# Capability Maturity Model
# for
## Extra Small Organizations
## Level 2

Terttu Orci

Umeå Universitet
Institutionen för datavetenskap

2000-09-10
Version 1.0

UMINF 00.13

# 1 Introduction

## 1.1 Background

Software CMM[1] has gained wide acceptance in software process improvement, especially in USA, but now even at the European market. Software CMM, called plain CMM in the sequel, is publically available and has been described in literature [3], and is also available on the world wide web. The gains of using CMM for improving the software processes of an organization have also been published by a number of large US organizations, e.g. [2]. In Sweden, only the large organizations have adopted CMM, e.g. Ericsson and ABB, but they have not published same detail of investment or ROI of adopting CMM as the US organizations. Exact figures might also be hard to determine, as it is difficult to derive the cause of an event to a single factor, like CMM use. Most likely, a success depends on several factors, not easily distinguishable from each other.

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, also information about implementing CMM is publically available, e.g. [1],[9], presenting guidelines and pitfalls for CMM implementation, supported by forms and templates.

A software development organization with notably symptoms of chaos caused by inmaturity, should start a software process improvement program. Such a program may adopt CMM, or any other model publically available, or it can be approached in other ways, e.g. by using GQM [8] to derive the attributes and metrics to be collected, which assess the state of the

[1] The abbreviations are explained in the Appendix A.

processes, and by developing and implementing improved processes based on the GQM analysis. CMM – as well as any other model presenting the maturity levels stepwise and to enough detail, e.g. SPICE [10], offer however a path which has historically proved to lead to success with high probability, and being fundamentally based on sound management practices, witnessed by the working philosophy of CMM developers at the Software Engineering Institute. The model offers an improvement ladder, by introducing more and more software engineering and management practices when climbing on the ladder. It is intuitively a comfortable way of executing an improvement program. Thereby we do not argue that CMM is the only or even the best model for getting good improvement results.

It is often argued that about 2/3 of the organizations worldwide would be on the Level 1 if assessed by CMM. When a chaos is already there, it is certainly wise to start an improvement program. It would, however, been even wiser to start an improvement program *before* the chaos became a reality. This possibility is for obvious reasons only open for small, newly initiated organizations, with a few software engineers, and with a simple management hierarchy. In the start-up phase, with a few software engineers, with one or at most a few versions of one product, the communication is simple, configuration management is simple, everybody knows what the colleque 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 specialised to certain tasks, and also an organizational hierarchy in place, there will be even more versions of the first product, but there will also be new products, 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 start to get complicated: it is not that easy to know what the colleque engineer is doing, it is not easy to know which version of the compiler was used to generate a particular version of a particular product to a particular customer, the management has no oversight over the development, but needs some visible evidence of the progress in the development projects. This is exactly the same situation as is the reality for the 2/3 of all the organizations worldwide.

## 1.2 The Problem

In order to avoid that common situation, an organization should have the focus on software process even before it is really needed, i.e. almost from its start. The existing models are not, however, adequate for such a situation. For example, CMM proposes more than 25 groups and organizational roles, with various tasks and responsibilities. In a small organization, there is not even enough people to fill all the roles proposed. Therefore, the models need to be scaled down to the needs and possibilities of small organizations. At the same time, it is important that such a model is applicable continuously in the course of time, if the organization grows. Therefore, the model should advice steps for introducing activities to be implemented in the course of time, otherwise it will cease to be useful after a while.

## 1.3 The Objectives and Methodology

The research reported in this paper aims to develop models like CMM for small and medium size organizations according to the requirements of being useful *before* chaos and *while* growing. The scope of the model is Level 2 of CMM. When the organization has reached the Level 2 maturity, most likely, the original CMM model can be used. In order to serve a business before chaos, already from its start, and to provide for growing needs, the SME businesses are classified into three classes: eXtra eXtra Small (XS), Extra Small (XS), and Small (S). Further, the type of business has its implications, whether it is the development of software for open market, or the provision of customer specific solutions of type information technology.

In this paper, we limit ourselves to organizations in their starting phase, i.e. for XS organizations, both for organizations developing software to an open market as well as organizations providing information technology solutions to specific customers. We are aware that the classes along the size and the type of business might not cover all possible lines of business. For example, an organization growing extremely fast might not find the models with three alternatives accurate, as they represent a stepwise growth, but a continuous model for growth might be needed. In this phase of the research, we limit ourselves to the stepwise growth model, however.

In order to propose CMM for XS, an revolutionary approach of experimental paradigm has been used. Such an approach starts with proposing a new model, and applying the model to case studies, measuring and analyzing, and validating the model. The new model is the CMM for XS. The application of the model to case studies will be done later, and will not be a part of this paper. The models XXS and S are described in [3],[4]. Some advice which model to use and when to change the model, are presented in [5]. The model building requires knowledge and understanding of the intentions of the CMM, as well as of the tasks and activities to be performed by the people possessing the roles. In additions to that, the conditions and situations in XS type of organizations must be well-understood, and the difficulties and obstacles when proposing and implementing quality improvement in small organizations [2].The task is to reduce the number of roles, which requires merging of several roles to one. For that reason, it is important to study the dependencies between the roles, in order to avoid the original intentions of the model to be lost. For example, it is intuitively clear that software quality assurance (SQA) role should not be possessed by a person who is part of the development team, as the SQA role is supposed to assure the quality of the work products developed, among other things.

The basis for the work is CMM Level 2 and characterization of XXS organizations. Some of the roles proposed are not relevant for small organisations. Some of the roles are relevant as to their tasks and responsibilities, but not to formally appoint a person to the role.

---

[2] Astrid Laryd, Pragma Konsult AB, has contributed with her competence in quality improvement work from Swedish industry. The association to the EC supported project EUREX (European Experience Exchange) of Terttu Orci has given some insight to the process improvement work in the European industry.

In Section 2, an XS organization is characterized, and the roles in CMM are classified. Further, the role taking over the responsibility of an RI role are presented. Section 3 presents a CMM for XS organizations, and Section 4 contains concluding remarks.

## 2 Characteristics and Roles in Extra Small Organizations

An eXtra Small organization has 3-15 employees, and a few product/product versions. 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. The roles PM, MS, STG, and SWM are essential and classified as RE. STG should be an internal role. Still, a person possessing one of those roles may very well work also in another role. In organizations developing customer specific solutions, SWM becomes the expert on computers and adjustments at the customer site. Further, we assume that subcontracting is not practized in a XS organization.

## 2.1 Roles and Responsibilities

Many of the roles in CMM do not imply any responsibilities, e.g. affected groups, but rather they appear in other roles' descriptions as partners for negotiations. Such roles are also manager and managers in SQA reporting chain, as it is not realistic to assume more than one manager in XXS organizations. Some roles are irrelevant, e.g the roles associated to SSM. Those roles are omitted in Table 1. A role in table under the heading Role in CMM, which should be formally appointed, appears also in the table under Role in charge. A role under Role in CMM, which does not need a formal appointment, but the tasks and responsibilities are assigned to another role, has that role under the heading Role in charge.

| **Role in CMM** | **Abbrev** | **Role in charge** |
|---|---|---|
| Contract management group | | MS |
| Customer SQA representative | CSQA | CSQA |
| Documentation Support Group | DSG | SE |
| Hardware engineering group | | SG |
| Marketing and Sales | MS | MS |
| Project Manager | PM | PM |
| Senior Manager | SM | SM |
| Software Configuration Management | SCM | SCM |
| Software Engineering group | SE | SE |
| Software estimating group | | PM |
| SoftWare Manager | SWM | SWM |
| Software Quality Assurance group | SQA | SQA |
| System engineering Group | SG | SG |
| System Test Group | STG | STG |

*Table 1. The roles*

In XS, the tasks of *Contract management group* are proposed to be carried out by Marketing and Sales (MS).

*Hardware engineering group* has no responsibility is CMM, rather the group is involved in negotiations concerning the requirements, etc. SG group, responsible for the entire system with software and hardware, is supposed to take over the tasks of hardware engineering group.

