,4%
Number of functions	16	27	59,3%
Number of equations	127	237	53,6%

Source: Ruiz et al. (2001)

Table 3 presents an overview of the extension made to the RDM model in order to introduce CMM2 process areas in the model. In this extended model, five new parameters, two new equations and six new functions were added to the model. The extended model also required the modification of a number of equations (13) in all four sectors, but mainly in the software production sector (9 equations).

Table 3 – Overview of the extension of the RDM model to introduce CMM2 activities

Elements of the extension	Total number of	In human resource sector	In control sector	In planning sector	In software production sector
New parameters	5				5
Replaced parameters	1		1		
New equations	2		1	1	
Modified equations	13	1	2	1	9
New functions	6		1	1	4
Modified functions	1		1		

4. The extended model: an instrument to analyze CMM2 policies

To analyze CMM2 policies and its impact on a firm, two scenarios were identified to evaluate the impact of two policies on performance measures (Productivity, Product quality and Ability to meet schedules).

4.1 Scenario building

The scenarios were build on two different software development strategies that firms in the sector can follow. The first strategy focuses on prevention policies during the initial steps where requirements management is key. The second strategy is built around controlling the project, where project management activities such as planning and monitoring are key elements of the strategy. These two scenarios represent two CMM2 policies that could be followed by management. According to the five experts who participated to the study, it is highly unlikely to see more than half of the efforts of a software development process assigned to CMM2 activities. Commonly, efforts assigned to CMM2 activities in such a projects are estimated to be 30 to 35% of the total project by the experts. Hence, both scenarios used in this research consider that the CMM2 activities make use of 35% of the total effort allocated to the project. Table 4 presents some of the initial data used to run the model. Table 4 presents the main parameters used to execute the model (based on the characteristics of the Brazilian SME in the software development sector). Other parameters used in the simulation are the same as the ones used by Abdel-Hamid and Madnick (1991) and Ruiz et al. (2001).

Table 4 – Main characteristics for the simulation

Project characteristics	
Project size, in tasks	122 tasks
Maximum of personal assigned to the project	3 persons
Project implication (assumed by the project leader)	80%
Estimated project timeline for development	102 days

Scenario A – Prevention policy: priority on requirements management

Scenario A, based on prevention, put emphasis on the identification and comprehension of requirements by allotting to this process area (requirements management) approximately 43% of the total effort dedicated to the CMM2 activities (or 15% of the total effort of the project activities as shown in Table). The other areas of CMM2 for this scenario all receive similar attention: 5% of the total effort assigned to activities the project activities or approximately 14% of the total effort dedicated to the CMM2 activities.

This strategy is based on the experience of the Brazilian SME that participated to this study. A vast literature presents the merits and importance of requirements management (Kamsties et al., 1998; Standish Group, 1998; Leffingwell and Widrig, 2000; Nikula et al., 2000; Wangenheim et al., 2003).

Table 5 - Scenarios for the analysis of CMM2 policies

Key parameters – CMM 2 level	Scenario A Prevention policy (requirements management)		Scenario B Control policy (project management)	
	% of total project effort	% of CMM2 effort	% of total project effort	% of CMM2 effort
Percentage_Effort_ Requirements Management	15%	43.2%	5%	14.2%
Percentage _Effort_ Planning	5%	14.2%	10%	28.7%
Percentage _Effort_ Monitoring	5%	14.2%	10%	28.7%
Percentage _Effort_ Configuration Management	5%	14.2%	5%	14.2%
Percentage _Effort_ Quality_Assurance	5%	14.2%	5%	14.2%
Total effort dedicated to CMM2 activities	35%	100%	35%	100%

Scenario B – Control policy: priority on project management

Two key parameters of the model (Planning and Monitoring) are the key elements of scenario B based on project management. In table 5, the effort assigned to these two areas is doubled what it was for scenario A, which combined is worth approximately 57% of the effort dedicated to the CMM2 activities (or 20% of the total project effort).

This strategy, often followed by firms in the industry, is often due to the inadequate (or lack of) customer participation in the identification of proper requirements and needs. The customer is often not involved in this critical activity and is replaced by inadequate interlocutors such as the software developer's sales force or even its development team. The firm must therefore react to this difficult start to the project by giving acute attention to project management activities.

