Measuring the requirements management key process area

Annabella Loconsole

Abstract

The purpose of this paper is to provide software measurements for implementation of the goals of the Requirements Management Key Process Area (KPA) of the Capability Maturity Model (CMM), and to ease the adoption of the KPA. The paper also provides practical guidance for people trying to implement the Requirements Management KPA. The CMM, developed by the Software Engineering Institute (SEI) is not well supported by software measurement and it is somewhat complex. An application of the Goal/Question/Metric (GQM) paradigm to the Requirements Management KPA is therefore presented. The metrics obtained will help immature companies to satisfy the goals of the Requirements Management KPA.

1. Introduction

Software processes are considered to be the main area for quality improvement. There are two main streams within Software Process Improvement (SPI) [17]. One is based on assessments of organisations' capability, e.g. Capability Maturity Model for Software (SW-CMM) [12], Software Process Improvement Capability dEtermination (SPICE) [6], BOOTSTRAP [11], and the ISO9000 family. The other is based on measurements of software practices within an organisation, e.g. Goal/Question/Metric (GQM) [1], Quality Improvement Paradigm (QIP) [2], and Application of Metrics in Industry (AMI) [14]. These approaches complement each other because software measurement is inherent to the concept of improvement, but they are seldom applied together [17]. The assessment-based approaches should always include measurements, because it is necessary to estimate the state of the software process before action is taken to improve it and compare it with the state thereafter. The SW-CMM developed by the Software Engineering Institute is intended to help software organisations to improve the maturity of their software processes, but is weakly supported by a measurement-based approach. Therefore this paper proposes a set of measures for the Requirements Management Key Process Area (KPA) of the SW-CMM. This work has been done with the aim of joining the assessment and measurement based methodologies mentioned above. The underlying assumption in the paper is that it is easier to focus the measurement and improvement activities on a specific process area, rather than trying to measure and improve all the process areas at once. This is especially true for small-medium size enterprises, which do not have enough resources to train people on complex frameworks like the SW-CMM. The Requirements Management KPA has been chosen because it is important to control the continuing definition of requirements as they change throughout the software life cycle. Such control over the requirements helps in anticipating and responding to requests of change [16]. The aim has been addressed by analysing the Requirements Management KPA of the SW-CMM and its key practices [13], and applying the GQM paradigm to it.

The first contribution of this paper is to give a general and comprehensive set of software measures for the implementation of the goals of the Requirements Management KPA within the SW-CMM (see [7] for a definition of measure and measurement). This set of measures constitutes a "pick list" that can be tailored to the specific enterprise, offering small to medium size enterprises the freedom to choose suitable subsets of software measures. Another contribution of this paper is a simple method of improving the Requirements

Management activity and the presentation of basic software measures, which are easy to understand and to use. The paper presents a practical approach and a practical guidance for people trying to fulfil the goals of the KPA. The method can also be used to evaluate the state of the Requirements Management activity.

At the time of writing, the measures are being tested at Ericsson Erisoft AB in Umeå, Sweden, which is a medium size company. The results of this empirical study will help the author to demonstrate that a joint approach is more complete than an assessment or measurement based approach. The measures obtained can be used in quantifying the amount of changes to requirements and to predict the cost for a change, helping to control requirements and changes to requirements.

The measures will be grouped by maturity level. For instance an immature enterprise can start measuring the total number of requirements and the number of changes to requirements throughout the software life cycle. This would provide better control and visibility into the Requirements Management activity, improving the software process a small step towards the goal of having a repeatable process. For each measure proposed, improvement actions will be suggested to help decision makers in their job. After these actions are taken, an enterprise can follow the approach described in "Application of Metrics in Industry" [14], which suggests to cycle four steps (analyse, act, metricate, and improve) to continue with the improvement of the KPA.

The remainder of this paper is organised as follows: the section 1.1 describes related research in software measurement for the SW-CMM, sections 2 and 3 present an overview of the CMM and the GQM respectively, the application of the GQM to the CMM is described in section 4, section 5 contains initial results of the experiment started in the before-mentioned company, and concluding remarks and future directions are presented in section 6.

