An Experience: A Small Software Company Attempting to Improve its Process

Sergio Otoya
Winapp Pty Ltd
8 Berry St North Sydney NSW 2060, Australia
sergio@winapp.com.au

Narciso Cerpa
School of Information Systems
University of New South Wales, Sydney 2052, Australia.
n.cerpa@unsw.edu.au

Abstract

The Capability Maturity Model[1] for software (CMM^{SM}) has been used by many organizations as a framework for improving the software development process. However, small organizations have encountered difficulties applying the CMM, since some of its key practices are inappropriate to their software projects. Thus, LOGOS[2] international has derived a tailored version of the CMM for small businesses, organizations, and projects.

This study presents the practices of a small organization (Winapp) attempting to improve its software process. The improvement practices were implemented on a needs basis, and business priorities, without using any proven framework. In this paper we compare the practices before and after the improvements based on developers' perceptions of team performance. The results from this study suggest that Winapp has obtained substantial gains from its process improvement, since it has improved the perceived quality of the product delivered for projects of larger size, with greater complexity and constraints.

1. Introduction

The Capability Maturity Model (CMM) developed by the Software Engineering Institute provides a theoretical framework for the software engineering process improvement [1]. This model consists of five levels of maturity, and the key practices that organizations must implement to achieve each level. The maturity levels are: Initial; Repeatable; Defined; Managed; and Optimizing. Organizations at the initial level are assumed to have a chaotic software engineering process and they would need to implement some key management practices: Requirement Management; Software Project Planning; Software Project Tracking and Oversight; Software Subcontract Management; Software Quality Assurance; and Software Configuration Management. These management practices help to put in place a basic disciplined process with the aim of obtaining a repeatable process.

Improved process maturity results in an increased productivity, better quality and more accurate schedule time [2]. However, it is difficult for most organizations to even achieve the first step of the ladder (Level 2), since there is not a practical implementation strategy for the key practices within each level [3]. There is a need for determining an implementation strategy for the key practices within each level according to the support they provide each other [4].

On the other hand, small software organizations, and small team projects may find it difficult to achieve high levels of maturity according to the CMM, since many of the key practices suggested by this model are inappropriate to small businesses and projects [5]. Therefore, the CMM has been tailored or generalized (LOGOS Tailored CMM) to apply it to any size business, organization or project [6]. Small project teams cannot cope with the overheads produced by the amount of documentation required by the CMM and they must use combined documents to reduce time. In small projects, teams usually have a flat structure, resulting in developers being assigned several roles due to scarce resources. This contrasts with the team structure and positions suggested by the CMM practices and makes the implementation of some practices difficult. The tailored version has solved these issues

[1] Capability Maturity Model and CMM^{SM} are service marks of Carnegie Mellon University.

[2] The LOGOS Tailored Version of the CMM for Small Businesses, Small Organizations, and Small Projects is copyrighted as a derivative work: copyright © 1995 by LOGOS International, Inc

among others, making it easier for small software organizations to implement this improvement framework.

In this paper we present the experience of a small service company that is attempting to improve its software process. This improvement until now has been driven by organizational needs, and implemented according to business priorities. However, we present it here within the tailored CMM framework, since this will be used as the basis for the implementation of further improvements. The following sections describe the organization, the needs for improvement, the improvement framework, and the achievements.

2. About Winapp

Winapp is a small service company based in Sydney, Australia whose areas of business are software development, consulting and developer training. Most of Winapp's software developments are based on Microsoft products such as Visual Basic, Access and SQL Server[3]. Winapp serves companies who seek the productivity gains of a PC based "client server" environment mainly in the areas of finance, banking, law, home security, electricity wholesale, and education among others.

2.1. Before Beginning Improvement (1993-1996)

Winapp started with one developer back in 1993 working from a home office and has grown to five employees since then. Initially, the lack of an appropriate operational infrastructure like a central file server provided developers with limited access to resources like e-mail and Internet facilities, restricting inter-client and -developer communication. At this time, there was no clear definition of the business direction, which impeded the growth of the organization.

2.2. After Beginning Improvement (1996-present)

Since then, Winapp has developed a business plan to identify its vision, mission and goals to cope with this problem. This plan includes an analysis of Winapp's strengths, weaknesses, opportunities and threats (SWOT). Winapp has developed strategies to achieve financial, human resources, marketing, operations, management and sales objectives. All of these objectives are related and support the principal aim of increasing the development and sales of quality software. In order to achieve this aim, Winapp realizes it must continue improving its software development process.

