Dynamic CMM for Small Organizations

A.Laryd, T.Orci
Umeå University, Department of Computer Science[1]
S-901 87 Umeå, Sweden
astrid.laryd@swipnet.se, terttu@dsv.su.se

Abstract

Software CMM has gained wide acceptance in software process improvement, especially in USA, but recently also at the European market. When the chaos is a fact in a software development organization, it is surely wise to initiate a software process improvement program. That is, however, a difficult task to undertake in addition to the normal duties in a chaotic situation. This suggests that it would be wiser to apply the process improvement principles already from the very start of the organization, at the time when they are not yet really needed. To this end, there is a need for models for small organizations. The existing models for software process improvement, e.g. CMM, are an overkill for several reasons, and moreover difficult to understand and comprehend for the management of small organizations. This paper presents a model called Dynamic CMM for small organizations, applicable from the very start and supporting organizational growth, thereby covering all the way from the start until the organization is ready to shift to the original Software CMM.

Keywords: SW-CMM, software process improvement, small organizations.

1 Introduction

Software CMM has gained wide acceptance in software process improvement, especially in USA, but recently also at the European market. Software CMM, called plain CMM in the sequel, has been described in literature (Paulk et al. 1995), and is also publicly available at the SEI web pages (http://www.sei.cmu.edu). The gains of using CMM in software process improvement have also been published by the SEI in a

[1] The work has been financially supported by NUTEK, the Swedish National Board for Industrial and Technical Development.

study concerning a number of large US companies. Implementing CMM in an organization is a difficult task, as are the most improvement efforts, requiring another way of thinking, and other tasks to be performed by the staff than they are used to. Today, information about implementing CMM is also publicly available, e.g. (Caputo 1998), as well as about other improvement activities, e.g. (Grady 1993), (Zahran 1997), including guidelines and pitfalls, metrics, forms, and templates.

No doubt, a software development organization with notably symptoms of chaos caused by immaturity should start a software process improvement program. Such a program may adopt CMM, some other model publicly available, or it can be approached using a local model developed for the purpose. CMM, for example, presents a number of maturity levels stepwise and to enough detail, thereby offering a roadmap, which is based on sound management practices, and is historically proven to lead to success with high probability. In terms of an improvement ladder, the model introduces more and more engineering and management practices. Intuitively, it is a comfortable way of executing an improvement program. Thereby we do not argue that CMM as a road map is the best one to obtain improvement.

It is common knowledge that the majority of the organizations world-wide are on CMM Level 1, characterized by reactive behaviour and chaos. When a chaos is already a fact, it is certainly wise to start an improvement program. It would, however, be even wiser to start an improvement program before the chaos is a reality. In fact, the wisest thing to do would be to start an improvement program when starting the operation of the organization. Obviously, we would in that case not think about an improvement, but see it as a sound business and engineering strategy.

The next best thing is to start an improvement program while an organization is a very small one, with a few software engineers, and with a simple management. In the start-up phase, with a few software engineers, with one or a few versions of one product or only one project running at a time, the communication is simple, configuration management is relatively easy, everybody knows what the colleague engineer is doing, neither the development work nor the management offers bigger problems.

However, if the organization is successful, it will grow. After some time there will be many more software engineers, one manager does not manage it all, but there will be managers specialized to certain tasks, with an organizational hierarchy in place. There will be more versions of the first product, but also new products or services, new application areas, the technology growth forces new tools and methods to be used - the list can be made much longer. At some point in time things will get complicated. It is no longer easy to know what the colleague engineer is doing. It is not easy to know which version of which compiler was used to generate a particular version of a particular product to a particular customer. The management looses oversight unless provided by visible evidence of the progress in development projects. This picture is the reality for the majority of all the companies world-wide.

1.1 The Problem

In order to avoid the picture painted above, an organization should have the focus on software processes before actually needed, i.e. more or less from its start. The existing models, e.g. CMM, are not, however, directly applicable for small organizations. CMM, for example, proposes more than 25 organizational roles, with various tasks and responsibilities. In a small organization, there is not enough people to fill the roles proposed, neither is there any need of many of those roles. Further, the models are usually described on a huge number of text pages, being cumbersone and time consuming to get oversight and comprehend. Therefore, the models need to be scaled down to the needs and possibilities of small companies. At the same time, it is important that such a model is applicable continuously in the course of time, while the organization grows, otherwise it will cease to be useful after a while.