*Software estimating group* is an expert group, possessing expert skills in estimation. In an XS, such expert group is not realistic to assume: in the beginning of an organization's life, there

is normally no such skills within the organization, neither is it realistic to hire those services. Estimation, to the extent done, is assumed to be handled by *project manager PM.*

*SoftWare Manager (SWM)* is the manager of the software development environment, but also of the operative software and hardware in the organization.

Documentation support might be needed, but is is not realisticto assume a formal role of *Documentation Support Group (DSG).* The responsibility is assigned to the *Software Engineer (SE).*

*CSQA* is relevant only for organizations developing customer specific solutions, and if the customer has a CSQA.

*System Engineering Group* (SG) is responsible for the systems with both software and hardware.

*Software configuration control board – SCCB* is required to exist according to CMM. In an XS, the tasks of the board are assigned to Software Configuration Management group (SCM).

*System Test Group* (STG) is supposed to be responsible for system tests, while SE has the responsibility of tests during development. Validation and verification including testing is of primary importance to obtain a quality product.

In Figure 1, the roles in an XS organization are presented. The associations 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. The frames represent projects. A role within a frame is specific to a project, e.g. each project has a project manager (PM), while the role SWM is shared between several projects. A shaded circle means that the role is new compared to the "previous" model, i.e. an organization growing from an XXS organization to an XS organization, introduces the shaded role.

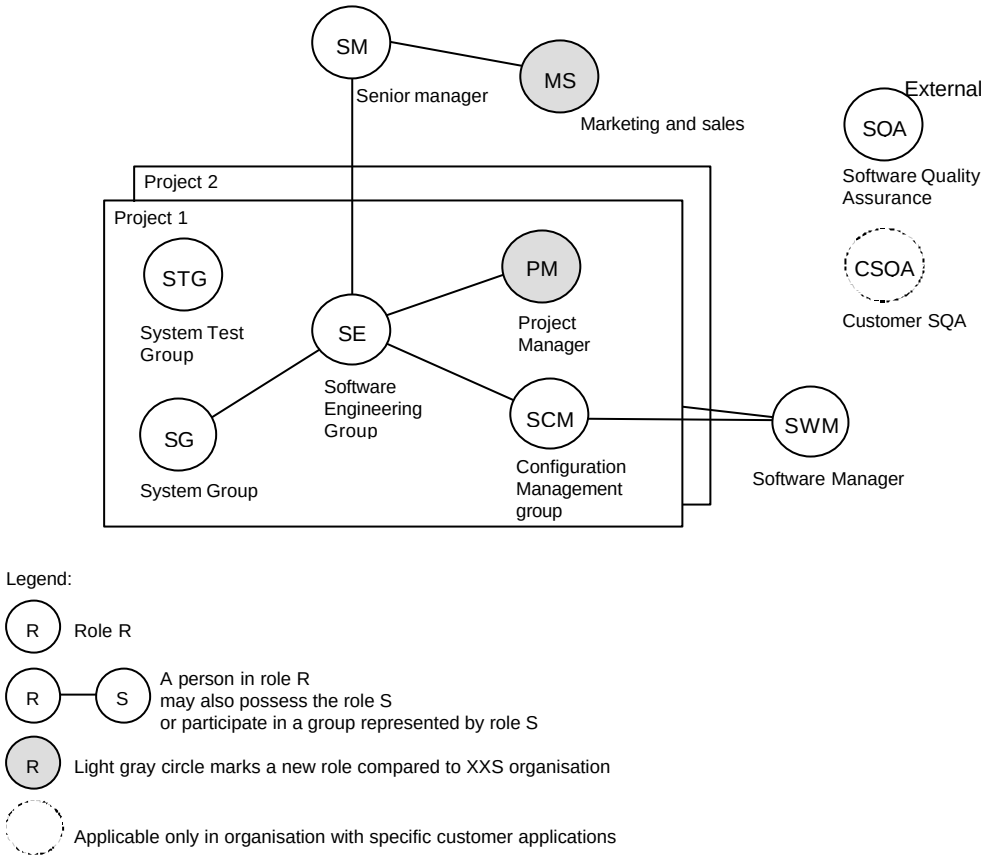


*Figure 1. The roles in an XS organization*

# 3 CMM Level 2 for XS Organizations

In this section, the CMM for XS organizations is presented. It is relevant for both types of organizations: organizations developing software to open market and organizations providing information technology solutions to specific customers. When the model differs, it will be explicitly stated.

The text is from CMM in [3], modified according to the changed roles.

## 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.

Requirements Management involves establishing and maintaining an agreement with the customer on the requirements for the software project. This agreement is referred to as the "system requirements allocated to the software." The "customer" may be interpreted as the system engineering group, the marketing group, another internal organization, or an external customer. The agreement covers both the technical and nontechnical (e.g., delivery dates) requirements. The agreement forms the basis for estimating, planning, performing, and track the software project's activities throughout the software life cycle.

The allocation of the system requirements to software, hardware, and other system components (e.g. humans) may be performed by a group external to the software engineering group (e.g., the system engineering group), and the software engineering group may have no direct control of this allocation. Within the constraints of the project, the software engineering group takes appropriate steps to ensure that the system requirements allocated to software, which it is responsible for addressing, are documented and controlled.

To achieve this control, the software engineering group reviews the initial and revised system requirements allocated to software to resolve issues before they are incorporated into the software project. Whenever the system requirements allocated to software are changed, the affected software plans, work products, and activities are adjusted to remain consistent with the updated requirements.

**Goals**

**Goal 1** *System requirements allocated to software are controlled to establish a baseline for software engineering and management use.*

**Goal 2** *Software plans, products and activities are kept consistent with the system requirements allocated to software.*

**Commitment to Perform**

**Commitment 1** *The project follows a written organizational policy for managing the system requirements allocated to software*

The system requirements allocated to the software are referred to as "allocated requirements" in these practices. The allocated requirements are the subset of the system requirements that are to be implemented in the software components of the system. The allocated requirements are a primary input to the software development plan. Software requirements analysis elaborates and refines the allocated requirements and results in software requirements which are documented.

This policy typically specifies that:

1. The allocated requirements are documented.
2. The allocated requirements are reviewed by:
   the software managers, and
   other affected groups.

   Examples of affected groups include:
   - system test (STG),
   - software engineering (SE),
   - system group (SG),
   - software quality assurance (SQA), and
   - software configuration management (SCM).

3. The software plans, work products, and activities are changed to be consistent with changes to the allocated requirements.

**Ability to Perform**

**Ability 1** *For each project, responsibility is established for analyzing the system requirements and allocating them to hardware, software and other system components.*

Analysis and allocation of the system requirements is not the responsibility of the software engineering group but is a prerequisite for their work.

This responsibility covers:

1. Managing and documenting the system requirements and their allocation throughout the project's life.
2. Effecting changes to the system requirements and their allocation.

**Ability 2** *The allocated requirements are documented.*

The allocated requirements include:

1. The nontechnical requirements (i.e., the agreements, conditions, and/or contractual terms) that affect and determine the activities of the software project.

   Examples of agreements, conditions, and contractual terms include:
   - products to be delivered,
   - delivery dates, and
   - milestones

2. The technical requirements for the software.

   Examples of technical requirements include:
   - end use, operator, support, or integration functions;
   - performance requirements;
   - design constraints;
   - programming language; and
   - interface requirements

3. The acceptance criteria that will be used to validate that the software products satisfy the allocated requirements.

**Ability 3** *Adequate resources and funding are provided for managing the allocated requirements.*

1. Software engineers are assigned to manage the allocated requirements.
2. Tools to support the activities for managing requirements are made available.

   Example of support tools include:
   - spreadsheet programs,
   - tools for configuration management,
   - tools for traceability, and
   - tools for text management.

**Ability 4** *Members of the software engineering group and other software-related groups are trained to perform their requirements management activities.*

Examples of training include:
- the methods, standards, and procedures used by the project, and
- the application domain.

## Activities Performed

**Activity 1** *The software engineering group reviews the allocated requirements before they are incorporated into the software project.*

1. Incomplete and missing allocated requirements are identified.
2. The allocated requirements are reviewed to determine whether they are:
   - feasible and appropriate to implement in software,
   - clearly and properly stated,
   - consistent with each other, and
   - testable
3. Any allocated requirements identified as having potential problems are reviewed with the system group responsible for analyzing and allocating

system requirements, and necessary changes are made.