4.2 The analysis of the scenarios

In this section, the three performance measures identified earlier (Productivity, Product quality and Ability to meet schedules) are used to compare and analyze the results of both scenarios.

The behavior of the productivity is strongly influenced by the preventive policy. However, this impact (10.4% reduction of productivity for both scenarios), as presented in figure 7b, may be due to the 5% effort dedicated to quality assurance. This thought is based on the sensitivity analysis, which showed that only quality assurance (of the five process areas) influenced productivity. Hence, using equivalent quality assurance effort for both scenarios might lead to incorrect results.

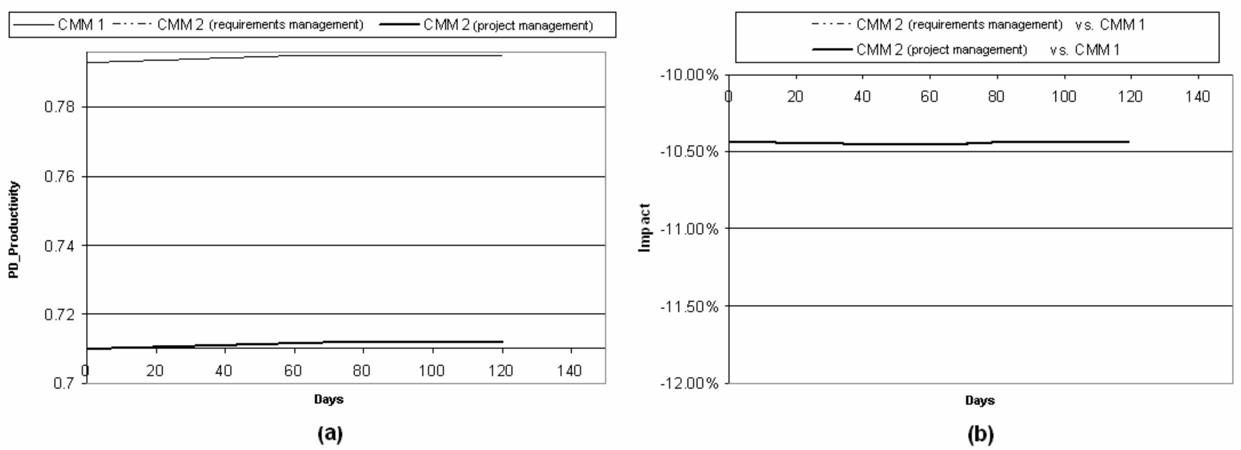


Figure 7 - Results of the variation in the effort assigned to CMM2 process areas on productivity

The influence of CMM2 activities on product quality is more important than on productivity as the quantity of errors diminishes when compared to CMM1 level (figure 8a). As shown in figure 8b, the quantity of errors is reduced by 22% for the prevention policy (requirements management), which is slightly higher than the 15% observed for the control policy (project management). This result confirms the importance of the initial steps of the project where requirements are identified (Sheldon et al., 1992; Demarco, 1995; Linberg, 1999).