1.1. Related Work

Several studies on software measurement for the SW-CMM have been done prior to this work. One of the most relevant has been done by Baumert and McWhinney [3]. Their report provides a set of indicators (composite measures) that are compatible with the measurement practices of the SW-CMM. The indicators are categorised according to the quality attributes they fulfil and not by KPAs. Raynus [15] confirms the connection between software measurement and the SW-CMM. His work examines relationships between measurable process quantities, and reviews the SW-CMM demonstrating that software measurement can be used to improve the behaviour of a software development organisation. His book represents a quantitative approach to software management and SPI. He suggests some metrics for the SW-CMM and their use. In goal oriented software improvement, the measurements must be focused on specific goals, for example specific process areas, while the authors mentioned above, suggest indicators across all the maturity levels. Therefore, this paper presents a comprehensive collection of measures that are focused on a specific KPA namely the Requirements Management KPA of the SW-CMM.

2. The Requirements Management KPA of the Capability Maturity Model

As shown in figure 1, the CMM is composed of 5 distinct levels (Initial, Repeatable, Defined, Managed, Optimising) [12].

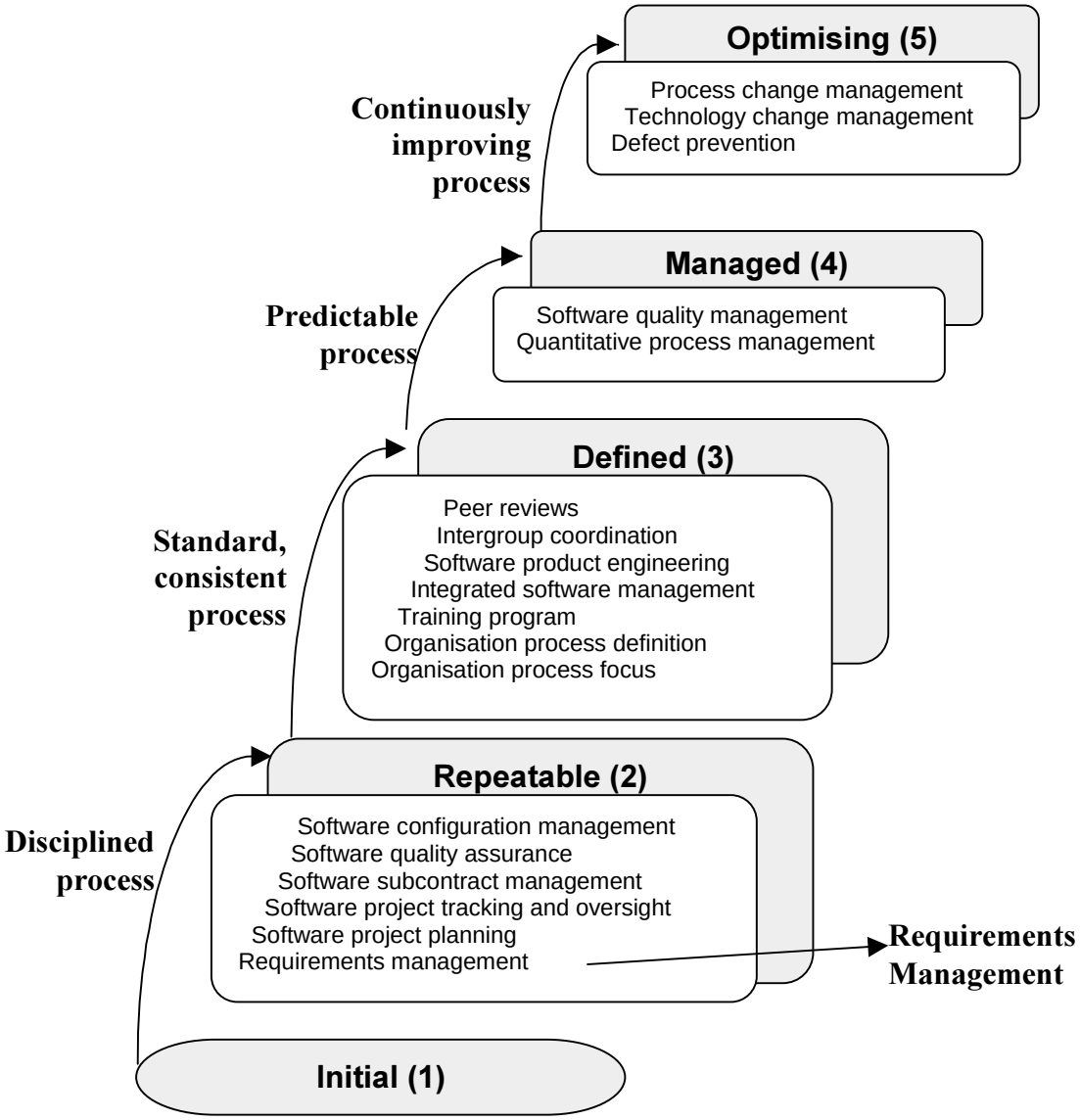

Figure 1: The Key Process Areas by Maturity Level [12].

Each CMM level, except the initial level, has several Key Process Areas (KPA). [12]. One level 2 KPA is Requirements Management. "*The purpose of Requirements Management is to establish a common understanding between the customer and the software project of the customer's requirements that will be addressed by the software project*" as defined in [13]. This means that the requirements of a software project should be complete, documented, unambiguous, controlled, etc., in order to design a software product, which satisfies the customer's needs. Very often, requirements change through the software development life cycle but the control of the change requests is poor. The activity of "Requirements Management" is focused on the control of the requirements gathering, establishing an agreement between the customer and the software team on the requirements, checking, reviewing and managing the changes on requirements. This activity is the process of ensuring that a software product produced from a set of requirements, will meet those requirements.

3. The Goal/Question/Metric Paradigm