1.2 The Objectives and Scope

The objectives of the research, partially presented in this paper, is to develop software process improvement models for small organizations, usable from the very start, dynamic in the sense of supporting growth, and easy to get oversight. In this paper, the first step to such a model, called Dynamic CMM, is presented. The reasons for selecting CMM as the basis are that it is well-known, well described, and publicly available.

The model is intended to be applicable from the start of the business until the number of employees exceeds 50, which is the limit for small organizations according to the European Community classification. By then, the original CMM should be an adequate tool for further improvement. If an organization, already before reaching the medium size organization status, would qualify as Level 2 organization, it is assumed that the maturity and the practise of Dynamic CMM provide the skills needed to apply the original CMM without bigger difficulties.

The evolution from one person organization to an organization with 50 employees may be extremely fast, e.g. as is currently the case in organizations providing Internet solutions and services, or it may be more moderate, occurring over a few years. Most probably, the pace in the extremely fast growing companies dictates different kind of models for process improvement than in those evolving in more moderate pace. To study extremely fast growing organizations is certainly important, but outside the scope of this paper.

1.3 Methodology

In the development of the Dynamic CMM for small organizations, a revolutionary approach of experimental paradigm has been used. Such an approach starts with proposing a new model, followed by application of the model in case studies, measuring, analysing, and validating the model.

The development of the model has been based on common sense, CMM knowledge, industrial experience and theoretical knowledge of software engineering in general, and process improvement in particular.

The model is intended to be used not only by software process improvement (SPI) professionals, but also by organization's internal staff, with some experience of SPI and by the management. The basic idea is that the model users can easily get an overview of the model, of the magnitude of the effort required and of the roles, responsibilities, tasks, and documents. With those intentions in mind, the original CMM has been interpreted, reduced, reorganized, and partially represented in a graphical notation.

The objective concerning *usability* for small organizations has implied a focus on the roles, and to decide which of them are applicable and realistic in small organizations, even in case the organization only consists of one person. The objective concerning *dynamics* of the model has implied a subclassification of small organizations into three subclasses, along the axes of number of employees and number of projects or products/product versions. CMM presents the model along KPAs organized by the common features. A model user seeking easy oversight does not find there, but looks for roles, responsibilities, and documents. Therefore, the objective of *oversight and simplicity* has implied studies of other forms for representation than the original CMM, which presents the model along KPAs but organized by the roles and their associated responsibilities and activities, and documents by type.

1.4 Outline

In Section 2, the basic idea and architecture of the original CMM are briefly described. Section 3 presents the roles in the models XXS, XS, and S, and for space reasons, only one Key Process Area (KPA) in more detail. The complete model is presented in (Laryd and Orci 1999). Section 4 discusses aspects of application of the model, and Section 5 contains concluding remarks and future research.

2 The Basics of CMM

CMM is a road map to software process improvement. Its architecture forms a ladder with an initial level and four steps. The initial level, Level 1, is characterized by reactive behaviour and chaos in the software development environment. The success of a software project is determined by heroic efforts of project members. At the other extreme, the level 5 called Optimizing, is characterized by mature software processes and organizational culture, with continuous improvement as a fundamental part of the culture. Level 2 is called Repeatable, implying that a success can be repeated in similar projects. The basic project management practices are in place, and the projects work towards scheduled milestones. Different projects can, however, work very differently. At Level 3, there is an organizational common process in place, to be tailored to the individual projects in a controlled way. Measurement is part of the practices used in

the organization. Level 4 Managed is characterized by management by measurement, i.e. the measurements of both process and product are used to control the projects.

It is said to take about two years for an organization to climb to Level 2 Repeatable from the chaos. How exactly the climbing is done, and what problems may be encountered, are described in literature, e.g. in (Caputo 1998). The number of steps and requirements to be fulfilled may seem overhelming, especially when taking the first step to Level 2. A new culture, promoting quality and organized work, is introduced. The resistance from the participating people is often highest at the first step.