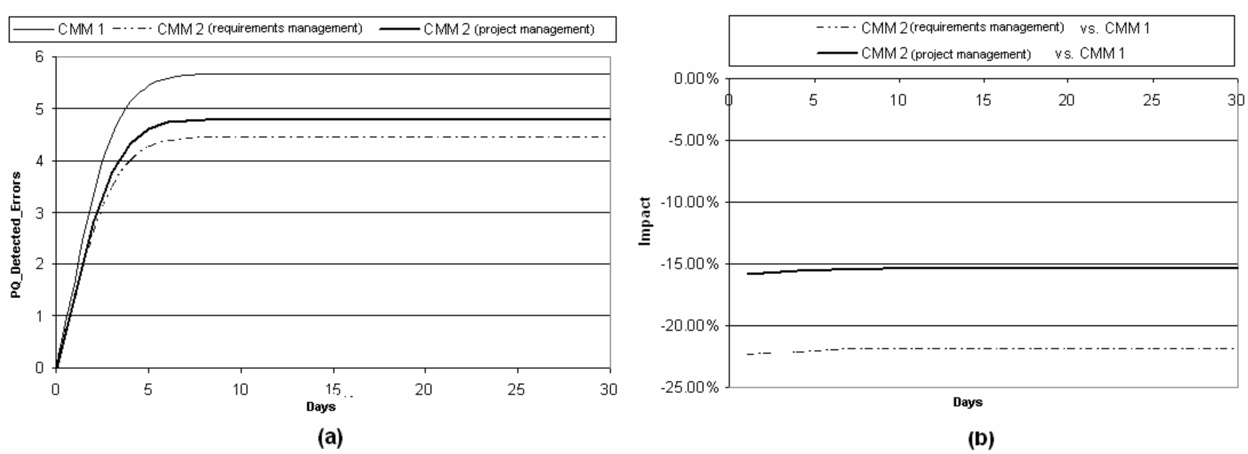


Figure 8 - Results of the variation in the effort assigned to CMM2 process areas on product quality (number of errors)

Finally, the third performance measure, ability to respect deadlines, is greatly influenced by CMM2 activities, but not in a good way (as shown in figure 9). The variation of more than 140%

is unexplainable and incoherent with the literature. As it was the case for the productivity performance measure, this might be caused by the effort assigned to CMM2 quality assurance activities.

These uncertainties (source of productivity and ability to respect of deadline) lead us to add another step to our research process, which is to run another simulation without this CMM2 process area that may disturb the results.

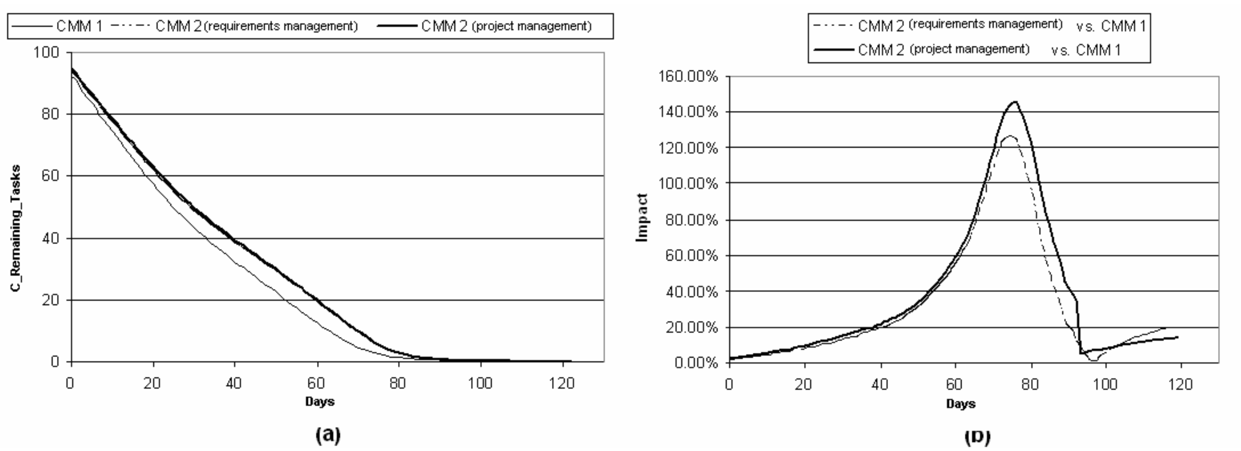


Figure 9 - Results of the variation in the effort assigned to CMM2 process areas on respect of deadlines

4.3 Revised scenarios

For the revised scenarios, the effort assigned to the CMM2 quality assurance (5% of the total project effort) was purposely eliminated, as the effort was equally distributed to the other four process areas (+1,25% of the total project effort each) in order to keep the overall 35% of the total project effort assigned to CMM2 activities. The new specifications for scenarios A' and B' are presented in Table 6.