The Goal/Question/Metric (GQM) paradigm is a method for helping an organisation to focus the measurement program on their goals. It states that an organisation should have specific goals in mind before data are collected [1]. GQM does not specify concrete goals. It is rather a structure for defining goals and refining them into a set of quantifiable questions These questions imply a specific set of metrics and data to be collected in order to achieve these goals. The GQM paradigm consists of three steps:

Purpose:
Analyse some
(objects: processes, products, other experience models)
for the purpose of
(why: characterisation, evaluation, prediction, motivation, improvement)
Perspective:
with respect to
(focus: cost, correctness, defect removal, changes, reliability, user friendliness,...)
from the point of view of
(who: user, customer, manager, developer, corporation,...)
Environment:
in the following context
(problem factors, people factors, resource factors, process factors,...)

Figure 2: Goal Template.

1. Specify a set of goals based on the needs of the organisation and its projects. Determine what should be improved or learned. The process of goal definition is supported by templates like the ones shown in figure 2 [1]. By using these templates it is possible to define the goals in terms of purpose, perspective, and environment. The identification of subgoals, entities, and attributes related to the subgoals is made in this step.
2. Generate a set of quantifiable questions. Business goals are translated into operational statements with a measurement focus. Basili and Rombach [1] provide different sets of guidelines to classify questions as product-related or process-related. The same questions can be defined to support data interpretation of multiple goals.
3. Define a set of metrics that provides the quantitative information needed to answer the quantifiable questions. In this step, the metrics suitable to provide information to answer the questions are identified and related to each question. Generally, each metric can supply information to answer several questions and sometimes a combination of metrics is needed to make up the answer of a question.

Once these steps are identified, data are collected and interpreted to produce an answer to the quantifiable questions defined to fulfil the goals of the organisation [1], [17].

4. Application of the Goal/Question/Metrics to the CMM

The CMM and the GQM can very easily be intertwined. The CMM defines one or more goals for each KPA as shown in figure 3. These goals can be used for the first step of the GQM. The CMM defines two distinct goals for the Requirements Management Key Process Area (KPA). The first ones states the following:

"System requirements allocated to software are controlled to establish a baseline for software engineering and management use" [13].