Winapp is also improving its infrastructure. In the last two years the company has moved to an office in a central business area and has implemented a domain and a central file server with permanent Internet connectivity that allows staff easy access to e-mail, newsgroups and other internet related resources. In addition, Winapp has obtained accreditation from Microsoft as a Certified Solution Provider.

In the next sections we will provide a more detailed description of the needs that Winapp had for process improvement, the improvement framework used and an analysis of the process improvement gains.

3. Needs for process improvement

As mentioned above, Winapp was formed in 1993 and began developing simple customized software for small clients. These development efforts were undertaken by a team made up of 1 or 2 developers handling very few of these projects concurrently. There was not much need for documentation and tight project management control due to the software application nature and size of the development team and project.

As Winapp grew in number of clients and the size and complexity of their projects, there was a need for larger development teams and subsequently a more controlled software development process was required. This was due to an increased number of projects being late, the difficulty in keeping track of project requirements and developer resources. The problems encountered during this stage were various and could be categorized as follows according to the CMM:

3.1. Requirements Management

The lack of requirements documentation and client approval made it difficult to keep track of project requirements to build a suitable system for the client on time. Indeed, project requirements were recorded on hand written sheets of paper and sometimes misplaced. For most of these projects the approval was a verbal 'Gentleman's agreement'. Thus, the client was able to include new requirements at various development stages, which in turn meant that the project would be delayed or undesired shortcuts would be taken during implementation.

The lack of written requirements made it difficult to communicate programming tasks to developers and resulted in several iterations of a system prototype. The lack of a proper requirement analysis meant that programmers would receive a set of sometimes-incomplete hand written requirements and were left to work out the details by themselves. The developer

[3] Microsoft Visual Basic, Access and SQL server are trademarks of Microsoft Corporation.

would then produce staged prototypes of the software for display to the client and at each iteration the client would clarify requirements and even introduce new ones. This ‘code and fix’ process without having written requirements, change management and prototype iteration control, was time consuming and usually introduced software faults.

3.2. Project Planning and Tracking

Regarding project planning, there was not a written plan, but a very rough estimate of planned time to completion was given to the client which usually resulted in the project being late. Programming tasks and their priorities were communicated to programmers either verbally or in an email message, which sometimes resulted in misunderstanding of tasks between the programmer and the planner. Sometimes it was difficult to reorganize task priorities and keep track of what had already been done or what was still outstanding due to the lack of a written project plan.

For slightly larger projects, there was no means of easily tracking how many hours were spent on any given task. Timesheets in the form of spreadsheets were recorded but it was very awkward to compile a project status report to measure actual versus planned. This resulted in clients complaining that they did not get enough feedback as to how their project was tracking. This lack of proper project management meant that Winapp could not attempt tendering for larger projects, since the constraints of time, quality and cost would have been difficult to manage.

Other project administrative tasks like billing clients and paying subcontractors took too much extra time at the end of the month. This sometimes resulted in either working long hours to get these done or the projects being late due to the lack of an accurate project planning and tracking process.

3.3. Subcontract Management

Initially Winapp conducted software projects by employing individual subcontractors. The lack of an appropriate policy for subcontractor management and the primitive project management approach mentioned above resulted in a less than adequate control of the subcontractor’s work. The use of subcontractors also generated problems such as the lack of loyalty to the company. Subcontractors often lack motivation and contribute ‘just enough’ to the project. While this approach would be sufficient to achieve software quality when subcontract management policies and procedures are in place, this would be detrimental when there is a lack of them.

3.4. Software Quality Assurance

One of the most difficult issues for the implementation of software quality assurance (SQA) in small organizations like Winapp, is the lack of independence of the SQA function due to the flat organizational structure common to this type of companies. In addition, Winapp did not have any written policies or procedures in place to assure the quality of the software. Perhaps the only informal procedures to ensure quality were cursory unit and system testing. Cursory unit testing was conducted on each module by the developer as a random check. Once modules were ‘successfully’ tested, system testing was conducted by the project leader to ensure the interaction among modules, functions and subsystems also in a random manner. After system testing had been finalized, the software was released to the client for acceptance testing. This resulted in a large number of client phone calls requesting support to repair software defects not found during the different informal testing layers.