At every level except Level 1, a number of Key Process Areas (KPAs) describe prioritized areas of improvement work. Every KPA includes a number of Goals to be fulfilled, by means of Common Features: Commitment to Perform, Ability to Perform, Activities Performed, Measurement and Analysis, and Verifying Implementation. The common feature Activities Performed describes the activities which are assumed to be performed in order to implement the KPA in question in the organization. The other common features assure that the practices are institutionalized in the organization.

A key idea of CMM is that the work should not be ad hoc, but conform to certain procedures, which have been defined, documented, and agreed upon, in the organization. The procedures must be documented. Otherwise there is an obvious risk that the procedures change in time - if for nothing else, then for individual variations in peoples preferences and the way of work. Written organizational policies are also an essential part of CMM.

This paper is restricted to Level 2 Repeatable. The KPAs at Level 2 are: Requirements Management (RM), Software project Planning (SPP), Software project Tracking and Oversight (SPTO), Software Subcontract Management (SSM), Software Quality Assurance (SQA), and Software Configuration Management (SCM).

3 Models for Small Organizations

A small organization has 1-50 employees. That range represents a wide variety of organizations. The employees adding to the count are assumed to be either development staff, management or marketing and sales, excluding administrative services and human resources staff. We believe that special conditions apply to an organization with one or two employees, because it is easy to be continuously informed of what the collegue is doing. Such organizations are here called eXtra eXtra Small organizations (XXS). As soon as the organization grows to include three persons, the loss of oversight and easy communication is at risk. Yet, an organization with three employees is very much smaller and easier to manage than an organization with nearly 50 employees with several products, several versions of each product or customers at a time. It seems adequate to further classify the organizations to eXtra Small (XS) with 3-15 employees and Small (S) with 16-50 employees.

One of the reasons for the need of a down-scaled model for small organizations is the large number of roles with responsibilities proposed in CMM. The basic assumption we make is that the responsibilities are essential, while the roles are only carriers

of responsibilities, and therefore, the responsibilities of a role may be taken over by a person working in another role without affecting the original intentions of CMM.

CMM Level 2 proposes totally 25 roles with some responsibilities. Some of the role names rather imply groups. However, some groups might very well consist of only one person, e.g. Documentation Support Group (DSG), while others, e.g. Software Configuration Management (SCM) involves a Software Configuration Control Board consisting of more than one person.

We have studied the roles, their interdependencies, responsibilities and activities. There is only one role, Software Quality Assurance group (SQA), that should not involve any person from the management, neither should it include persons involved in the software development. The reasons are obvious: quality assurance without impartiality does not fulfill its purpose.

The roles may be one of three types. First, a role may be important enough to be included as a role, assigned to a person with associated responsibilities, e.g. Senior Management (SM) as every organization must have such a role unregarded the size. Second, a role may be important in terms of activities and responsibilities, but without need for a formal role. In such a case, the activities and responsibilities can be carried out by a person having some other role, with the consequence that the responsibilities and activities of that person will be more extensive than implied by his formal role. An example of such a role is Documentation Support Group (DSG) which is irrealistic in a one person company, but still there is some documentation to do, assumed to be taken over by software engineers (SE). Third, some roles are irrelevant, e.g. an organization not applying subcontracting does not need Software Subcontract Management (SSM). The roles with significant responsibility and applicable to small organizations are presented in Table 1.

In this section, the three models for small organizations will be presented. The models include, in addition to the roles, the KPAs with goals, and common features. One of the intentions of the models presented in this paper is simplicity and easy oversight, by emphasizing the major activities and responsibilities of the roles rather than all the details. The models include graphical overview of the KPAs and their activities. For space reasons, only one KPA is shown in this paper. The complete model is found in (Laryd and Orci 1999). The models are intended to be applicable for organizations developing software products to open market, as well as for organizations developing customer specific solutions. In case there is a difference in roles or otherwise between these organizations, it is indicated in the text.

3.1 eXtra eXtra Small Organizations

An XXS organization is assumed to have one or two employees and one version of the first product is under way. Alternatively, the first customer project has been initiated. This is the starting scenario for many software development organizations. If there is only one employee, the person is both the manager in the role Senior Manager (SM) and a Software Engineer (SE). If there are two employees, the one is both SM and SE, the other only SE. Still, two persons have a good chance to get insight into each

others work, and we see no reason to have formally assigned project management. By that, we do not claim that project management is abandoned, project planning and tracking are important activities. The implication is only that project management activities can be carried out informally by SM. SoftWare Manager (SWM) is responsible for the software development environment, but also for the operative software and hardware in the organization. In companies developing customer specific solutions, SWM becomes the expert on computers and adjustments at the customer site. Still, a person possessing one of those roles may very well work also in another role.

