Software Process Improvement and the Capability Maturity Model

lic. Marc De Smet

The amount of embedded software is increasing rapidly, both in size per product and in the number of products with embedded software. The current software development processes cannot keep up with this growth and are not able to deliver the necessary reliability required for such products. In this article, Marc De Smet explains how such software development processes can be improved dramatically, which role the Capability Maturity Model can play in these improvements, and how this model compares to ISO 9000.

Software Process Improvement

Lots of software development departments world-wide are trying to improve their software capabilities. A Software Process Improvement (SPI) approach as shown in Figure 1 can be followed.

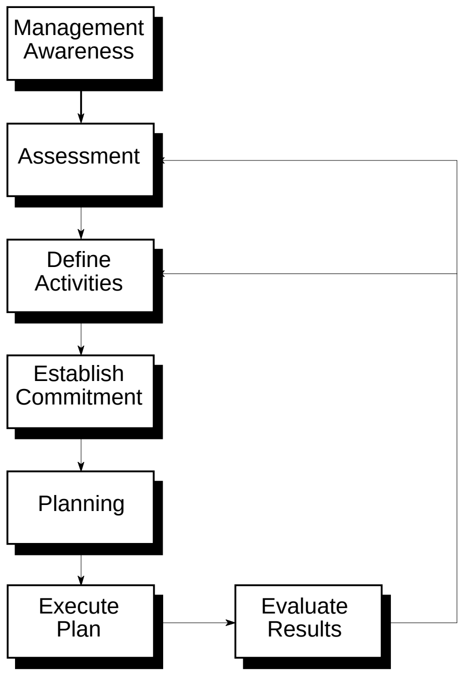

Figure 1: Approach for Software Process Improvement.

A SPI effort is only started when senior management of the organization recognizes the seriousness of the software development problem and agrees to invest sufficiently in SPI. Then an SPI program that will go through a loop several times, is started.

The loop starts with an assessment. During the assessment, a trained team of software professionals interview people in the software development department and all the other departments that have an interface with software development. The assessors also look into available documentation (e.g., plans, specifications, designs). The resulting assessment report contains the following main chapters.

- *Current situation*
  This chapter describes the current software development process.
- *Strengths and weaknesses*
  This chapter describes the strengths in the current development process that can serve as a basis for further improvement, and the weaknesses that result in risks.
- *Recommendations*
  This chapter typically offers approximately seven of the highest priority improvement recommendations.

Based on the recommendations a SPI team can define the activities necessary to realize the recommendations. For each activity an effort estimate is also given so that the total cost of the first loop in SPI is known.

Management can now decide on a final budget for the SPI program. Typical budgets are 10-15%

of the software developing capacity (for small departments). Knowing the budget, the SPI team can schedule the activities in a Short Term SPI Plan (STP). Such an STP typically covers one year. During the execution of the STP, both the STP activities and the consequences of the execution (in terms of shorter lead times, lower development costs, higher quality) should be tracked. This tracking is done by means of normal project management tracking techniques, and periodical intermediate assessments. Intermediate assessments may lead to readjustments in the program, as depicted by the inner loop of Figure 1. Usually two to three years are necessary to implement all the recommendations of the assessment. This brings us back in the outer loop in Figure 1.

The Capability Maturity Model

The Capability Maturity Model (CMM) has been developed by the Software Engineering Institute (SEI) at Carnegie Mellon University in Pittsburg. It is an implementation of Total Quality Management for embedded software development. Although the CMM has been developed to support SPI for embedded software, it can also be used to support SPI in other environments, such as the development of business software, hardware, etc.

According to the CMM, a software development organization is at one of five levels of maturity, as shown in Figure 2.

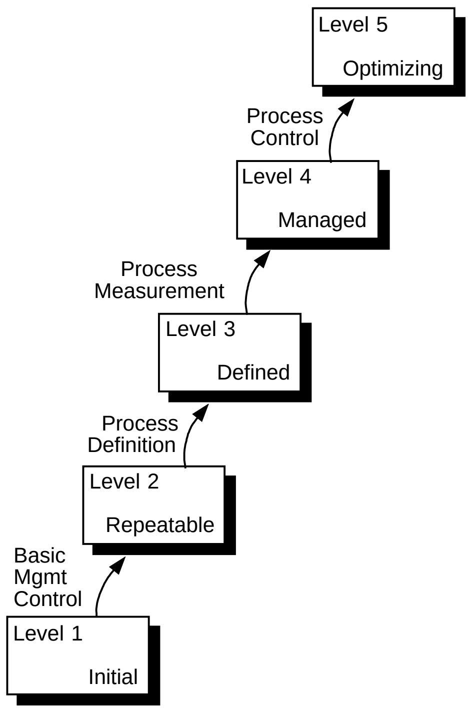

Figure 2: The five levels of the CMM.