Since there was not a written test plan, it was usually difficult to reproduce a software defect, making it harder for the programmer to repair bugs reported by the client. When the defect was eventually repaired, the problem and solution were not recorded at all. Thus, subsequent similar problems were investigated from ‘scratch’ as opposed to relying on past experiences.

Another important issue was the lack of standards for software design and development. A very informal design was conducted by the project leader for database structures and by each developer for user interfaces, which resulted in varied design qualities depending on the developer’s experience. During coding, developers used to write programs with little or no naming standards and lack of internal documentation. Therefore, maintenance of this code was difficult and time consuming. If a code routine was built in a generic form, it was not reused subsequently due to the developer’s ignorance of its existence or the lack of documentation. This resulted in duplicated effort due to ‘Reinventing the wheel’. In addition, error checking and handling was not always done in a standard style, which resulted in a variety of outcomes for different functions within the same software application and between systems. All of these issues contributed to delivering lower quality software.

3.5. Software Configuration Management

Two of the most important functions of SCM such as version control and change management, help developers to identify, control and retrieve software products such as development tools, operating systems and the code modules written by the development team. Winapp had only partially implemented version control, which caused several problems to developers. For example, versions of software were kept in compressed files and sometimes it was difficult to go back to a particular software version. Another issue causing problems to developers was the lack of a standard directory structure, which made moving a whole project from one developer machine to another very difficult. It usually took a few hours before this move could be done successfully, and in some instances it was easier for developers to swap machines. The lack of a standard directory structure also resulted in the misplacement of critical software development documentation. Therefore, it was fundamental for Winapp to implement an appropriate version control system in a central file server with the aim of improving the identification, control and retrieval of software items in a timely and accurate manner.

In terms of software installation, the developer would create a set of diskettes with all the required files for the installation. The project manager would subsequently use these floppies for the on-site manual installation process by copying the contents onto the client's hard disk. This process created problems since the software installation was not previously tested under a similar target platform.

3.6. Training

Although the CMM does not include training as a key process area at level 2, Winapp required that each developer be up to date with current technologies and releases of development tools. However, Winapp did not provide any training program, and it was left to each individual to research in his/her own time to upgrade their skills.

4. Improvement framework

Winapp implemented an improvement framework based on the needs described in the previous section. While this improvement framework is presented and analyzed according to the CMM, it was not necessarily developed and implemented according to it. Winapp attempted an implementation of those key processes that were fundamental for improving software quality, control and company growth. This improvement framework was lead by the Winapp directors and supported by the software development team.

4.1. Requirements Management

Requirements management has been partially improved. Winapp has tried to standardize and facilitate the requirements elicitation phase. To achieve this, the company has created a requirements analysis template with standard sections and pre written text. In this document Winapp has introduced the usage of 'shall' statements as a means for defining requirements. The project manager writes shall statements and further breaks them down into sub-requirements. Further analysis of the sub-requirements allows the project manager to estimate the time assigned to each of them, and these estimates are used to calculate the total project duration.

Whereas previously Winapp only had verbal agreement with the client over the requirements, currently the client is required to sign off the requirements document before commencing any work on the project. This accepted requirements document is subsequently used for controlling the introduction of new requirements at any time and for the acceptance of the final product by the client. Different versions of the requirements document are kept in a version control system and each of these versions includes the requirements changes made to the previous one. Once the requirements document is approved by the client, the project manager enters the requirements and estimates once again into an internally developed timesheet system (WinTimesheet) for project control as described in the next section.

4.2. Project Planning and Tracking

Project planning has been improved since requirements and estimates are recorded in the requirements document and in WinTimesheet by the project manager. Each developer records in WinTimesheet the actual time spent on individual tasks. This system allows us to:

- Keep track of actual versus estimated.
- Produce status reports at any time, classifying the information by project, period and/or employee.
- Produce billing sheets and invoices
- Keep track of employee utilization rates and performance.

More accurate estimates are being produced, since they are based on actual times from previous similar projects recorded in the WinTimesheet database.

Each of the tasks involved for completing a sub-requirement is recorded onto Microsoft Outlook[4] for each client and project. In addition, other issues are recorded under different standard categories such as Information (INF), Enhancements (ENH) or Bugs. The use of MS Outlook gives Winapp the ability to easily categorize and sort records. Thus, it facilitates the tracking of tasks and issues to completion.