4. Commitments resulting from the allocated requirements are negotiated with the affected groups.

Examples of affected groups include:

- software engineering,
- project manager,
- system group,
- system test group,
- software quality assurance,
- software configuration management, and
- marketing and sales.

Refer to Activity 6 of the Software Project Planning key process area for practices covering negotiating commitments.

**Activity 2** *The software engineering group uses the allocated requirements as the basis for software plans, work products, and activities.*

The allocated requirements:

1. Are managed and controlled.

"Managed and controlled" implies that the version of the work product in use at a given time (past or present) is known (i.e., version control), and changes are incorporated in a controlled manner (i.e., change control).

If a greater degree of formality than is implied by "managed and controlled" is desired, the work product can be placed under the full discipline of configuration management, as is described in the Software Configuration Management key process area.

2. Are the basis for the software development plan.
3. Are the basis for developing the software requirements.

**Activity 3** *Changes to the allocated requirements are reviewed and incorporated into the software project.*

1. The impact to existing commitments is assessed, and changes are negotiated as appropriate.
   - Changes to commitments made to individuals and groups external to the organization are reviewed with senior management.

Refer to Activity 4 of the Software Project Planning key process area and Activity 3 of the Software Project Tracking and Oversight key process area for practices covering commitments made external to the organization.

   - Changes to commitments within the organization are negotiated with the affected groups.

Refer to Activities 5, 6, 7, and 8 of the Software Project Tracking and Oversight key process area for practices covering negotiating changes to commitments.

2. Changes that need to be made to the software plans, work products, and activities resulting from changes to the allocated requirements are:
    - identified,
    - evaluated,
    - assessed for risk,
    - documented,
    - planned,
    - communicated to the affected groups and individuals, and tracked to completion.

## Measurement and Analysis

**Measurement 1** *Measurements are made and used to determine the status of the activities for managing the allocated requirements.*

Examples of measurements include:

- status of each of the allocated requirements
- change activity for the allocated requirements, and
- cumulative number of changes to the allocated requirements, including total number of changes proposed, open, approved and incorporated into the system baseline.

## Verifying Implementation

**Verification 1** *The activities for managing the allocated requirements are reviewed with senior manager on a periodic basis.*

The primary purpose of periodic reviews by senior management is to provided awareness of and insight into software process activities at an appropriate level of abstraction and in a timely manner. The time between reviews should meet the needs of the organization and may be lengthy, as long as adequate mechanisms for exception reporting are available.

Refer to Verification 1 of the Software Project Tracking and Oversight key process area for practices covering the typical content of senior management oversight reviews.

**Verification 2** *The activities for managing the allocated requirements are reviewed with the project manager on both a periodic and event-driven*

*basis.*

Refer to Verification 2 of the Software Project Tracking and Oversight key process area for practices covering the typical content of project management oversight reviews.

**Verification 3** *The software quality and assurance reviews and/or audits the activities and work products for managing the allocated requirements and reports the results.*

Refer to the Software Quality Assurance key process area.

At a minimum, these reviews and/or audits verify that:

1. The allocated requirements are reviewed, and problems are resolved before the software engineering group commits to them.
2. The software plans, work products, and activities are appropriately revised when the allocated requirements change.
3. Changes to commitments resulting from changes to the allocated requirements are negotiated with the affected groups.

## Software Project Planning

The purpose of Software Project Planning is to establish reasonable plans for performing the software engineering and for managing the software project.

Software Project Planning involves developing estimates for the work to be performed, establishing the necessary commitments, and defining the plan to perform the work.

The software planning begins with a statement of the work to be performed and other constrains and goals that define and bound the software project (those established by the practices of the Requirements Management key process area). The software planning process includes steps to estimate the size of the software work products and the resources needed, produce a schedule, identify and assess software risks, and negotiate commitments. Iterating through these steps may be necessary to establish the plan for the software project (i.e., the software development plan).

This plan provides the basis for performing and managing the software project's activities and addresses the commitments to the software project's customer according to the resources, constraints, and capabilities of the software project.

### Goals

**Goal 1** *Software estimates are documented for the use in planning and tracking the software project.*

**Goal 2** *Software project activities and commitments are planned and documented.*

**Goal 3** *Affected groups and individuals agree to their commitments related to the software project.*

**Commitment to Perform**

**Commitment 1** *A project manager is designated to be responsible for negotiating commitments and developing the project's software development plan.*

**Commitment *2*** *The project follows a written organizational policy for planning a software project.*

This policy typically specifies that:

1. The system requirements allocated to software are used as the basis for planning the software project.

Refer to Activity 2 of the Requirements Management key process area.

2. The software project's commitments are negotiated between:
   - the project manager, and
   - the software manager.

3. Involvement of other engineering groups in the software activities is negotiated with these groups and is documented.

Examples of other engineering groups include:
- system group and,
- system test group.

4. Affected groups review the software project's:
   - software size estimates,
   - effort and cost estimates,
   - schedules, and
   - other commitments.

Examples of affected groups include:

- software engineering ,
- project manager,
- system group,
- system test group,
- software quality assurance,
- software configuration management, and
- marketing and sales.

5. Senior manager reviews all software project commitments made to individuals and groups external to the organization.

6. The project's software development plan is managed and controlled.

The term "software development plan" is used throughout these practices to refer to the overall plan for managing the software project. The use of "development" terminology is not intended to exclude software maintenance or support projects and should be appropriately interpreted in the context of the individual project.

"Managed and controlled" implies that the version of the work product in use at a given time (past of present) is known (i.e., version control) and changes are incorporated in a controlled manner (i.e., change control).
If a greater degree of control than is implied by "managed and controlled" is desired, the work product can be placed under the full discipline of configuration management, as is described in the Software Configuration Management key process area.

## Ability to Perform

**Ability 1** *A documented and approved statement of work exists for the software project.*

1. The statement of work covers:
   - scope of the work,
   - technical goals and objectives,
   - identification of customers and end users,

     The end users referred to in these practices are the customer designated end users or representatives of the end users.

   - imposed standards,
   - assigned responsibilities,
   - cost and schedule constraints and goals,
   - dependencies between the software project and other organizations,

     Examples of other organizations include:

- the customer, and
- joint venture partners.

- resource constraints and goals, and
- other constraints and goals for development and/or maintenance.

2. The statement of work is reviewed by:

   - the project manager,
   - the software manager, and
   - other affected groups.

3. The statement of work is managed and controlled.

**Ability 2** *Responsibilities for developing the software development plan are assigned.*

1. The project manager, directly or by delegation, coordinates the project's software planning.
2. Responsibilities for the software work products and activities are partitioned and assigned to software manager in a traceable, accountable manner.

   Examples of software work products include:
   - work products for delivery to the external customer or end users, as appropriate;
   - work products for use by other engineering groups; and
   - major work products for internal use by the software engineering group.

**Ability 3** *Adequate resources and funding are provided for planning the software project.*

1. Where feasible, experienced individuals, who have expertise in the application domain of the software project being planned, are available to develop the software development plan.
2. Tools to support the software project planning activities are made available.

   Examples of support tools include:
   - -preadsheet programs,
   - estimating models, and
   - project planning and scheduling programs.

**Ability 4** *The software manages, software engineers, and other individuals involved in the software project planning are trained in the software*

*estimating and planning procedures applicable to their areas of responsibility.*

**Activities Performed**

**Activity 1** *The software engineering group participates on the project proposal term.*

1. The software engineering group is involved in:

    - proposal preparation and submission,
    - clarification discussions and submissions, and
    - negotiations of changes to commitments that affect the software project.

2. The software engineering group reviews the project's proposed commitments.

    Examples of project commitments include:
    - the project's technical goals and objectives;
        - the system and software technical solution;
        - the software budget, schedule, and resources; and
        - the software standards and procedures.