We make the assumption that subcontracting is not realistic in a two-person organization, implying that Software Subcontracting Management KPA is irrelevant. System Test Group (STG) is supposed to be responsible for system tests, while SE has the responsibility of tests during the development. Validation and verification is of primary importance to obtain a quality product. Still, an internal STG role is hardly realistic in a two person company. For organizations with customer specific development, the customer is the most adequate group for acceptance tests. In product development organizations in the beginning of the business, an external test service can be used.

SQA should not be shared with persons involved in development activities or in the management. In a two person organization, it is not realistic to appoint an internal SQA. In an organization with specific customers, Customer SQA (CSQA) can be used to conduct SQA activities, if SQA is available at the customer site. In product development organizations, an external SQA service can be bought.

In CMM, there is Hardware Engineering Group, developing hardware components. Further, System Engineering Group (SG) is responsible for the systems with both software and hardware. We ignore Hardware Engineering Group and to the extent their services are needed, they are assumed to be handled by SG. The systems aspects are of course important, both for the product developing organizations as the software may include both software and hardware, but also from the point of view of providing customer specific solutions including hardware as a part of the delivery. The roles in an XXS organization are presented in Figure 1.

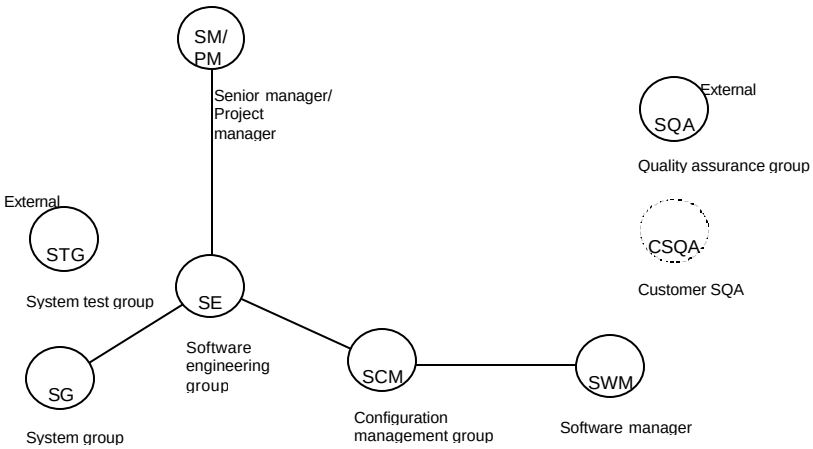

Fig. 1. The roles in an XXS organization

In the models, a circle denotes a role. A dotted line around a circle means that the role is relevent only in organization developing customer specific solutions. The links between the roles have the semantics of "shared-by", which is a binary relation, more exactly an equivalence relation. Therefore, transitivity applies. For example, the activities and responsibilities of SE may be shared by SG. From Figure 1 it is obvious that model is applicable in an organization with only one person, who possesses the roles SM/PM, SE, SG, SCM, and SWM, and buys external services for STG and SQA, or alternatively CSQA may be used if applicable.

3.2 eXtra Small Organizations

An eXtra Small organization has 3-15 employees, and a few products/product versions or projects. If the organization is developing products for open market, the time of entering XS model is probably the time of the first release, i.e. the time when the hard work with successive releases and change management begins. Moreover, the life after the first product is of strategic importance. In order to keep the first product successful and on the market, the organization needs to grow. Most probably, new products are on the way, also implying need to recruit more staff. Going from XXS to XS, new roles are needed. Correct project management requires Project Manager (PM) role. Most probably, marketing and sales (MS) is a role with increasing importance at

the time an organization grows out from XXS. System testing (STG) should become an internal role as the number of products/product versions and projects grows.

Figure 2 presents the model for XS organizations. The shading of a circle indicates that the role is an additional one compared to the previous level. For example, when growing out of XXS, PM and MS are roles to be added. The frames depicts projects. The roles within a frame denote roles specific for each project, while the roles outside the frame are roles having responsibility in the whole organization. For example, there is SCM role in each project. SWM is general for the organization, but the person in SWM role may belong to SCM in one or several projects. SM and MS are general for the organization, possibly one and the same person. SM can also work as SE.