[4] Microsoft Outlook is a trademark of Microsoft Corporation.

The project manager conducts status report meetings with clients on a weekly or fortnightly basis depending on the size and complexity of the project. The use of WinTimesheet allows Winapp to generate status reports with very little effort and to recognize activities or tasks that are running late or that have a good chance of being late.

The client and staff administration burden has been greatly relieved by the hiring of a secretary who manages the billing and payroll tasks.

4.3. Subcontract / Employee Management

Currently Winapp only employs developers as full time staff. This policy was decided during our business plan development to build up loyalty and company identity. To fulfil this aim, Winapp has introduced a performance review system where the directors set each employee's goals and objectives and align them with the company's goals and objectives. At the end of each period the employees are financially rewarded for the attainment of their goals during the review period. Winapp is also attempting to build team spirit and loyalty by having training sessions at a holiday resort and company paid dinners for the staff.

As mentioned in the previous section and as a way of managing employee's times, Winapp is currently using WinTimesheet to keep track of the number of hours worked by the staff on specific project sub-requirements.

4.4. Software Quality Assurance

Winapp has improved its testing procedures by making use of a template to produce system test plans. The system test plan test cases are developed by the project manager at a high level and are further refined by the developers. On the other hand, module testing has not been standardized yet and its quality still depends on each developer's experience. The test cases in the system test plan are aligned with each of the functional requirements. This system test plan is also used as an instrument for acceptance testing by the client.

Developers conduct preliminary iterations of the system test plan and record the defects into a Microsoft Outlook public folder for the project. These defects are then reviewed and must be repaired by a developer before continuing on to the next iteration. The project manager conducts the final iteration of the system test plan before releasing the product for acceptance testing by the client. Clients sign off the system at the end of the acceptance testing and get 60 days warranty from there on.

Winapp is currently developing design and coding standards. Although they are not fully developed yet, these partially documented standards for coding and interface design have been enforced through peer reviews. These reviews are conducted by a fellow developer with the aim of assuring an improved software quality. For example, a code review consists of checking for compliance with coding standards and appropriateness of algorithms. This code review is conducted after the module level testing but before the system test plan. Requirements and design reviews are conducted by a fellow project manager at the end of each respective phase. All of these reviews can also be conducted on an event driven basis.

The analysis, design and development process have been aided by the usage of tools such as CRC cards, UML sequence diagrams, UML use cases, entity relationship diagrams, data flows and process diagrams. The usage of these tools has helped the company to standardize the development process and improve productivity.

A tools library has been created by Winapp and its usage has been promoted to maximize common routines reuse. This library is currently being reorganized into logical groups and help files, which will be published in the company's Intranet. In addition, an internal Visual Basic add-in library to insert routines into modules has been created to allow the standardization of routine headers, description and error checking. Winapp has also designed and implemented a class hierarchy for 'Import Engine', a generic system that imports files from one format into any ODBC[5] compliant database. This system has already been reused for different clients and it will continue to be used in future development projects. These libraries have improved productivity and software quality since all of these components have been fully tested in their first releases.

All Winapp's developers have achieved Microsoft Visual Basic and Access professional status, which has helped the company obtain accreditation as Microsoft Solution Provider.

4.5. Software Configuration Management

Winapp is currently utilizing Microsoft Source Safe[6] version control and change management software in a central file server, to manage different versions of source and binary files as well as project documentation. A standard directory and version structure has been implemented, which facilitates the 'checking out' of files by developers onto their own development machines. This standard directory structure also allows easier movement of projects from one development machine to another. This software

[5] Microsoft ODBC is a trademark of Microsoft Corporation.
6 Microsoft Source Safe is a trademark of Microsoft Corporation.

facilitates the retrieval of previously stored versions of a software product. Microsoft Source Safe implements change management control by storing the changes made from one version to the other for a work product.

In terms of software installation, Winapp currently builds a "ghost"[7] file with the client's target platform. This file is subsequently used to rebuild "clean" target platforms to test the pre built setup kits. This process has had a big impact in reducing the number of configuration faults during the system installation at the client's site.

4.6. Training

Winapp has implemented a knowledge-sharing program among the staff. Each staff member is responsible for choosing a development topic of his/her own interest, researching it and then giving a 30-minute talk and demonstration to the rest of the group. Each employee is also sent on 1 or 2 training courses a year to constantly update their skills. In addition, senior staff provide coaching to less-experienced employees.