It focuses on the control of requirements to set up a baseline, that is a kind of standard by which things are measured or compared. If the requirements are not controlled, there will be no clear picture of the final product, because the final product is based on the requirements. The second goal of the Requirements Management KPA states the following:

"Software plans, products and activities are kept consistent with the system requirements allocated to software" [13].

The main focus of this goal is the consistency between the requirements and any software product created from those requirements. This consistency would result in the design of the product required by the customer. The goals presented can be redefined by applying the goal template in figure 2 as follows. To *Analyse* the system requirements allocated to software *for the purpose of* establishing a baseline *with respect to* the control of the requirements *from the point of view of* academy and the software manager*, in the context of* the company where the Requirements Management is implemented. The second goal can be redefined as follows: to *Analyse* software plans, work products and activities *for the purpose of* consistency with the system requirements allocated to software *from the point of view* of academy and the software manager, *in the context of* the company where the Requirements Management is implemented.

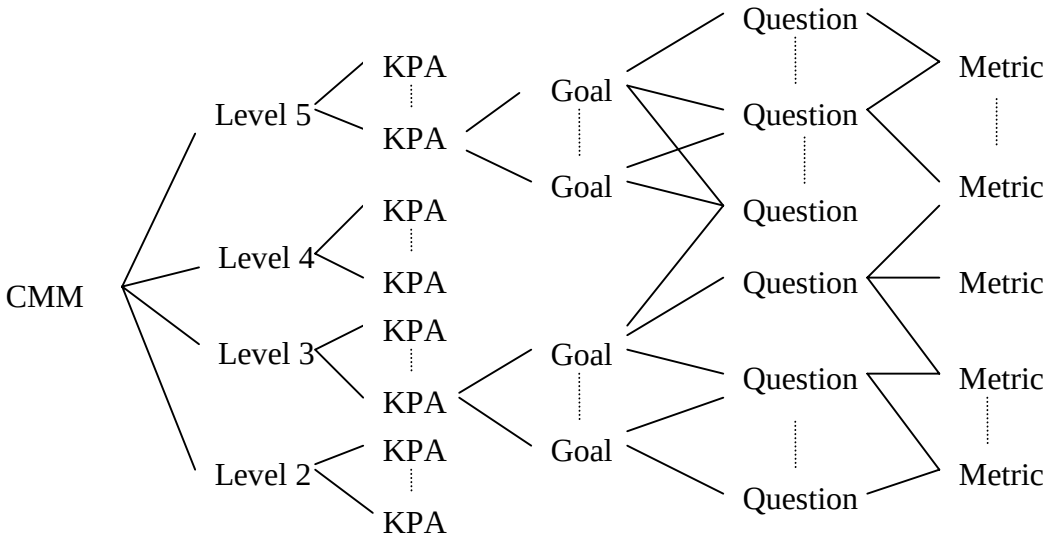

Figure 3: Relationship between CMM and GQM

The second step in the GQM paradigm is to generate a set of quantifiable questions. The questions have been produced by applying the guidelines for process related questions [1], analysing the goals of the Requirements Management KPA and the its Key Practices [13] word by word, reading papers [8], [3], discussing with colleagues, and browsing web pages. For some questions a rationale will be given to better understand the meaning and/or the utility of a certain question.

4.1. Questions for the First Goal of the Requirements Management KPA

By analysing the first goal, a question arisen is: how can the requirements be controlled? And why should we control them? We know that it is not possible to specify in the beginning exactly what the customer wants. Neither it is possible to dictate the frequency or desirability of changes. The changes can come at the worst moment and impede our ability to finish a

project with the available resources. The only possibility is to control the continuing definition of requirements as they change throughout the life cycle [16].

Table 1: Questions and measures for the first goal of the Requirements Management KPA.