Level 1: Initial
Until the process is under statistical control, orderly progress in process improvement is not possible. While there are many degrees of statistical control, the first step is to achieve rudimentary predictability of schedules and costs. Projects in Level 1 organizations are usually done in a rather ad hoc and chaotic way. Cost, lead time, and quality are unpredictable. These organizations can produce good software, but usually at a high price. Quality is completely dependent on the capabilities of individuals. Such organizations should first focus on basic management control issues such as project management, configuration management, and software quality assurance.

Level 2: Repeatable
The organization has achieved a stable process with a repeatable level of statistical control by initiating rigorous project management of commitments, costs, schedules, and changes. Projects can use prior experience in doing familiar work. The development process, however, is still completely dependent on individuals. Cost and quality are still predicted in an unreliable way. Predictions about lead time become reliable. Attention should now be given to define the development process, training, and technical practices.

Level 3: Defined
The organization has defined the process as a basis for consistent implementation and better understanding. The process definition, however,

is still qualitative definition. Cost and lead time can be reliably predicted. Quality predictions, however, remain unreliable. At this point advanced technology can usefully be introduced. Attention should be focused on quantifying the process.

Level 4: Managed
The organization has initiated comprehensive process measurements and analysis. The process is now quantified. Also, quality can be reliably predicted. This is when the most significant quality improvements begin. Attention should here be focused on continuous improvement through defect prevention and quantifiable technology changes.

Level 5: Optimizing
The organization now has a foundation for continuing improvement and optimization of the process. Cost, lead time, and quality continuously improve.

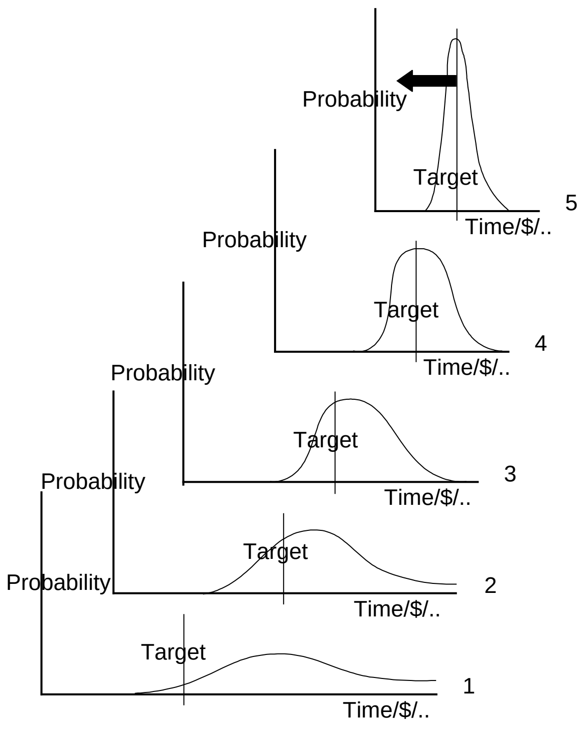

Figure 3: Process capability per maturity level.

Advantages of higher CMM levels

The main advantages for organizations operating at the higher levels of the CMM are related to the following issues.

Process capability & performance prediction
The maturity of an organization's software development process helps to predict a project's ability to achieve its goals. Projects in Level 1 organizations experience wide variations in achieving cost, schedule, functionality, and quality targets.
As illustrated in Figure 3, three improvements in meeting target goals are observed as the organization's software process matures.

- The difference between target results and actual results decreases across projects
- The variability of actual results around target results decreases
- Target results improve, i.e., costs decrease, development time becomes shorter, and productivity and quality increase.

Visibility of the software process
People outside the project lack insight into the project's process. Figure 4 illustrates the level of visibility into project status and performance afforded to management at each level of the process maturity.

At level 1 the software process is a black box. Managers have an extremely difficult time establishing the status of the project's progress and activities.
At level 2 the basic management controls which have been installed offer visibility into the project on defined occasions. The process can be seen as a succession of black boxes offering management visibility at transition points (milestones).
At level 3 the internal structure of the boxes is visible. The internal structure represents the way the organization's standard software development process is being applied.
At level 4 managers are able to measure progress and problems. They have an objective, quantitative basis for making decisions.