Table 6 - Revised scenarios for the analysis of CMM2 policies

Key parameters – CMM 2 level	Scenario A'- Revised prevention policy (requirements management)		Scenario B'- Revised control policy (project management)	
	% of total project effort	% of CMM2 effort	% of total project effort	% of CMM2 effort
Percentage_Effort_ Requirements Management	16,25%	46.3%	6,25%	17.9%
Percentage _Effort_ Planning	6,25%	17.9%	11,25%	32.1%
Percentage _Effort_ Monitoring	6,25%	17.9%	11,25%	32.1%
Percentage _Effort_ Configuration Management	6,25%	17.9%	6,25%	17.9%
Percentage _Effort_ Quality_Assurance	0%	0%	0%	0%
Total effort dedicated to CMM2 activities	35%	100%	35%	100%

Using these new parameters for the revised scenarios, there is very little variation of productivity measure when comparing CMM1 to both revised scenarios (figure 10), which is coherent with the sensitivity analysis.

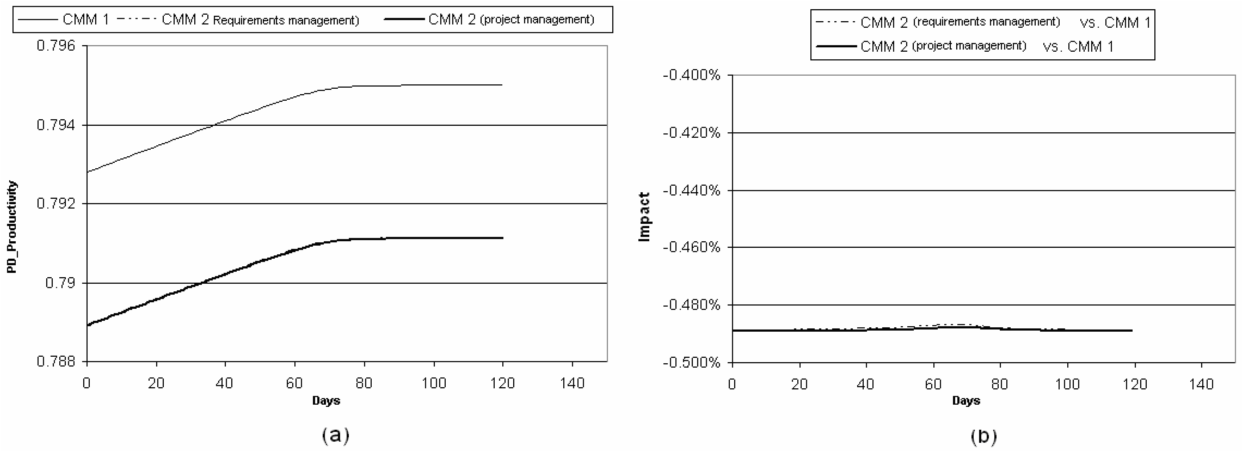


Figure 10 - Results of the variation in the effort assigned to CMM2 process areas on productivity for revised scenarios

The product quality measure is still highly sensible to CMM2 activities for the revised scenarios, although the elimination of the “quality assurance parameter” cut the observed results in half (figure 11a). The gaps between CMM1 and CMM2 are of -13% (of number of errors) for the revised prevention policy (requirements management), and of -6% for the revised control policy (project management) (figure 11b).