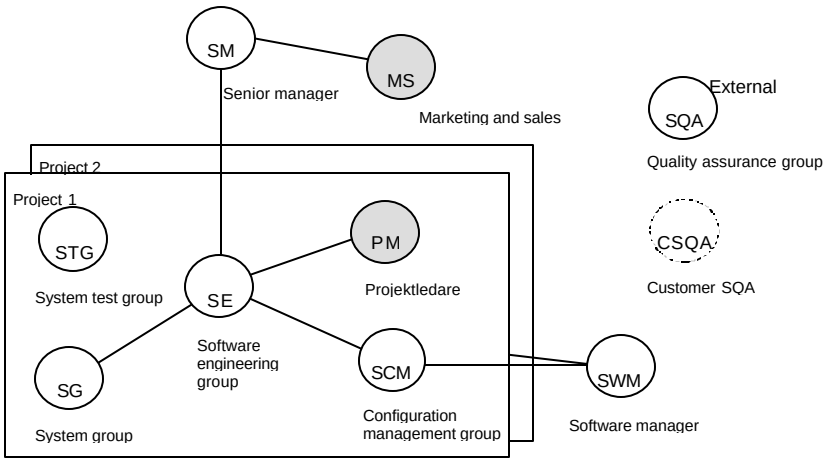

Fig. 2. The roles in an XS organization

3.3 Small Organizations

In a Small organization, the number of employees is 16-50, and there are several products/versions or projects running in parallel. In particular for organizations with product development, documentation support is of great importance implying introduction of Documentation Support Group (DSG). Also, software subcontracting is relevant. This implies the introduction of Software Subcontract Management KPA with the role Software Subcontract Manager (SSM). In some organizations, SM may not be involved in MS activities, but in some organizations, sharing SM and MS by one persons is considered important. Therefore, we leave this possibility open by

retaining the link between MS and SM. The links imply possibility, not a mandatory sharing of responsibilities. MS is related to DSG, and the same person can share DSG and MS roles. Further, until subcontracting is widely practised, the person in role SM may also have the role SSM. Quality assurance in this size of the organization motivates an independent internal SQA, which is also important for obtaining continuity in the software process improvement activities. SCM is a more requiring task in an organization with several projects/products/product versions, and therefore it is assumed that the person in SCM role does not also work as SE. The model for an S organization is presend in Figure 3.

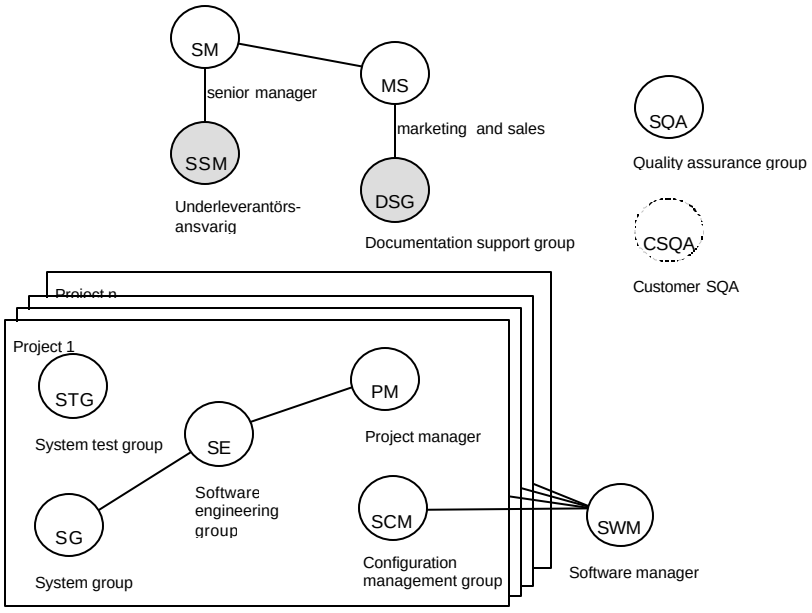

Fig. 3. The roles in an S organization

3.4 Roles and Responsibilities

In Table 1, most of the roles and groups from CMM with some kind of responsibility are presented. In original CMM, the total number of roles is larger, including for example first-line software manager, software estimation group to mention a few. We have omitted all the roles without some significant responsibility. For each of the roles in Table 1 it is indicated whether the role is explicitly included in respective model as is, or if another role is taking the responsibilities of the role. If Abbreviation is the same as the role name indicated under a model, the role is included in the model in

case. Otherwise there is a role name, distinct from the Abbreviation, under respective model. That role is supposed to take the responsibilities of the role in case.