**Activity 2** *Software project planning is initiated in the early stages of, and in parallel with, the overall project planning.*

**Activity 3** *The software engineering group participates with other affected groups in the overall project planning throughout the project's life.*

1. The software engineering group reviews the project-level plans.

**Activity 4** *Software project commitments made to individuals and groups external to the organization are reviewed with senior manager according to a documented procedure.*

**Activity 5** A software life cycle with predefined stages of manageable size is identified or defined.

Examples of software life cycles include:
- waterfall,
- overlapping waterfall,

- spiral,
- serial build, and
- single prototype/overlapping waterfall.

**Activity 6** *The project's software development plan is developed according to a documented procedure.*

This procedure typically specifies that:

1. The software development plan is based on and conforms to:
   - the customer's standards, as appropriate;
   - the project's standards;
   - the approved statement of work; and
   - the allocated requirements.

2. Plans for software related groups involved in the activities of the software engineering group are negotiated with those groups, the support efforts are budgeted, and the agreements are documented.

   Examples of software-related groups include:
   - software quality assurance, and
   - software configuration management.

   Examples of other engineering groups include:
   - system group, and
   - system test group.

3. Plans for involvement of the software engineering group in the activities of other software-related groups and other engineering groups are negotiated with those groups, the support efforts are budgeted, and the agreements are documented.

4. The software development plan is reviewed by:
   - the project manager,
   - the software manager, and
   - other affected groups.

5. The software development plan is managed and controlled.

**Activity 7** *The plan for the software project is documented.*

In the key practices, this plan or collection of plans is referred to as the software development plan.

Refer to Activity 1 of the Software Project Tracking and Oversight key process area for practices concerning the project's use of the software development plan.

The software development plan covers:

1. The software project's purpose, scope, goals, and objectives.
2. Selection of a software life cycle.
3. Identification of the selected procedures, methods, and standards for developing and/or maintaining the software.

   Examples of software standards and procedures include:
   - software development planning,
   - software configuration management,
   - software quality assurance,
   - software design,
   - problem tracking and resolution, and
   - software measurement.

4. Identification of software work products to be developed.
5. Size estimates of the software work products and any changes to the software work products.
6. Estimates of the software project's effort and costs.
7. Estimated use of critical computer resources.
8. The software project's schedules, including identification of milestones and reviews.
9. Identification and assessment of the project's software risks.
10. Plans for the project's software engineering facilities and support tools.

**Activity 8** *Software work products that are needed to establish and maintain control of the software project are identified.*

Refer to Activity 4 of the Software Configuration Management key process area.

**Activity 9** *Estimates for the size of the software work products (or changes to the size of software work products) are derived according to a documented procedure.*

This procedure typically specifies that:

1. Size estimates are made for all major software work products and activities.

   Examples of software size measurements include:
   - function points,
   - feature points,
   - lines of code,
   - number of requirements, and
   - number of pages.

Examples of types of work products and activities for which size estimates are made include:

- operational software and support software,
- deliverable and nondeliverable work products,
- software and nonsoftware work products (e.g., documents), and
- activities for developing, verifying, and validating work products.

2. Software work products are decomposed to the granularity needed to meet the estimating objectives.
3. Historical data are used where available.
4. Size estimating assumptions are documented.
5. Size estimates are documented, reviewed, and agreed to.

Examples of groups and individuals who review and agree to size estimates include:

- the project manager, and
- the software manager.

**Activity 10** *Estimates for the software project's effort and costs are derived according to a documented procedure*.

This procedure typically specifies that:

1. Estimates for the software project's effort and costs are related to the size estimates of the software work products (or the size of the changes).
2. Productivity data (historical and/or current) are used for the estimates when available; sources and rationale for these data are documented.

    - The productivity and cost data are from the organization's projects when possible.
    - The productivity and cost data take into account the effort and significant costs that go into making the software work products.

    Examples of significant costs that go into making the software work products include:

    - direct labor expenses,
    - overhead expenses,
    - travel expenses, and
    - computer use costs.

3. Effort, staffing, and cost estimates are based on past experience.

    - Similar projects should be used when possible.
    - Time phasing of activities is derived.
    - Distributions of the effort, staffing, and cost estimates over the software life cycle are prepared.

4. Estimates and the assumptions made in deriving the estimates are documented, reviewed, and agreed to.

**Activity 11** *Estimates for the project's critical computer resources are derived according to a documented procedure.*

Critical computer resources may be in the host environment, in the integration and testing environment, in the target environment, or in any combination of these.

This procedure typically specifies that:

1. Critical computer resources for the project are identified.

   Examples of critical computer resources include:
   - computer memory capacity,
   - computer processor use, and
   - communications channel capacity.

2. Estimates for the critical computer resources are related to the estimates of:
   - the size of the software work products,
   - the operational processing load, and
   - the communications traffic.

3. Estimates of the critical computer resources are documented, reviewed, and agreed to.

**Activity 12** *The project's software schedule is derived according to a documented procedure.*

This procedure typically specifies that:

1. The software schedule is related to:

   - the size estimate of the software work products (or the size of changes), and
   - the software effort and costs.

2. The software schedule is based on past experience.

   - Similar projects are used when possible.

3. The software schedule accommodates the imposed milestone dates,

critical dependency dates, and other constraints.
4. The software schedule activities are of appropriate duration and the milestones are of appropriate time separation to support accuracy in progress measurement.
5. Assumptions made in deriving the schedule are documented.
6. The software schedule is documented, reviewed, and agreed to.

**Activity 13** *The software risks associated with the cost, resource, schedule, and technical aspects of the project are identified, assessed, and documented.*

1. The risks are analyzed and prioritized based on their potential impact to the project.
2. Contingencies for the risks are identified.

Examples of contingencies include:
- schedule buffers,
- alternate staffing plans, and
- alternate plans for additional computing equipment.

**Activity 14** *Plans for the project's software engineering facilities and support tools are*

*prepared.*

1. Estimates of capacity requirements for these facilities and support tools are based on the size estimates of the software work products and other characteristics.

Examples of software development facilities and support tools include:
- software development computers and peripherals,
- software test computers and peripherals,
- target computer environment software, and
- other support software

2. Responsibilities are assigned and commitments are negotiated to procure or develop these facilities and support tools.
3. The plans are reviewed by all affected groups.

**Activity 15** *Software planning data are recorded.*

1. Information recorded includes the estimates and the associated information needed to reconstruct the estimates and assess their reasonableness.
2. The software planning data are managed and controlled.

## Measurement and Analysis

**Measurement 1** *Measurements are made and used to determine the status of the software planning activities.*

Examples of measurements include:

- completions of milestones for the software project planning activities compared to the plan; and
- work completed, effort expended, and funds expended in the software project planning activities compared to the plan.

## Verifying Implementation

**Verification 1** *The activities for software project planning are reviewed with senior manager on a periodic basis.*

The primary purpose of periodic reviews by senior management is to provide awareness of, and insight into, software process activities at an appropriate level of abstraction and in a timely manner. The time between reviews should meet the needs of the organization and may be lengthy, as long as adequate mechanisms for exception reporting are available.

1. The technical, cost, staffing, and schedule performance is reviewed.
2. Conflicts and issues not resolvable at lower levels are addressed.
3. Software project risks are addressed.
4. Action items are assigned, reviewed, and tracked to closure.
5. A summary report from each meeting is prepared and distributed to the affected groups and individuals.

**Verification 2** *The activities for software project planning are reviewed with the project manager on both a periodic and event-driven basis.*

1. Affected groups are represented.
2. Status and current results of the software project planning activities are reviewed against the software project's statement of work and allocated requirements.
3. Dependencies between groups are addressed.
4. Conflicts and issues not resolvable at lower levels are addressed.
5. Software project risks are reviewed.
6. Action items are assigned, reviewed, and tracked to closure.
7. A summary report from each meeting is prepared and distributed to the affected groups and individuals.

**Verification 3** *The software quality assurance group reviews and/or audits the*

*activities and work products for software project planning and reports the results.*

Refer to the Software Quality Assurance key process area.

At a minimum, the reviews and/or audits verify:

1. The activities for software estimating and planning.
2. The activities for reviewing and making project commitments.
3. The activities for preparing the software development plan.
4. The standards used for preparing the software development plan.
5. The content of the software development plan.