Questions	Measures
1 What is the current status of each requirement?	• Status of each requirement
2 What is the level of the stability of the requirements?	• # initial requirements • # final requirements • # changes per requirement
3 Why are the requirements changed?	• # initial requirements • # final requirements • # changes per requirement • # test cases per requirement • Type of change to requirements • Reason of change to requirements • Major source of request for a change to requirements • Phase where change was requested
4 What is the cost of changing the requirements?	• Cost of change to requirements • Size of a change to requirements
5 Is the number of changes to requirements manageable?	• Total # Requirements • # changes to requirements proposed • # changes to requirements open • # changes to requirements approved • # changes to requirements incorporated into base line • # changes to requirements rejected • The computer software configuration item(s) (CSCI) affected by a change to requirements • Major source of request for a change to requirements • Requirement type for each change to requirements • # requirements affected by a change
6 Does the number of changes to requirements decrease with time?	• # changes to requirements per unit of time
7 How are affected groups and individuals informed about the changes?	• Notification of Changes (NOC) shall be documented and distributed as a key communication document • # affected groups and individuals informed about NOC
8 How many other requirements are affected by a require ment change?	• # requirements affected by a change
9 In what way are the other requirements affected by a requirement change?	• Type of change to requirements • Reason of change to requirements • Phase where change was requested
10 Is the size of the requirements manageable?	• Size of requirements
11 How many incomplete, inconsistent and missing allocated requirements are identified?	• # incomplete requirements • # inconsistent requirements • # missing requirements
12 Does the number of “To Be Done” (TBD) decrease with time?	• # TBDs in requirements specifications • # TBDs per unit of time
13 How are the requirements defined and documented?	• Kind of documentation
14 Are the requirements scheduled for implementation into a particular release actually addressed as planned?	• # requirements scheduled for each software build or release
15 How many requirements are included in the baseline?	• # baselined requirements • phase when requirements are baselined

Any information on requirements can help to establish control. Especially important is to know the starting set and final set of requirements. To increase the control of the

requirements, their status as well as their stability could be investigated (see questions 1 and 2 in table 1). The possible status of a requirement could be: new, analysed, approved, documented, rejected, incorporated into the baseline, designed, implemented, tested, etc. Requirements stability is concerned with the changes made in requirements, therefore a set of questions (see questions 3-9 in table 1) about requirements changes can be defined to refine the question 2. The level of requirements stability can be measured also by having information about the size of the requirements and by identifying problematic requirements (see questions 10-12 in table 1).

Once there is control over the requirements, a baseline must be established. Therefore some questions about how the requirements are documented, and how many of them are included in the baseline, are defined (see questions 13-15 in table 1).

Table 1 shows all the questions defined and the measures proposed to give information to answer questions. Please observe that some of these questions are also used for the second goal of the Requirements Management KPA, for instance, questions 13 and 14.

4.2. Questions for the Second Goal of the Requirements Management KPA

The purpose of the second goal is mainly to keep consistency between the requirements and the software project, therefore it is suggested to keep traceability among the software documents. Traceability between requirements and the software project facilitates the analysis of the effects of a software change and reduces the effort to locate the causes of a product failure. Tracking the requirements and changes made to requirements can help to maintain traceability (questions 1-7) among the requirement documents.

When the changes to requirements have been implemented, other documents are most probably affected by this change. Therefore it is suggested to check the status of software plans, work products, and activities, which could be "identified, evaluated, assessed, documented, planned, communicated to affected groups and individual, and tracked to completion".

Table 2: Questions and measures for the second goal of the Requirements Management KPA.

Questions	Measures
1 Does the software product satisfy the requirements?	• # initial requirements • # final requirements • # test cases per requirement • Type of change to requirements
2 What is the impact of the changes to requirements on the software project?	• Effort expended on Requirements Management activity • Time spent in upgrading • # documents affected by a change
3 What is the status of the changes to software plans, work products, and activities?	• Status of software plans, work products, and activities
4 Are the requirements scheduled for implementation into a particular release actually addressed as planned?	• # requirements scheduled for each software build or release
5 How are the requirements defined and documented?	• Kind of documentation
6 Does the number of TBDs prevent satisfactory completion of the product?	• # TBDs in requirements specifications
7 Are all development work products consistent with the requirements?	• # inconsistencies

5. Measures for the Goals of the Requirements Management KPA

The third step of the GQM is to define sets of metrics that provide the quantitative information necessary to answer the questions. In this paper, only a set of measures is presented, which is shown in tables 1 and 2. There are overlaps among the questions for the two goals and among the measures. The same measure can be used to give information to answer different questions. Some of the measures are suggested by the SW-CMM (status of allocated requirements, change activity, and cumulative number of changes to allocated requirements). Other possible measures are the number of test cases assigned to each requirement, by which it is possible to check how many requirements are verifiable; the size and the cost of a change request which could make it possible to predict the project cost and the schedule. Other measures on requirements are the ones recommended by [9] who measure attributes of requirements like ambiguity, modifiability, priority, etc. However, these are measures on the requirements and not on the requirements management process therefore these are out of the scope of this paper.