Table 1. The roles in CMM Level 2

Role in CMM	Abbrev	XXS	XS	S
Customer SQA representative	CSQA	CSQA	CSQA	CSQA
Documentation Support Group	DSG	SE	SE	DSG
Hardware engineering group	HE	SG	SG	SG
Marketing and Sales	MS	SM	MS	MS
Project Manager	PM	SM	PM	PM
Project Software Manager	PSM	SM	PM	PM
Senior Manager	SM	SM	SM	SM
Software Configuration Management	SCM	SCM	SCM	SCM
Software Engineering group	SE	SE	SE	SE
SoftWare Manager	SWM		SWM	SWM
Software Project Manager	SPM	SM	PM	PM
Software Quality Assurance group	SQA	SQA	SQA	SQA
Software Subcontract Manager	SSM			SSM
System engineering Group	SG	SG	SG	SG
System Test Group	STG	STG	STG	STG

For each role, there are responsibilities and activities. Here, only the responsibilities and activities for one role, SWM, are described as an example of the representation, in Table 2.

Table 2. The Responsibilities and Activities of SWM

Type	Object of concern
Is responsible for	- the development environment HW and SW
Reviews	-statement of work -project plan -resource estimates -quality assurance plan -configuration management plan

	-the progress against the project plan -specification for subcontract -subcontract

3.5 Documents

CMM requires a large number of documents, required as the basis for activities or as input or output to/from activities. The phrase "according to a documented procedure" appears frequently in CMM text. The documents can be classified as follows:

- policy documents - one for each KPA
- documented procedures, e.g. requirements management, project planning, estimation
- plans, e.g. configuration management plan, project plan, quality assurance plan
- status reports, e.g. quality assurance report

In CMM, the documents are described in text, but it is hard to get oversight of the number and the types of documents is possible to get. In our model, documents are classified by type along the above document types.

3.6 Overview and Activities

One of the intentions with the model is to be easy to get overview. Graphichs support both overview and structure and therefore, the activities are represented graphically, in a simplied form. A graphical overview of only the KPA Requirements Management is given in Figure 4. The activities take system requirements as input, part of which will be allocated to software, and finally resulting in software requirements to be used as the basis for project planning. The activities are reviewed periodically with SM, and periodically as well as event-driven with PM. SQA reviews the activities and work products of the KPA, indicated by SQA outside the KPA frame. SM is responsible for a written organizational policy, and PM for measurement data concerning the status of requirements management activities. An overview of the KPA is given in Figure 4 and the activities in Figure 5.