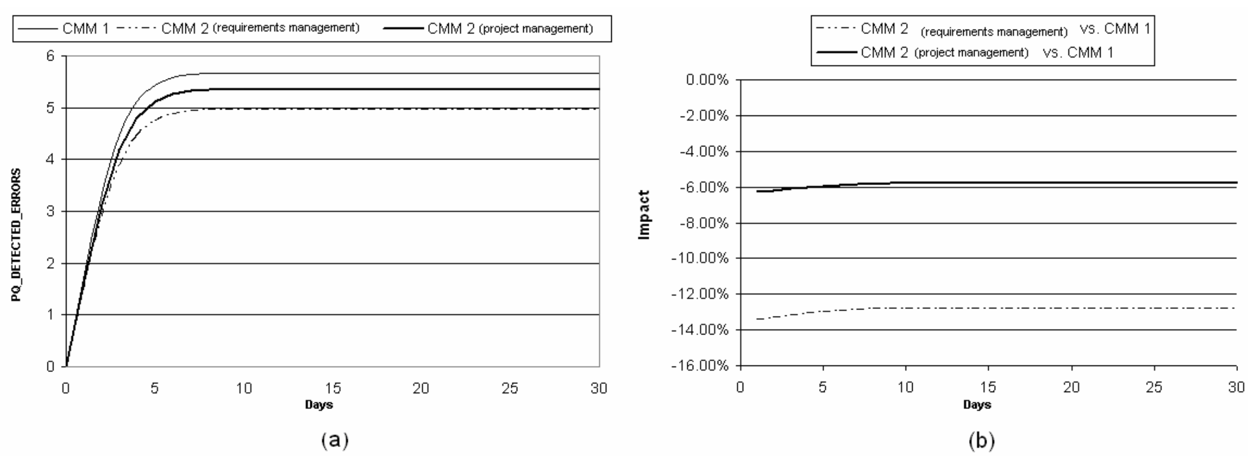


Figure 11 - Results of the variation in the effort assigned to CMM2 process areas on product quality (number of errors) for revised scenarios

The analysis of figure 12, which presents the influence of the revised scenarios on the ability to respect deadlines performance measure, leads to the following observations:

1- In the first part of the project, the efforts assigned to the process areas Planning and Monitoring are responsible for the behavior of the measures in figure 12a. The graph clearly shows that scenario A’ is better the scenario B’ with respect to short term deadlines since it is able to identify more efficiently the remaining tasks at the initial steps of the project.

2- Moreover, in figure 12b, the effect observed at the project's mid-point (60 days in the simulation), which is a significant increase in the remaining tasks' estimation, is due to the efforts assigned to requirements management. At that point, scenario B' observes a lower increase in the estimation of remaining tasks than scenario A', which basically means that scenario B' has more tasks completed than scenario A'. This leads to the inverse and contradictory conclusion that scenario B' is better than scenario A'. Hence, this surprising result claims that project management has a higher impact than requirement management on ability to respect deadlines.

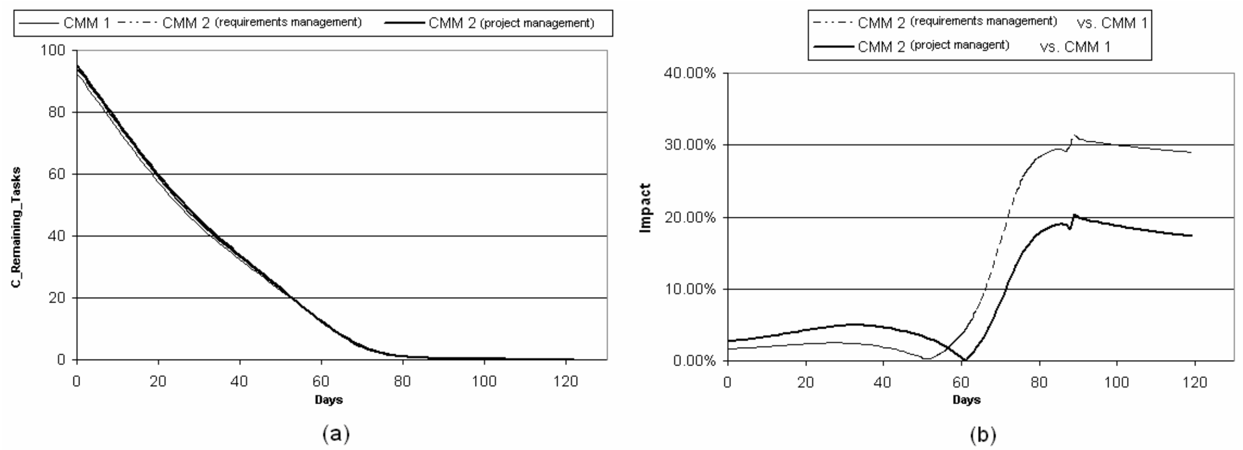


Figure 12 - Results of the variation in the effort assigned to CMM2 process areas on respect of deadlines for revised scenarios

The analysis of the two scenarios indicates that, in general, the adoption of CMM2 practices based on requirements management yields more positive results than policies based on project management. One of the principal conclusions of this research is that the CMM's software quality assurance process area has a sizeable impact on productivity and that all CMM process areas impact scheduling activities. Also, the process areas associated with project management (software project planning and software project tracking) have very little impact on product quality as opposed to the other process areas whose impact is much more substantial on this performance measure.