5. What has Winapp gained?

One way of measuring what Winapp has gained since the introduction of the improvement framework at Winapp, would be to compare the performance of developers on projects, before and after this framework. To achieve this we must first describe the criteria to be used in the comparison, analyze the improvement produced by the framework and state future developments at Winapp. All these will be discussed below.

5.1. Criteria

The comparison of the practices before and after the improvement is based on the team performance on projects as perceived by two Winapp developers who were part of the development team from the beginning (1993) until now. The criteria for team performance are categorized as project characteristics, quality of project management and planning, quality of the development process, and quality of the product delivered. These perceived criteria are described as follows:

- project characteristics include size, complexity, constraints and cost.
- quality of project management and planning is measured by the level of project planning and control undertaken on employees/sub-contractors during the life of the project.
- quality of the development process is measured by the degree of requirements analysis, design, coding and testing performed during the project.
- quality of the product delivered is measured according to the frequency of requirements, design, logical, interface, configuration faults, and user complaints.

The developers have given their perceptions according to a Likert scale ranging from 1 to 5, where 1 is the lowest and 5 the highest. For example, "4" means a fairly large project size, or a high frequency of faults when it refers to interface errors. In the case of project cost, Table 1 shows the transformation of project cost to the Likert scale.

Scale	Range (AU$)
1	0-10K
2	10-50K
3	50-100K
4	100-300K
5	300K+

Table 1. Transformation of project cost to Likert scale.

5.2. Data Analysis

The developer's perceptions were based on eight projects before and thirteen after the improvement. The averages of these perceptions are shown in Table 2.

An analysis of the developer's perceptions shows that the size, complexity, constraint and cost of the projects after the improvement were on average higher than before. This suggests that these projects needed an improved project management and development process to be able to cope with them. Indeed, the perceptions on project management and planning, and quality of development process support this. Project planning is perceived to have a 66% increase while employee/subcontract management improved by 52%. This is not a surprise since Winapp started to produce written project plans and to use the WinTimesheet database to keep track of the project schedule and compare estimated versus actuals. This facilitated project planning, scheduling and tracking as well as employee management.

The quality of the development process shows a definite improvement according to the developer's perceptions. Table 2 illustrates that requirements analysis has improved by 157%, design by 128%, and coding and testing by 152%. This was expected, since Winapp introduced the use of a requirements analysis document template, which facilitated the requirements gathering, and the requirements reviews by a peer project manager. Design, coding and testing were also

7 Ghost is a trademark of Symantec

improved due to the usage of partially documented standards, design tools to facilitate and automate routine tasks, system and acceptance test plan document templates and peer code reviews. However, perhaps one of the most important improvements in the development process is the move to development and usage of standard reusable software components such as a class hierarchy for ‘Import Engine’, a ‘tools’ library and add-in libraries to Visual Basic. The needs for reliability and reusability at Winapp make the utilization of off-the-shelf and the in-house development of components essential to achieve software quality. Winapp has also considered developing Internet-based applications where a component-based approach is crucial for successful software development.

As a consequence of the improvement in project management and planning, and in the quality of the development process, the quality of the product delivered has also improved. An analysis of the developer’s perception shows that there are now fewer faults than before in terms of requirements, design, logical, interface and configuration errors as well as user complaints. The requirements faults have on average decreased by 57%, while the design, logical, interface and configuration faults by about 35%. The reduction in configuration faults is surely supported by the fact that Winapp is currently using “ghost” files to build target client platforms. In addition, user complaints have also been reduced by 33%. It has to be noted that there was a definite decrease in requirements faults since Winapp started using the requirements document template and implemented requirements reviews.

On the other hand, there are also some benefits, which are difficult to quantify but are important for achieving Winapp’s goals. We refer here to staff morale, client and staff communication and rapport, and company image.

Categories	Before	After
Project Characteristics		
Size	2.88	3.38
Complexity	2.88	3.77
Constraints	1.63	2.69
Cost	2.25	2.38
Project Management and Planning		
Employee/Subcontractor	2.25	3.42
Project planning	1.63	2.69
Quality of Development Process		
Requirements analysis	1.38	3.54
Design	1.25	2.85
Coding and testing	1.75	4.42
Quality of Product Delivered		
Requirements errors	3.50	1.50
Design errors	2.75	1.69
Logical errors	2.75	1.83
Interface errors	2.75	1.77
Configuration errors	2.38	1.50
User Complaints	2.75	1.85

Table 2. Averages of perceptions before and after the improvement.