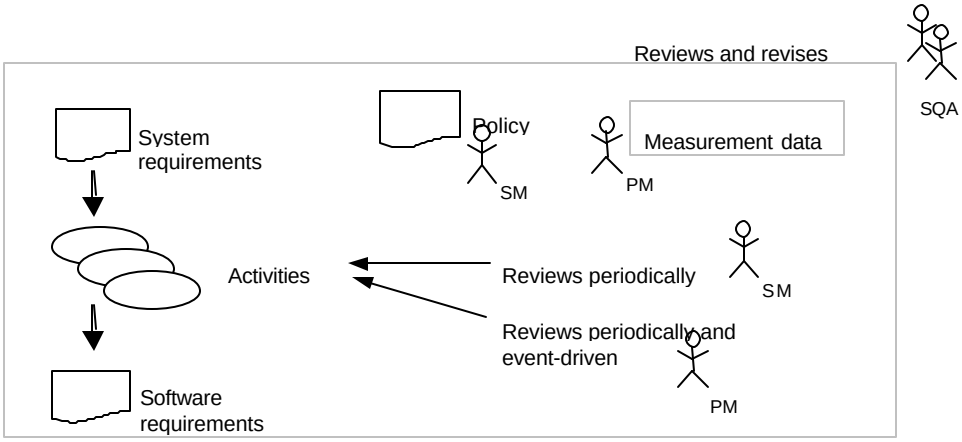

Fig. 4 Requirements Management – overview

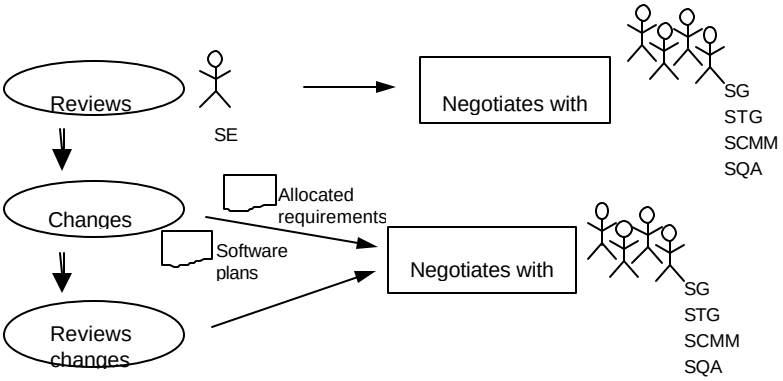

Fig. 5 Requirements Management - activities

4 Applying Dynamic CMM

Dynamic CMM is intended to be adequate to start with for any organization up to 50 employees. The term Dynamic indicates the capability of the model to allow growth while applying the model. Applying the model implies to find an adequate entry point,

application of the model as a road map, and to shift to the next model when appropriate.

4.1 Entry to the Model

If there is a management decision to use Dynamic CMM as the road map for an improvement program, an appropriat entry point to the model is to be found. The basic parameters are the number of employees and the number of products/product versions or projects. The number of employees includes development, management, and marketing and sales staff, but excludes administrative and human resources staff. The number of products and product versions is applicable for product development organizations, while the number of projects is applicable for organizations developing solutions to specific customers. Table 3 gives the basics for the decision of an appropriate entry point to the model.

Table 3. The entry points to Dynamic CMM

	1-2 empl	3-15 empl	>15 empl
1 product/version or project	XXS	XS	S
2-5 products/versions or projects	-	XS	S
>6 products/versions or projects	-	S	S

The alternatives marked with "-" are unusual if not unrealistic. For example, an organization with 2-5 products/product versions, with 1-2 employees might have a theoretical chance to stay on the market with a large number of subcontractors, but it is more a rule than an exception that the company growth implies growth in the number of employees.

If the organization characteristics are outside the limits of the table, it is assumed that the original CMM can be used.

4.2 Application of the Model

The roles and responsibilities are important. They are visible from the respective model and role descriptions. Also, a number of documents serving as the basis must be written, e.g. all the policy documents and procedures for performing the activities. Although all this must be done in order to reach Level 2, it is not realistic to introduce all the KPAs at the same time. It is common knowledge that any but a stepwise introduction of improvement activities are deemed to fail. The order of KPAs is not fixed by the original intentions of CMM. However, assuming that there are no defined and documented processes, but chaos and reactive way of work, it would not make much good to introduce SQA as the first improvement step. At least one other KPAs should be in place before SQA is meaningful. Project management is important as it has con-

sequences for the delivery time and budget. However, if the entry point is XXS, PM is informally handled. The order of the formally handled KPAs, Requirement Management (RM) and Software Configuration Management (SCM), is not generally determined. Which one is introduced first is a question of the situation at hand. For example, if configuration management is experienced as a big problem, it would be natural to start to bring some order in that. At the other hand, if the requirements are volatile, and problems are encountered by introduction of new requirements, RM should probably get first priority.

If the entry point is XS or S, SPP and SPTO are maybe the most adequate KPAs to start with. For the order of RM and SCM, the same reasons should apply as in XXS organization.

4.3 Shifting to the next model

Growth is allowed *within* each model according to the model limits. At the time the number of employees or the number of projects, products, or product versions overrides the limit, the shift to the next higher model is suggested. The introduced roles in the respective shift are visible from the shadowed circles in XS and S models in Figures 2 and 3. The deleted links are not visible, and thereby deserve a comment.