## Software Project Tracking and Oversight

The purpose of Software Project Tracking and Oversight is to provide adequate visibility into actual progress so that management can take effective actions when the software project's performance deviates significantly from the software plans.

Software Project Tracking and Oversight involves tracking and reviewing the software accomplishments and results against documented estimates, commitments, and plans, and adjusting these plans based on the actual accomplishments and results.

A documented plan for the software project (i.e., the software development plan, as described in the Software Project Planning key process area) is used as the basis for tracking the software activities, communicating status, and revising plans. Software activities are monitored by the management. Progress is primarily determined by comparing the actual software size, effort, cost, and schedule to the plan when selected software work products are completed and at selected milestones. When it is determined that the software project's plans are not being met, corrective actions are taken. These actions may include revising the software development plan to reflect the actual accomplishments and replanning the remaining work or taking actions to improve the performance.

### Goals

**Goal 1** *Actual results and performance are tracked against the software plans.*

**Goal 2** *Corrective actions are taken and managed to closure when actual results and performance deviate significantly from the software plans.*

**Goal 3** *Changes to software commitments are agreed to by the affected groups and individuals.*

### Commitment to Perform

**Commitment 1** *A project manager is designated to be responsible for the project's software activities and results.*

**Commitment 2** *The project follows a written organizational policy for managing the software project.*

This policy typically specifies that:

1. A documented software development plan is used and maintained as the basis for tracking software project.
2. The project manager is kept informed of the software project's status and issues.
3. Corrective actions are taken when the software plan is not being achieved, either by adjusting the performance or by adjusting the plans.
4. Changes to the software commitments are made with the involvement and agreement of the affected groups.

   Examples of affected groups include:
   - software engineering,
   - project manager,
   - system group,
   - system test group,
   - software quality assurance,
   - software configuration management, and
   - marketing and sales.

5. Senior management reviews all commitment changes and new software project commitments made to individuals and groups external to the organization.

### Ability to Perform

**Ability 1** *A software development plan for the software project is documented and approved.*

Refer to Activities 6 and 7 of the Software Project Planning key process area for practices covering the software development plan.

**Ability 2** *The project manager explicitly assigns responsibility for software work products and activities.*

The assigned responsibilities cover:

1. The software work products to be developed or services to be provided.
2. The effort and cost for these software activities.
3. The schedule for these software activities.
4. The budget for these software activities.

**Ability 3** *Adequate resources and funding are provided for tracking the software project.*

1. The software manager and the project manager are assigned specific responsibilities for tracking the software project.
2. Tools to support software tracking are made available.

Examples of support tools include:

- spreadsheet programs, and
- project planning/scheduling programs.

**Ability 4** *The software managers are trained in managing the technical and personnel aspects of the software project.*

Examples of training include:

- managing technical projects;
- tracking and oversight of the software size, effort, cost, and schedule; and
- managing people.

## Activities Performed

**Activity 1** *A documented software development plan is used for tracking the software activities and communicating status.*

Refer to Activity 7 of the Software Project Planning key process area for practices covering the content of the software development plan.

This software development plan is:

1. Updated as the work progresses to reflect accomplishments, particularly when milestones are completed.

2. Readily available to:
   - the software engineering group (including all subgroups, such as software design),
   - the software managers,
   - the project manager,
   - senior management, and
   - other affect groups.

**Activity 2** *The project's software development plan is revised according to a documented procedure.*

Refer to Activity 6 of the Software Project Planning key process area to practices covering the activities for producing the software development plan.

This procedure typically specifies that:

1. The software development plan is revised, as appropriate, to incorporate plan refinements and incorporate plan changes, particularly when plans change significantly.

   Interdependencies between the system requirements allocated to software, design constraints, resources, costs, and schedule need to be reflected in all changes to the plan.

2. The software development plan is updated to incorporate all new software project commitments and changes to commitments.
3. The software development plan is reviewed at each revision.
4. The software development plan is managed and controlled.

   "Managed and controlled" implies that the version of the work product in use at a given time (past or present) is known (i.e., version control) and changes are incorporated in a controlled manner (i.e., change control).

   If a greater degree of control than is implied by "managed and controlled" is desired, the work product can be placed under the full discipline of configuration management, as is described in the Software Configuration Management key process area.

A**ctivity 3** *Software project commitments and changes to commitments made to individuals and groups external to the organization are reviewed with senior management according to a documented procedure.*

**Activity 4** *Approved changes to commitments that affect the software project are communicated to the members of the software engineering group and other software-related groups.*

Examples of other software-related groups include:

- software quality assurance, and
- software configuration management.

**Activity 5** *The sizes of the software work products (or sizes of the changes to the software work products) are tracked, and corrective actions are taken as necessary.*

Refer to Activity 9 of the Software Project Planning key process area for practices covering derivation of size estimates.

1. Sizes for all major software work products (or the sizes of the changes are tracked.
2. Actual size of code (generated, fully tested, and delivered) is compared to the estimates documented in the software development plan.
3. Actual units of delivered documentation are compared to the estimates documented in the software development plan.
4. Overall projected size of the software work products (estimates, combined with actuals) is refined, monitored, and adjusted on a regular basis.
5. Changes in size estimates of the software work products that affect software commitments are negotiated with the affected groups and are documented.

**Activity 6** *The project's software effort and costs are tracked, and corrective actions are taken as necessary.*

Refer to Activity 10 of the Software Project Planning key process area for practices covering the derivation of cost estimates.

1. Actual expenditures of effort and costs over time and against work completed are compared to the estimates documented in the software development plan to identify potential overruns and underruns.
2. Software costs are tracked and compared to the estimates documented in the software development plan.
3. Effort and staffing are compared to the estimates documented in the software development plan.
4. Changes in staffing and other software costs that affect software commitments are negotiated with the affected groups and are documented.

**Activity 7** *The project's critical computer resources are tracked and corrective actions are taken as necessary.*

Refer to Activity 11 of the Software Project Planning key process area for practices covering the derivation of computer resource estimates.

1. The actual and projected uses of the project's critical computer resources

are tracked and compared to the estimates for each major software component as documented in the software development plan.
2. Changes in estimates of critical computer resources that affect software commitments are negotiated with the affected groups and are documented.

**Activity 8** The project's software schedule is tracked, and corrective actions are taken as necessary.

Refer to Activity 12 of the Software Project Planning key process area for practices covering derivation of the schedule.

1. Actual completion of software activities, milestones, and other commitments is compared against the software development plan.
2. Effects of late and early completion of software activities, milestones, and other commitments are evaluated for impacts on future activities and milestones.
3. Software schedule revisions that affect software commitments are negotiated with the affected groups and are documented.

Activity 9 *Software engineering technical activities are tracked, and corrective actions are taken as necessary.*

1. Members of the software engineering group report their technical status to their first-line manager on a regular basis.
2. Software release contents for successive builds are compared to the plans documented in the software development plan.
3. Problems identified in any of the software work products are reported and documented.
4. Problem reports are tracked to closure.

**Activity 10** *The software risks associated with cost, resource, schedule, and technical aspects of the project are tracked.*

Refer to Activity 13 of the Software Project Planning key process area for practices covering identification of risks.

1. The priorities of the risks and the contingencies for the risks are adjusted as additional information becomes available.
2. High-risk areas are reviewed with the project manager on a regular basis.

**Activity 11** *Actual measurement data and replanning data for the software project are recorded.*

Refer to Activity 15 of the Software Project Planning key process area for practices covering recording of project data.

1. Information recorded includes the estimates and associated information needed to reconstruct the estimates and verify their reasonableness.
2. The software replanning data are managed and controlled.
3. The software planning data, replanning data, and the actual measurement data are archived for use by ongoing and future projects.

**Activity 12** *The software engineering group conducts periodic internal reviews to track technical progress, plans, performance, and issues against the software development plan.*

These reviews are conducted between:

1. The project manager and software manager.

**Activity 13** *Formal reviews to address the accomplishments and results of the software project are conducted at selected project milestones according to a documented procedure.*

These reviews:

1. Are planned to occur at meaningful points in the software project's schedule, such as the beginning or completion of selected stages.
2. Are conducted with the customer, end user, and affected groups within the organization, as appropriate.