Once the three steps of the Goal Question Metric paradigm are defined, an organisation has to determine ranges for "good data" (for instance a company could accept no more then three change requests per week). Data must be collected and compared to the "good data", and eventually improvement actions are taken. A comparison of the data for the actual project with the once collected for previous projects will provide a baseline for the requirements and give meaning to the measures.

6. Testing the measures in a company

At the time of writing, the measures are being tested at Ericsson Erisoft AB in Umeå, Sweden, which is a medium-size company. People at the company used SW-CMM and improved their software process to level 2 but caused by a re-organisation they decreased their CMM maturity level. A previous study made in this company [4] shows that people at the company apply some basic measurements but they do not perform any significant evaluation of the data collected.

The test of the measures is conducted following the guidelines in [18] about how to conduct an experiment; the first step of the experiment process (definition of the experiment) has been done.

Object: Requirements Management activity at Ericsson Erisoft AB in Umeå, Sweden
Purpose: Evaluate the impact of the measures to improve the Requirements Management activity
Quality focus: Control of the requirements
Perspective: Academy
Context: Medium-size company

While planning the study (which is the second step of the experiment process suggested in [18]), the author is learning the company's organisational structure, and the software processes used in the company. The software processes used are PROPS as management process and RUP (Rational Unified Process) [10] as software development process. The author is also going to map the terminology used by people in the company to the CMM terminology. Some data from studying one particular increment have been collected (a project can have many increments and each increment is a refinement of the software product). A preliminary conclusion of this study is that the domain analysis is not done very deeply and the requirements are added and changed during the software life cycle.

7. Concluding Remarks and Future Directions

An application of the GQM to the Requirements Management KPA has been reported. The result of the application of the GQM to the CMM, is a list of measures for each KPA.

The set of questions and measures presented should be tailored to the particular organisation. All level 1 companies interested in improving the Requirements Management activity are suggested to select a subset of these measures especially useful to start with. For instance, a level 1 organisation has most probably poorly defined requirements. The visibility of the process is very low at this level, and it is difficult to measure the process. Therefore, it is suggested that they count the number of requirements and changes to those requirements to establish a baseline. Other level 1 companies (like the ones referred in [5]) need to document the requirements before starting to measure. A company like Ericsson-Erisoft in Umeå (who has improved the process through the CMM in the past), can collect all the data regarding the change activity. This is possible because information about change requests is stored in reports and database.

The measures produced in this paper provide the organisation with improved visibility and better insight into the Requirements Management activity, improving the software process a small step towards the goal of having a repeatable process. The measures can be used in quantifying the amount of changes to requirements and to predict the cost for a change, helping to control requirements and changes to requirements. If the process is repeatable, more information on requirements can and should be collected, such as type of each requirement (database requirement, interface requirement, performance requirement, etc.) and change requests to each type. In general, the metrics collection will vary with the maturity of the process.

This work has been done with the aim of joining the assessment and measurement based methodologies mentioned above. The aim has been addressed by analysing the Requirements Management KPA of the SW-CMM and its key practices [13], and applying the GQM paradigm to it. The underlying assumption in the paper is that it is easier to focus the measurement and improvement activities on a specific process area, rather than trying to measure and improve all the process areas at once. The results are relevant to areas such as Requirements Management, Software Measurement, Software Process Improvement, Software Quality, etc.