5. Conclusion

The objective of this research was to better understand the dynamics of software project management in SMEs. More precisely, this study analyzes the performance variations of such projects conducted in SMEs evolving from a CMM1 maturity level to a CMM2 level. Hence, a simulation model was developed to help researchers in software development and management teams in SMEs understand the impact of different management policies and practices according to guidelines of the CMM framework.

The analysis of two typical CMM2 scenarios (prevention and control) clearly shows that intensifying efforts in the early stages of a software project (requirements management) is more interesting than allocating efforts to project management approaches (planning and control) later on in a project.

This research certainly has limits. The fact that the RDM reference model used to build the extended model was considered to be a good representation of a CMM1 software development process is a possible limitation. However, authors such as Paulk et al. (1993) have noticed that a majority of firms in the sector are considered to be at least of CMM level 1, even though they may not have allocated any effort to improve the software development process. The number of experts (4) for the Knowledge Elicitation method and the use of only a single Brazilian SME for different steps of the research are also limits, but both the experts and the firm, involved in the sector for a number of years, represent well the industry.

The extended model is only a first step to modelling the entire CMM through a systems dynamics approach. Apart from using more experts, projects and SMEs to validate the model, other future research avenues come to mind:

- Characterizing the "survival mode" of SMEs and of the pressures that may affect software development policies in these firms. Issues such as outsourcing development activities, different cultures, lack of qualified workforce are just examples of pressures that SMEs keep an eye on.

- Further build and provide in-depth analysis of the assurance quality sector of the model, which mainly impacts productivity and the ability to respect deadlines.

Several contributions also arise from this research. From a theoretical standpoint, the combination of systems dynamic and CMM in an SME context was a first. From a practical standpoint, the extended model that comes out of this project is an interesting decision support system (DSS) that could be useful for training purposes (for managers in the field of software development) and for decision making in general. The application of system dynamics to planning and control problems of project management in software is an intriguing research area. These types of model are useful to help uncover the "hidden" feedback loops that underlie sub-optimal project performance. Documenting and evaluating that process with system dynamics models is a useful mean to provide additional information about "how well" the process is going by enriching the bounded-rationality of decision-makers.

Literature cited

Abdel-Hamid, T. and S. E. Madnick, Eds. Software project dynamics: an integrated approach: Prentice-Hall, Inc.ed. 1991.

Baker, R. The corporate politics of CMM ratings. Communications of the ACM, v.39, n.9, p.105-106. 1996.

Brodman, J. G. and D. L. Johnson. What small business and small organizations say about the CMM: experience report. Proceedings of the 16th international conference on Software engineering: IEEE Computer Society Press, p. 331-340, 1994.

Christie, A. M. Simulation in support of CMM-based process improvement. Journal of Systems and Software, v.46, n.2-3, p.107-112, 1999.

Demarco, T. Why does software cost so much? and other puzzles of the Information Age. New York: Dorset House Pub. 1995. ix, 237 p.

Diaz, M. and J. Sligo. How Software Process Improvement Helped Motorola. IEEE Software, v.14, n.5, p.75-81. 1997.

Donzelli, P. and G. Iazeolla. A dynamic simulator of software processes to test process assumptions. Journal of Systems and Software, v.56, n.1, p.81-90. 2001.

Dybå, T. Improvisation in Small Software Organizations. IEEE Software, v.17, n.5, p.82-87. 2000.

Ford, D. N. and J. Sterman. Expert knowledge elicitation to improve mental and formal models. Cambridge, MA: Sloan School of Management, Massachusetts Institute of Technology, 1997.

Gibbs, W. W. Software 's chronic crisis. Scientific American**:** p. 86-95. 1994.

Goldenson, D. R. and J. Herbsleb. After the appraisal : a systematic survey of process improvement, its benefits, and factors that influence success. Pittsburgh, Pa.: Carnegie Mellon University Software Engineering Institute. 1995. 60 p.

Herbsleb, J., D. Zubrow, D. Goldenson, W. Hayes and M. Paulk. Software quality and the Capability Maturity Model. Communications of the ACM, v.40, n.6, p.30-40. 1997.

Humphrey, W. S. Modeling implications of the personal software process. Proceedings of the 5th international software process workshop on Experience with software process models: IEEE Computer Society Press, p.74-77.1990.