   The end users referred to in these practices are the customer designated end users or representatives of the end users.

3. Use materials that are reviewed and approved by the responsible software managers.
4. Address the commitments, plans, and status of the software activities.
5. Result in the identification and documentation of significant issues, action items, and decisions.
6. Address the software project risks.
7. Result in the refinement of the software development plan, as necessary.

**Measurement and Analysis**

**Measurement 1** *Measurements are made and used to determine the status of the software tracking and oversight activities.*

Examples of measurements include:

- effort and other resources expended in performing the tracking and oversight activities; and

- change activity for the software development plan, which includes changes to size estimates of the software work products, softare cost estimates, critical computer resource estimates, and schedule.

**Verifying Implementation**

**Verification 1** *The activities for software project tracking and oversight are reviewed with senior manager on a periodic basis.*

The primary purpose of periodic review by senior management is to provide awareness of, and insight into software process activities at an appropriate level og abstraction and in a timely manner. The time between reviews should meet the needs of the organization and may be lengthy, as long as adequate mechanisms for exception reporting are available.

1. The technical, cost, staffin, and schedule performance is reviewed.
2. Conflicts and issues not resolvable at lower levels are addressed.
3. Software project risks are addressed.
4. Action items are assigned, reviewed, and tracked to closure.
5. A summary status report from each meting is prepared and distributed to the affected groups.

**Verification 2** *The activities for software project tracking and oversight are reviewed with the project manager on both a periodic and event-driven basis.*

1. Affected groups are represented.
2. The technical, cost, staffing, and schedule performance is reviewed against the software development plan.
3. Use of critical computer resources is reviewed; current estimates and actual use of these critical computer resources are reported againt the original estimates.
4. Dependencies between groups are addressed.
5. Conflicts and issues not resolvable at lower levels are addressed.
6. Softare project risks are addressed.
7. Action items are assigned, reviewed, and tracked to closure.
8. A summary report from each meeting is prepared and distributed to the accefted groups.

**Verification 3** *The software quality assurance group review and/or audits the activities and work products for software project tracking and oversight and reports the results.*

Refer to the Software Quality Assurance key process area.

At a minimum, the reviews and/or autidts verify:

1. The actiities for reviewing and revising commitments.
2. Activities for revising the software development plan.
3. The content of the revised software development plan.
4. The activities for tracking the software project's cost, schedule, risks, technical and design constraints, and funtionality and performance.
5. The activities for conducting the planned technical and management reviews.

## Software Quality Assurance

The purpose of Software Quality Assurance is to provide management with appropriate visibility into the process being used by the software project and of the products being built.

Software Quality Assurance involves reviewing and auditing the software products and activities to verify that they comply with the applicable procedures and standards and providing the software project and other appropriate managers with the results of these reviews and audits.

The software quality assurance group works with the software project during its early stages to establish plans, standards, and procedures that will add value to the software project and satisfy the constraints of the project and the organization’s policies. By participating in establishing the plans, standards, and procedures, the software quality assurance group helps ensure they fit the project’s needs and verifies that they will be usable for performing reviews and audits throughout the software life cycle. The software quality assurance group reviews project activities and audits software work products throughout the life cycle and provides management with visibility as to whether the software project is adhering to its established plans, standards, and procedures.

Compliance issues are first addressed within the software project and resolved there if possible. For issues not resolvable within the software project, the software quality assurance group escalates the issue to an appropriate level of management for resolution.

This key process area covers the practices for the group performing the software quality assurance function. The practices identifying the specific activities and work products that the software quality assurance group reviews and/or audits are generally contained in the Verifying Implementation common feature of the other key process areas.

**Goals**

**Goal 1** *Software quality assurance activities are planned.*

**Goal 2** *Adherence of software products and activities to the applicable standards, procedures, and requirements is verified objectively.*

**Goal 3** *Affected groups and individuals are informed of software quality assurance activities and results.*

**Goal 4** *Noncompliance issues that cannot be resolved within the software project are addressed by senior manager.*

**Commitment to Perform**

**Commitment 1** *The project follows a written organizational policy for implementing software quality assurance (SQA).*

This policy typically specifies that:

1. The SQA function is in place on all software projects.
2. The SQA group has a reporting channel to senior management that is independent of:

    - the project manager,
    - the project's software engineering group, and
    - the software configuration management.

    Organizations must determine the organizational structure that will support activities that require independence, such as SQA, in the context of their strategic business goals and business environment.

    Independence should:

    - provide the individuals performing the SQA role with the organizational freedom to be the "eyes and ears" of senior management on the software project;
    - protect the individuals performing the SQA role from performance appraisal by the management of the software project they are reviewing; and
    - provide senior manager with confidence that object information on the process and products of the software project is being reported.

    The independence in XS means that SQA is either CSQA if the organization is developing solutions to specific customers and the customer has SQA function in the organization. Otherwise, the service of SQA is hired from external sources.

3. Senior manager periodically reviews the SQA activities and results.

## Ability to Perform

**Ability 1** *A group that is responsible for coordinating and implementing SQA for the project (i.e., the SQA group) exists, and is CSQA or external hired service.*

A group is the collection of departments, managers, and individuals who have responsibility for a set of tasks or activities. A group could vary from a single individual assigned part time, to several part-time individuals assigned from different departments, to several individuals dedicated full time. Considerations when implementing a group include assigned tasks or activities, the size of the project, the organizational structure, and the organizational culture. Some groups, such as the software quality assurance group, are focused on project activities, and others, such as the software engineering process group, are focused on organization-wide activities.

**Ability 2** *Adequate resources and funding are provided for performing the SQA activities.*

1. A manager is assigned specific responsibilities for the project's SQA activities.
2. A senior manager, who is knowledgeable in the SQA role and has the authority to take appropriate oversight actions, is designated to receive and act on software noncompliance items.
3. Tools to support the SQA activities are made available.

Examples of support tools include:

- workstations,
- database programs,
- spreadsheet programs, and
- auditing tools.

**Ability 3** *The members of the software project receive orientation on the role, responsibilities, authority, and value of the SQA group.*

## Activities Performed

**Activity 1** *A SQA plan is prepared for the software project according to a documented procedure.*

This procedure typically specifies that:

1. The SQA plan is developed in the early stages of, and in parallel with, the overall project planning.
2. The SQA plan is reviewed by the affected groups and individuals.

   Examples of affected groups and individuals include:
   - the software manager;
   - the project manager;
   - customer SQA representative (if applicable);
   - the senior manager, and
   - the software engineering group.

3. The SQA plan is managed and controlled.

   "Managed and controlled" implies that the version of the work products in use at a given time (past or present) is known (i.e., version control) and changes are incorporated in a controlled manner (i.e., change control).

   If a greater degree of control than is implied by "managed and controlled" is desired, the work product can be placed under the full discipline of configuration management, as is described in the Software Configuration Management key process area.

**Activity 2** *The SQA group's activities are performed in accordance with the SQA plan.*

The plan covers:

1. Responsibilities and authority of the SQA group.
2. Resource requirements for the SQA group (including staff, tools, and facilities).
3. Schedule and funding of the project's SQA group activities.
4. The SQA group's participation in establishing the software development plan, standards, and procedures for the project.
5. Evaluations to be performed by the SQA group.

   Examples of products and activities to be evaluated include:
   - operational software and support software,
   - deliverable and nondeliverable products,
   - software and nonsoftware products (e.g., documents),
   - product development and product verification activities (e.g., executing test cases), and
   - the activities followed in creating the product.

6. Audits and reviews to be conducted by the SQA group.

7. Project standards and procedures to be used as the basis for the SQA group's reviews and audits.
8. Procedures for documenting and tracking noncompliance issues to closure.

   These procedures may be included as part of the plan or may be included via reference to other documents where they are contained.

9. Documentation that the SQA group is required to produce.
10. Method and frequency of providing feedback to the software engineering group and other software-related groups on SQA activities.