The measures obtained, will be used for the elicitation of requirements information in the before-mentioned company. For each measure proposed, improvement actions will be suggested to help decision makers in their job. After these actions are taken, an enterprise can follow the approach described in "Application of Metrics in Industry" [14], which suggests to cycle four steps (analyse, act, metricate, and improve) to continue with the improvement of the KPA. Based on the data collected, suggestions for improvement of the Requirements Management activity will be given. The measurements will be automated as far as possible because personnel treat measurement as boring. Finally, application of the GQM to all the KPAs of the CMM levels will be considered.

8. Acknowledgements

Special thanks to my supervisor Jürgen Börstler for his suggestions and contribution to this paper and to all the people who have supported and contributed to the writing of this work.

9. References

[1] Basili, V.R. and Rombach, H.D., "The TAME project: Towards improvement-oriented software environments", in IEEE Transactions on Software Engineering 14(6), 1988, pp.758-773.

[2] Basili, V.R., Caldiera, C., and Rombach, H.D., "Experience Factory", Encyclopædia of Software Engineering Volume 1, (Marciniak, J.J., editor), John Wiley & Sons, 1994, pp. 469-476.

[3] Baumert, J.H. and McWhinney, M.S., "Software Measures and the Capability Maturity Model", Software Engineering Institute Technical Report, CMU/SEI-92-TR-25, ESC-TR-92-0, 1992.
[4] Bergström, C., "Process Metrics for Ericsson Erisoft AB- a proposal", Umeå University Report Umnad 292/2000, Umeå, Jan. 2000.
[5] Cline, A., "Overcoming Cultural barriers to the adoption of Object technology", in OOPSLA´97 conference proceeding, Atlanta, Oct. 1997.
[6] El Eman, K., Drouin, J.N., and Melo, W., "SPICE The Theory and Practice of Software Process Improvement and Capability Determination", IEEE Computer Society Press, Los Alamitos, California, 1997.
[7] Fenton, N.E. and Pfleeger, S.L., "Software Metrics - A Rigorous & Practical Approach", 2nd Edition, International Thomson Publishing, Boston, MA, 1996.
[8] Goodbrand, A.D. and Wang, Q., "Software Measurement Plan for the Requirements Management Key Process Area of the Capability Maturity Model for SENG623 Inc.", March 1997, http://www.cpsc.ucalgary.ca/~alang/SENG623/, (12 Feb. 2001).
[9] Hammer, T.F., Huffman, L.L. and Rosenberg, L.H., "Doing Requirements Right the First Time", CrossTalk, Dec 1998, pp. 20-25.
[10] Kruchten, P., "The Rational Unified Process- An Introduction", Addison Wesley Longman Ltd, Reading Massachusetts, May 1999.
[11] Kuvaja P. and Bicego, A., "BOOTSTRAP - Europe's Assessment Method", IEEE Software, May 1993, pp. 83-85.
[12] Paulk, M.C., Curtis, B., Chrissis, M.B. and Weber, C.V., "Capability Maturity Model for Software Version 1.1", Software Engineering Institute Technical Report, CMU/SEI-93-TR-24, ESC-TR-93-177, Pittsburgh, PA, 1993.
[13] Paulk, M. C., Weber, C.V., Garcia, S., Chrissis, M.B. and Bush, M., "Key Practices of the Capability Maturity Model Version 1.1", Software Engineering Institute Technical Report, CMU/SEI-93-TR-25 ESC-TR-93-178, Pittsburgh, PA, 1993.
[14] Pulford, K., Kunntzmann-Combelles, A., and Shirlaw, S., "The Ami Handbook: A Quantitative Approach to Software Management", Addison Wesley Longman Ltd. UK, 1996.
[15] Raynus, J., "Software Process Improvement with CMM", Artech House Publishers, Boston, 1999.
[16] Reifer, D.J., "Requirements Management: The search for Nirvana", IEEE Software, May/June 2000, pp. 45-47.
[17] van Solingen, R., and Berghout, E., "The Goal/Question/Metric Method - A practical Guide for Quality Improvement of Software development", McGRAW-Hill Companies, London, 1999.
[18] Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B. and Wesslen, A., "Experimentation in Software Engineering An Introduction", Kluwer Academic Publishers, Boston/Dordrecht/London, 2000.