Humphrey, W. S., T. R. Snyder and R. R. Willis. Software Process Improvement at Hughes Aircraft. IEEE Software, v.8, n.4, p.11-23. 1991.

Kahen, G., M. M. Lehman, J. F. Ramil and P. Wernick. System dynamics modelling of software evolution processes for policy investigation: Approach and example. Journal of Systems and Software, v.59, n.3, p.271-281. 2001.

Kamsties, E., K. Hörmann and M. Schlich. Requirements Engineering in Small and Medium Enterprises. Requirements Engineering, v.3, n.2, p.84-90. 1998.

Keil, M. and D. Robey. Blowing the whistle on troubled software projects. Communications of the ACM, v.44, n.4, p.87-93. 2001.

Kellner, M. I., R. J. Madachy and D. M. Raffo. Software process simulation modeling: Why? What? How? Journal of Systems and Software, v.46, n.2-3, p.91-105. 1999.

Kelly, D. P. and B. Culleton. Process Improvement for Small Organizations. Computer, v.32, n.10, p.41-47. 1999.

Leffingwell, D. and D. Widrig. Managing software requirements: a unified approach. Reading, Mass.: Addison-Wesley. 2000. 491 p. (The Addison-Wesley object technology series)

Linberg, K. R. Software developer perceptions about software project failure: a case study. Journal of Systems and Software, v.49, n.2-3, p.177-192. 1999.

Nikula, U., J. Sajaniemi and H. Kälviäinen. A State-of-the-Practice Survey on Requirements Engineering in Small and Medium Sized Enterprises. Lappeenranta University of Technology. Lappeenranta, Finland. 2000. (TBRC Research Report 1)

Otoya, S. and N. Cerpa. An Experience: A Small Software Company Attempting to Improve its Process. Software Technology and Engineering Practice. Pittsburgh, Pennsylvania, 1999.

Paulk, M. C., B. Curtis, M. B. Chrissis and et al. Capability Maturity Model for Software. Software Engineering Institute, Carnegie Mellon University. Pittsburgh: August 1991. (CMU/SEI-91-TR-24)

Paulk, M. C., B. Curtis, M. B. Chrissis and C. V. Weber. Capability Maturity Model for Software, Version 1.1. Software Engineering Institute, Carnegie Mellon University. Pittsburgh. 1993. (CMU/SEI-93-TR-24)

Pfahl, D. and K. Lebsanft. Integration of system dynamics modeling with descriptive process modeling and goal-oriented measurement. Journal of Systems and Software, v.46, n.2-3, p.135-150. 1999.

Ruiz, M., I. Ramos and M. Toro. A simplified model of software project dynamics. Journal of Systems and Software, v.59, n.3, p.299-309. 2001.

Rus, I., J. Collofello and P. Lakey. Software process simulation for reliability management. Journal of Systems and Software, v.46, n.2-3, p.173-182. 1999.

Scott, R., T. Garner and D. Ticoll. A Fine Balance : The Impact of Offshore IT Services on Canada’s IT Landscape. PricewaterhouseCoopers, p.42. 2004

Sheldon, F. T., K. M. Kavi, R. C. Tausworthe, J. T. Yu, R. Brettschneider and W. W. Everett. Reliability measurement: from theory to practice. IEEE Software, v.9, n.4, p.13-20. 1992.

Standish Group, I. CHAOS: A Recipe for Success, www.standishgroup.com. 1998.

Sterman, J. Business dynamics : systems thinking and modeling for a complex world. Boston: Irwin/McGraw-Hill, 2000

Villalón, J. A. C.-M., G. C. Agustín, T. S. F. Gilabert, A. D. A. Seco, L. G. Sánchez and M. P. Cota. Experiences in the Application of Software Process Improvement in SMES. Software Quality Control, v.10, n.3, p.261-273. 2002.

Wangenheim, C. G. V., T. Punter and A. Anacleto. Software Measurement for Small and Medium Enterprises. Fraunhofer IESE. October 2003.

Wohlwend, H. and S. Rosenbaum. Software improvements in an international company. Proceedings of the 15th international conference on Software Engineering. Los Alamitos, CA, p.212-220. 1993.