**Activity 3** *The SQA group participates in the preparation and review of the project's software development plan, standards, and procedures.*

1. The SQA group provides consultation and review of the plans, standards, and procedures with regard to:

   - compliance to organizational policy,
   - compliance to externally imposed standards and requirements (e.g., standards required by the statement of work),
   - standards that are appropriate for use by the project,
   - topics that should be addressed in the software development plan, and
   - other areas as assigned by the project.

2. The SQA group verifies that plans, standards, and procedures are in place and can be used to review and audit the software project.

**Activity 4** *The SQA group reviews the software engineering activities to verify compliance.*

1. The activities are evaluated against the software development plan and the designated software standards and procedures.

   Refer to the Verifying Implementation common feature in the other key process areas for practices covering the specific reviews and audits performed by the SQA group.

2. Deviations are identified, documented, and tracked to closure.
3. Corrections are verified.

**Activity 5** *The SQA group audits designated software work products to verify compliance.*

1. The deliverable software products are evaluated before they are delivered

to the customer.
2. The software work products are evaluated against the designated software standards, procedures, and contractual requirements.
3. Deviations are identified, documented, and tracked to closure.
4. Corrections are verified.

**Activity 6** *The SQA group periodically reports the results of its activities to the software engineering group.*

**Activity 7** *Deviations identified in the software activities and software work products are documented and handled according to a documented procedure.*

This procedure typically specifies that:

1. Deviations from the software development plan and the designated project standards and procedures are documented and resolved with the appropriate software task leaders, software managers, or project manager, where possible.
2. Deviations from the software development plan and the designated project standards and procedures not resolvable with the software task leaders, software managers, or project manager are documented and presented to the senior manager designated to receive noncompliance items.
3. Noncompliance items presented to the senior manager are periodically reviewed until they are resolved.
4. The documentation of noncompliance items is managed and controlled.

**Activity 8** *The SQA group conducts periodic reviews of its activities and findings with the customer's SQA personnel, as appropriate.*

## Measurement and Analysis

**Measurement 1** *Measurements are made and used to determine the cost and schedule status of the SQA activities*.

Examples of measurements include:

- completions of milestones for the SQA activities compared to the plan;
- work completed, effort expended, and funds expended in the SQA activities compared to the plan; and
- numbers of product audits and activity reviews compared to the plan.

## Verifying Implementation

**Verification 1** *The SQA activities are reviewed with senior manager on a periodic basis.*

The primary purpose of periodic reviews by senior management is to provide awareness of and insight into software process activities at an appropriate level of abstraction and in a timely manner. The time between reviews should meet the needs of the organization and may be lengthy, as long as adequate mechanisms for exception reporting are available.

Refer to Verification 1 of the Software Project Tracking and Oversight key process area for practices covering the typical content of senior management oversight reviews.

**Verification 2** *The SQA activities are reviewed with the project manager on both a periodic and event-driven basis.*

Refer to Verification 2 of the Software Project Tracking and Oversight key process area for practices covering the typical content of project management oversight reviews.

## Software Configuration Management

The purpose of Software Configuration Management is to establish and maintain the integrity of the products of the software project throughout the project's software life cycle.

Software Configuration Management involves identifying the configuration of the software (i.e., selected software work products and their descriptions) at give points in time, systematically controlling changes to the configuration, and maintaining the integrity and traceability of the configuration throughout the software life cycle. The work products placed under software configuration management include the software products that are delivered to the customer (e.g., the software requirements document and the code) and the items that are identified with or required to create these software products (e.g., the compiler).

A software baseline library is established containing the software baselines as they are developed. Changes to baselines and the release of software products built from the software baseline library are systematically controlled via the change control and configuration auditing functions of software configuration management.

This key process area covers the practices for performing the software configuration management function. The practices identifying specific configuration items/units are contained in the key process area that describe the development and maintenance of each configuration item/unit.

**Goals**

**Goal 1** *Software configuration management activities are planned.*

**Goal 2** *Selected software work products are identified, controlled, and available.*

**Goal 3** *Changes to identified software work products are controlled.*

**Goal 4** *Affected groups and individuals are informed of the status and content of software baselines.*

**Commitment to Perform**

**Commitment 1** *The project follows a written organizational policy for implementing software configuration management (SCM).*

This policy typically specifies that:

1. Responsibility for SCM for each project is explicitly assigned
2. SCM is implemented throughout the project's life cycle.
3. SCM is implemented for externally deliverable software products, designated internal software work products, and designated support tools used inside the project (e.g., compilers).
4. The project establish or have access to a repository for storing configuration items/units and the associated SCM records.

   The contents of this repository are referred to as the "software baseline library" in these practices.

   The tools and procedures for accessing this repository are referred to as the "configuration management library system" in these practices.

   Work products that are placed under configuration management and treated as a single entity are referred to as configuration items.

   Configuration items are typically decomposed into configuration components, and configuration components are typically decomposed into units. In a hardware/software system, all of the software may be considered as a single configuration item, or the software may be decomposed into multiple configuration items. In these practices the term "configuration items/units" is used to refer to the elements under configuration management.

5. The software baselines and SCM activities are audited on a periodic

basis.

**Ability to Perform**

**Ability 1** *A board having the authority for managing the project's software baselines (SCM) exists or is established.*

The SCM:

1. Authorizes the establishment of software baselines and the identification of configuration items/units.
2. Represents the interests of the project manager and all groups who may be affected by changes to the software baselines.

   Examples of affected groups include:
   - software engineering,
   - system group,
   - system test group,
   - software quality assurance,
   - software configuration management, and
   - marketing and sales.

3. Reviews and authorizes changes to the software baselines.
4. Authorizes the creation of products from the software baseline library.

**Ability 2** *A group that is responsible for coordinating and implementing SCM for the project (i.e., the SCM group) exists.*

A group is the collection of departments, managers, and individuals who have responsibility for a set of tasks or activities. A group could vary from a single individual assigned part time, to several part-time individuals assigned from different departments, to several individuals dedicated full time. Considerations when implementing a group include assigned tasks or activities, the size of the project, the organizational structure, and the organizational culture. Some groups, such as the software quality assurance group, are focused on project activities, and others, such as the software engineering process group, are focused on organization-wide activities.

The SCM group coordinates or implements:

1. Creation and management of the project's software baseline library.
2. Development, maintenance, and distribution of the SCM plans, standards, and procedures.
3. The identification of the set of work products to be placed under SCM.

   A work product is any artifact from defining, maintaining, or using a

software process.

4. Management of the access to the software baseline library.
5. Updates of the software baselines.
6. Creation of products from the software baseline library.
7. Recording of SCM actions.
8. Production and distribution of SCM reports.

**Ability 3** *Adequate resources and funding are provided for performing the SCM activities.*

1. A manager is assigned specific responsibilities for SCM.
2. Tools to support the SCM activities are made available.

Examples of support tools include:

- workstations,
- database programs, and
- configuration management tools.

**Ability 4** *Members of the SCM group are trained in the objectives, procedures, and methods for performing their SCM activities*.

Examples of training include:

- SCM standards, procedures, and methods; and
- SCM tools.

**Ability 5** *Members of the software engineering group and other software-related groups are trained to perform their SCM activities*.

Examples of other software-related groups include:

- software quality assurance, and
- documentation support.

Examples of training include:

- the standards, procedures, and methods to be followed for SCM activities performed inside the software engineering group and other software-related groups; and
- the role, responsibilities, and authority of the SCM group.

## Activities Performed

**Activity 1** *A SCM plan is prepared for each software project according to a documented procedure*.

This procedure typically specifies that:

1. The SCM plan is developed in the early stages of, and in parallel with, the overall project planning.
2. The SCM plan is reviewed by the affected groups.
3. The SCM plan is managed and controlled.

   "Managed and controlled" implies that the version of the work products in use at a given time (past or present) is know (i.e., version control) and changes are incorporated in a controlled manner (i.e., change control).

   If a greater degree of control that is implied by "managed and controlled" is desired, the work product can be placed under the full discipline of configuration management, as is described in this key process area.

**Activity 2** *A documented and approved SCM plan in used as the basis for performing the SCM activities.*

The plan covers.

1. The SCM activities to be performed, the schedule of activities, the assigned responsibilities, and the resources required (including staff, tools, and computer facilities).
2. The SCM requirements and activities to be performed by the software engineering group and other software-related groups.