5.3. What does Winapp have to do now?

In the future, Winapp intends to invert the requirement documentation process by initially entering the requirements and sub-requirements into WinTimesheet, and automatically generating the requirements document to be approved by the client. Winapp also plans to extend WinTimesheet to facilitate the retrieval and comparison of estimated versus actual with the aim of assisting the estimation of new projects. The company also expects to make standard the utilization of an appropriate project planning tool such as Microsoft Project[8] to plan and control projects. In addition, Winapp will introduce post project/phase reviews (depending on the project duration) that will provide us with feedback on the software development process with the aim of improving it.

Winapp is also planning to set policies for SQA and implement them through procedures. These policies should aim to achieve an independent SQA function despite the flat structure due to the small size of the organization. Some objectives are: to complete and publish in the Intranet the written design and coding standards; to standardize reviews through checklists at each phase of development. In addition, Winapp is currently designing a knowledge base that will contain software faults and their fixes with the aim of aiding novice developers by increasing their productivity as well as the quality of their work.

For software configuration management, Winapp plans to: set written policies; create the role of software configuration manager; collect statistics from MS Source Safe to measure the number of change requests per unit of time within other measurements.

Since Winapp has decided to only employ full time staff, the company will not give importance to the implementation of key practices of the subcontract management key process area as suggested by the CMM.

6. Conclusions

The aims of this study were to measure the gains at Winapp attributable to process improvements and

[8] Microsoft Project is a trademark of Microsoft Corporation.

determine what to do next. Although the improvement framework was driven by organizational requirements and implemented on a need basis, it was here analyzed within the CMM framework with the aim of classifying the level of improvement.

Results from this study suggest that although the company seems to be still at level 1 according to the CMM or its tailored version, the gains from the current improvement process are a big step forward and an encouragement to continue improving. It is important to highlight that by initially improving project planning and management, the rest of the key process areas showed an improvement as well.

Another important outcome from this study was the knowledge gained by Winapp management on an improvement framework (CMM). Winapp's management has seen the need for having a defined software development process, which could provide the means for its own improvement, with the aim of supporting business growth without compromising quality. After this study, Winapp is considering evaluating a proven improvement framework such as the LOGOS tailored CMM to meet this need.

A comparison of Winapp's current practices and the key practices required by the CMM at level 2, highlights those weak areas in the software development process that Winapp must immediately address before expecting any further improvement. Some of the issues are:

- fully document all non-written or partially written policies, procedures and standards
- study developers roles and responsibilities to comply with the tailored CMM requirements (e.g. role of software configuration manager)
- implement policies and procedures for SQA ensuring the independence of this function according to the constraints presented by the size and flat structure of the organization and development teams
- fully implement the key practices currently in place according to the tailored CMM

Therefore, the next step in improving Winapp's process will be to set up a practical strategy for achieving the level 2 according to the LOGOS tailored CMM. We believe that this study has given a different direction to Winapp's process improvement approach, and has made its management aware of the potential advantages of using a proven framework.

7. References:

[1] Paulk, M. C., Curtis, B., Chrissis, M. B., and Weber, C. V., 1993. *Capability Maturity Model For Software*, Version 1.1. Pittsburgh, PA: Software Engineering Institute.

[2] Brodman, J. G., and Johnson, D. I., 1996. Return on investement from software process improvement as measured by U.S. industry. *Crosstalk*, 9(4).

[3] Johnson, D. I., and Brodman, J. G., 1996. Realities and rewards of software process improvement. *IEEE Software*, 29(11):99-101.

[4] Bilotta, J. G., and McGrew, J. F., 1998. A Guttman Scaling of CMM Level 2 Practices: Investigating the Implementation Sequences Underlying Software Engineering Maturity. *Empirical Software Engineering*, 3:159-177.

[5] Brodman, J. G., and Johnson, D. I., 1994. What Small Business and Small Organizations Say About the CMM. *In Proceedings of the 16th International Conference on Software Engineering*, Sorrento Italy.

[6] Johnson, D. I., and Brodman, J. G., 1997. Tailoring the CMM for small business, small organisations, and small projects. *Software Process Newsletter*, 8: 1-6.