When shifting from XXS to XS, SM no longer has the informal role of PM, but PM becomes an explicit role. However, if SM shares the role of SE, and SE and PM roles are shared by the same person, then by transitivity, that person is SM. It should be observed that the links only imply a possibility of sharing the roles, but are not mandatory. The external role of STG becomes an internal one.

When shifting from XS to S, SM is no longer supposed to work as SE, and thereby not as PM either. The new roles are SSM and DSG, but as they can be shared by SM and MS, there is no requirement of hiring of new people. SQA also becomes an internal role in the organization, no longer hired when needed.

5 Concluding Remarks

In this paper, we have presented Dynamic CMM for small organizations. The model is a step in a research project intending to develop software process improvement and quality models for small organizations. Quality focus is devoted an ever increasing interest in the software community, both research and trade and industry, and it is obvious that there is consciousness of the price and profit of such activities. Still, to actually initiate a process improvement program, an organization needs a road map to start with and to follow. It is vital that the road map is easy to understand, get a quick overview of the magnitude of the effort required, and changes to be expected. Both the management, the engineers and the support staff need to understand and accept the initiative, otherwise the risk of failure is overhelming.

5.1 Considering the Objectives

The model is a simplified variant by will, to make it easier to introduce. The table of responsibilities and activities for the different roles, e.g. Table 2, is an easy way to get oversight. Later, more details can be captured from (Laryd and Orci 1999), or if needed, from CMM. Activity diagrams represent the main activities of the KPAs.

The objectives of the work were to develop models for small organizations, usable from the very start, dynamic in the sense of supporting growth, and easy to get oversight. The *usability* from the very start is intended to be fulfilled by offering a model for any size of company up to 50 employees. The *support for growth* is intended to be provided by the three models and some criteria for shifting from one to another. The easyness of *oversight* is intended to be taken care of the graphical representation, the role models, tables of responsibilities and activities for respective role, and the lists of documents by type.

The model must be applied to real case studies, in order to assess its value for software process improvement. It will, however, be difficult to distinguish the contributions of the model as such and the contributions caused by the improvement effort and people related contribution.

Nevertheless, the model is going to be used in industry, and the first case study is in progress. The organization is of size S, a product development company. So far, a rather informal assessment has been conducted, and a number of weaknesses has been identified. The organization has used the roles and responsibility descriptions and appointed people to the roles. The next thing is to develop all the required documents in KPAs RM, SPP, and SPTO. The organization has no subcontractors, and can postpone the SSM to a later stage. SQA person is already in place, but the SQA activities will be started after RM, SPP, and SPTO will be in place. SCM is already practised to a certain extent, and that KPA will only require minor changes to the existing working practices.

What may turn out from the case study is that the model for S is not appropriate in some sense. It might complicate the communication and development work. For example, if project management turns out to be hard to perform in a way prescribed by the model, it may be an indication that our model is not down-scaled enough. It might also be because the people are unfamiliar with formal project management. Similar questions may certainly come up also in the other KPAs. The essential task in such a case is to try to infer the real cause of the problem: whether it is the model, or something else.

5.2 Further Research

Further research ideas involve development of an integrated model for small organizations, not only involving software development process improvement, but also adequate properties of human resource management in software development companies, based on People-CMM (Curtis et al, 1995), and software process improvement on

individual as well as on team level, based on PSP (Personal Software Process) (Humphrey 1995) and TSP (Team Software process) (Humphrey 2000), respectively.

References

Caputo K. Implementation Guide - Choreographing Software Process Improvement. Addison Wesley, 1998.

Curtis, B.,Hefley, W.E., Miller, S. People Capability Maturity Model. Software Engineering Institute, Carnegie Mellon University, 1995.

Grady, R.B. Practical Software Metrics for Project Management and Process Improvement. Prentice Hall, 1992.

Humphrey W. S. A Discipline for Software Engineering. Addison Wesley, 1995.

Humphrey W. S. Introduction to the Team Software process. Addison Wesley, 2000.

Laryd, A., Orci, T. Dynamic Capability Maturity Model for Small Organizations. Umeå University, 1999.

Paulk, M.C. et al. The Capability Maturity Model - Guidelines for Improving the Software Process, Addison-Wesley, 1995.

Zahran, S. Software Process Improvement. Practical Guidelines for Business Success. Addison-Wesley, 1997.