**Activity 3** *A configuration management library system is established as a repository for the software baselines.*

This library system:

1. Supports multiple control levels of SCM.

   Examples of situations leading to multiple levels of control include:
   - differences in the levels of control needed at different times in the life cycle (e.g., tighter control as product matures).
   - differences in the levels of control needed for software-only systems vs. systems which include both hardware and software.

2. Provides for the storage and retrieval of configuration items/units.
3. Provides for the sharing and transfer of configuration items/units between the affected groups and between control levels within the library.
4. Helps in the use of product standards for configuration items/units.
5. Provides for the storage and recovery of archive versions of configuration items/units.
6. Helps to ensure correct creation of products from the software baseline library.
7. Provides for the storage, update, and retrieval of SCM records.

8. Supports production of SCM reports.
9. Provides for the maintenance of the library structure and contents.

Examples of library maintenance functions include:
- backup/restoring of library files, and
- recovery from library errors.

**Activity 4** *The software work products to be placed under configuration management are identified.*

1. The configuration items/units are selected based on documented criteria.

Examples of software work products that may be identified as configuration items/units include:
- process-related documentation (e.g., plans, standards, or procedures),
- software requirements,
- software design,
- software code units,
- software test procedures,
- software system build for the software test activity,
- software system build for delivery to the customer or end users,
- compilers, and
- other support tools.

2. The configuration items/units are assigned unique identifiers.
3. The characteristics of each configuration item/unit are specified.
4. The software baselines to which each configuration item/unit belongs are specified.
5. The point in its development that each configuration item/unit is placed under configuration management is specified.
6. The person responsible for each configuration item/unit (i.e., the owner, from a configuration management point of view) is identified.

**Activity 5** Change requests and problem reports for all configuration items/units are initiated, recorded, reviewed, approved, and tracked according to a documented procedure.

**Activity 6** *Changes to baselines are controlled according to a documented procedure.*

This procedure typically specifies that:

1. Reviews and/or regression tests are performed to ensure that changes have not caused unintended effects on the baseline.
2. Only configuration items/units that are approved by the SCM are entered into the software baseline library.
3. Configuration items/units are checked in and out in a manner that maintains

the correctness and integrity of the software baseline library.

Examples of check-in/out steps include:

- verifying that the revisions are authorized,
- creating a change log,
- maintaining a copy of the changes,
- updating the software baseline library, and
- archiving the replaced software baseline.

**Activity 7** *Products from the software baseline library are created and their release is controlled according to a documented procedure.*

This procedure typically specifies that:

1. The SCM authorizes the creation of products from the software baseline library.
2. Products from the software baseline library, for both internal and external use, are built only from configuration items/units in the software baseline library.

**Activity 8** *The status of configuration items/units is recording according to a documented procedure.*

This procedure typically specifies that:

1. The configuration management actions are recorded in sufficient detail so that the content and status of each configuration item/unit are known and previous versions can be recovered.
2. The current status and history (i.e., changes and other actions) of each configuration item/unit are maintained.

**Activity 9** *Standard reports documenting the SCM activities and the contents of the software baseline are developed and made available to affected groups and individuals.*

Examples of reports include:

- SCM meeting minutes,
- change request summary and status,
- trouble report summary and status (including fixes),
- summary of changes made to the software baselines,
- revision history of configuration items/units,
- software baseline status, and
- results of software baseline audits.

**Activity 10** *Software baseline audits are conducted according to a documented procedure.*

This procedure typically specifies that:

1. There is adequate preparation for the audit.
2. The integrity of software baselines is assessed.
3. The structure and facilities of the configuration management library system are reviewed.
4. The completeness and correctness of the software baseline library contents are verified.
5. Compliance with applicable SCM standards and procedures is verified.
6. The results of the audit are reported to the project software manager.
7. Action items from the audit are tracked to closure.

## Measurement and Analysis

**Measurement 1** *Measurements are made and used to determine the status of the SCM activities.*

Examples of measurements include:

- number of change requests processed per unit time;
- completions of milestones for the SCM activities compared to the plan; and
- work completed, effort expended, and funds expended in the SCM activities.

## Verifying Implementation

**Verification 1** *The SCM activities are reviewed with senior manager on a periodic basis.*

The primary purpose of periodic reviews by senior management is to provide awareness of and insight into software process activities at an appropriate level of abstraction and in a timely manner. The time between reviews should meet the needs of the organization and may be lengthy, as long as adequate mechanisms for exception reporting are available.

Refer to Verification 1 of the Software Project Tracking and Oversight key process area for practices covering the typical content of senior management oversight reviews.

**Verification 2** *The SCM activities are reviewed with the project manager on both a periodic and event-driven basis.*

Refer to Verification 2 of the software Project Tracking and Oversight key process area for practices covering the typical content of project management oversight reviews.

**Verification 3** *The SCM group periodically audits software baseline to verify that they conform to the documentation that defines them.*

**Verification 4** *The software quality assurance group reviews and/or audits the activities and work products for SCM and reports the results.*

Refer to the Software Quality Assurance key process area.

At a minimum, the reviews and/or audits verify:

1. Compliance with the SCM standards and procedures by:

   - the SCM group,
   - the software engineering group, and
   - other software-related groups.

2. Occurrence of periodic software baseline audits.

# 6 Concluding Remarks

We have presented a CMM Level 2 for XS organizations.

The presentation of CMM for eXtra Small organizations follows the format of the software CMM, being easy to find specifics about, but having the shortcoming of difficulties in getting oversight. For that reason, other formats should be provided, e.g. role by role descriptions of the responsibilities and less formalised tasks, e.g. participation in groups for negotiations and the like. The flowgraph technique might also be useful, providing a picture of information flowing between the roles. One graph would get too complex, making the oversight to be lost. One graph for each KPA could be a reasonable choice. It is important to model the dynamics in the KPAs. Although the dynamics is implicit in the text format, it is difficult to get an overview picture of what is happening when applying CMM to software development.

## 6.1 Future Work

Other formats of presentation of the model are needed, e.g. the diagram of the activities in the KPAs. Further, the model as presented in this paper, does not include the growing needs of the organisation. To extend the model to include advice on the way when growing is the task of future work.

## References


[1] Caputo K, Implementation Guide – Choreographing Software Process Improvement. Addison-Wesley, 1998.

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

[3] Orci, T, Capability Maturity Model for Extra Extra Small Organizations, Level 2. Umeå University, Dept of Computing Sciences, UMINF 00.12, 2000.

[4] Orci, T, Capability Maturity Model for Small Organizations, Level 2. Umeå University, Dept of Computing Sciences, UMINF 00.14, 2000.

[5] Laryd, A, Orci, T, Dynamic Capability Maturity Model for Small Organizations, in Proceedings for ASSE2000, Tandil, Argentina, 2000.

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

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

[8] Van Solingen R, Berghout E, The Goal/Question/Metric Method. A Practical Guide for Quality Improvement of Software Development. McGraw Hill, 1999.

[9] http://www.sei.cmu.edu/

[10] http://www.esi.es/Projects/SPICE


## Appendix A Abbreviations

| | |
|---|---|
| ABB | Asea Brown Boveri |
| CMM | Capability Maturity Model |
| GQM | Goal-Question-Metric |
| KPA | Key Process Area |
| MS | Marketing and Sales |
| PM | Project Manager |
| RM | Requirements Management |
| ROI | Return On Investment |
| SCCB | Software Configuration Control Board |
| SCM | Software Configuration Management |
| SCMG | Software Configuration Management Group |
| SE | Software Engineer |
| SM | Senior Management |

| | |
|---|---|
| SME | Small and Medium sized Enterprise |
| SPP | Software project Planning |
| SPTO | Software project Tracking and Oversight |
| SPICE | Software Process Improvement and Capability dEtermination |
| SWM | Software Manager |
| SQA | Software Quality Assurance |
| SSM | Software Subcontract Management |
| STG | Software Test Group |
| SG | System engineering Group |
| S | Small enterprise |
| XXS | eXtra eXtra Small enterprise |
| XS | eXtra Small enterprise |