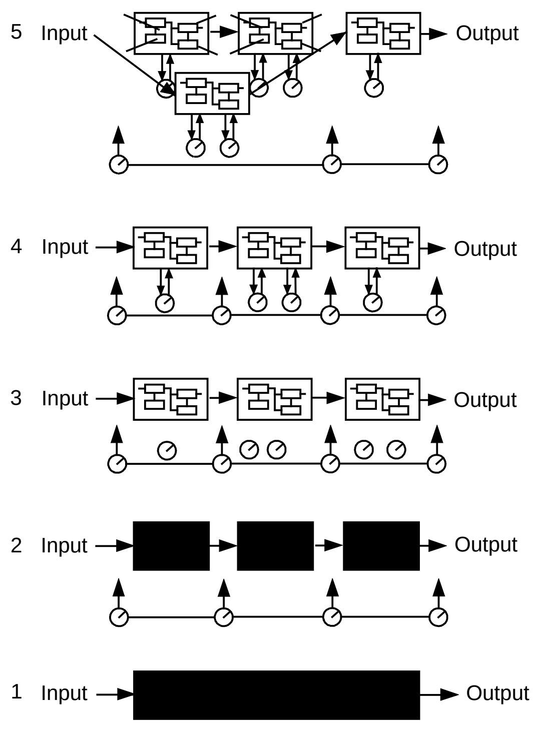

Figure 4: Management's visibility into the software process.

At level 5, finally, new and improved ways of building the software are continually tried in a controlled manner.

Incurred risk
As an organization's software process matures, the productivity and quality of its projects increase. These increases result in a proportional decrease of project risk. Since risk also increases when the project size increases, it is essential to enhance the organization's maturity before it starts to tackle large projects.

Increasing quality
Typically the quality of the products developed increases dramatically. The number of post-release defects per 1,000,000 lines of code can drop from approximately 10,000 in Level 1 organizations to zero defect development in Level 5 organizations.

Comparison CMM with ISO 9000

ISO 9000 differs from CMM in the following issues.

- *Levelling*
  ISO 9000-3 does not have different levels, whereas the CMM recognizes five well-defined levels.

- *Expression*
  ISO 9000 is the specification of a set of requirements (standards to satisfy, prescriptions), whereas large parts of the CMM are based on process descriptions. ISO 9000 stresses what is done, CMM how it is done.

- *Orientation*
  ISO 9000 is organization oriented and aims to ensure that the organization is able to operate in a quality-assuring way. The CMM is process and project oriented in that it aims to improve the development process.

- *Emphasis*
  ISO 9000 requires to define an organization that can ensure that it operates according to the prescribed way of working (quality planning and control). The CMM focuses on continuous improvement.

- *Scope*
  The CMM is concerned solely with software development, and indirectly with maintenance of the software. ISO 9000 is equally concerned with development, supply, and maintenance. ISO, on the other hand, does not focus on software. Even the ISO 9000-3 guidelines for the implementation of ISO 9000 for software, is not as concrete and practical as the CMM.

- *Commitment*
  ISO 9000 emphasizes the customer-supplier relationship, the external view focused on the commitment between the supplier and the customer. The CMM view is based on internal commitments between departments

(e.g., between software and hardware development departments) within the supplier's organization.

- *Measurement*
  Both models stress the importance of monitoring and verification of the software process. Within the CMM, however, the need for quantitative information is explicitly stated, being aimed at establishing statistical process control.

- *Process/technology view*
  Both models cover process control as well as technology control of software development. But in ISO 9000-3 these two aspects are not explicitly separated. The CMM explicitly separates the two issues, but covers the process issue more fully than the technology issue.

A comparison between ISO 9000 and CMM also reveals the following relationships.

- An organization at level 3 of the CMM should normally have no difficulty in acquiring an ISO 9000 certificate.

- Since the certifying authorities for ISO 9000 are often somewhat lacking in software knowledge, the fact that an organization has acquired the ISO certificate does not automatically mean that it is operating at level 3 of the CMM.

Other SPI-related efforts

Some consulting companies have designed their own models to support SPI. In the meanwhile, the SEI has not rested on its laurels. After CMM 1.0 (1991), it has released CMM 1.1 (1993) and is working on CMM 2.0 (due in 1996). The SEI has also released the People Management Capability Maturity Model (PM-CMM), and is working on the System Capability Maturity Model (S-CMM). □

Literature

An introduction to the CMM is given in
Mark C. Paulk, Bill Curtis, Mary Beth Chrissis, Charles V. Weber,
Capability Maturity Model for Software, Version 1.1,
Software Engineering Institute, Carnegie Mellon University, CMU/SEI-93-TR-24

The full description of the model is found in
Mark C. Paulk, Charles V. Weber, Suzanne M. Garcia, Marybeth Chrissis, Marilyn Bush,
Key Practices of the Capability Maturity Model for Software, Version 1.1,
Software Engineering Institute, Carnegie Mellon University, CMU/SEI-93-TR-25

Pasfoto De Smet

Lic. Marc De Smet is a consultant at Management Information Systems in Zoersel (Belgium). He has been involved in SPI activities in several European and Asian countries.