

Introduction to Software Process Improvement

Introduction

The crisis in software has been well documented. In order to compete successfully in the international market, we, as software professionals, need to improve both the quality of our software products and our ability to work within time and budget constraints. These improvements depend strongly on process as well as technology.

Modern technology can help us combat the software crisis, yet as Fred Brooks warns us, there is no technological “silver bullet” to rescue us. Talented people are important in any software organization. Nevertheless, people need to be supported by a good working environment. Software development is hampered by changing requirements, unpredictable schedules, lack of standards, and insufficient training more than by a lack of effort on the part of professionals. Curtis, Krasner, and Iscoe documented these issues effectively in their report describing their interviews with software professionals. In short, problems with the process, rather than the technology, cause a substantial number of the problems in software development and maintenance.

This brief document begins with a history of the Software Engineering Institute (SEI) and its software process work. Terminology is introduced and the five levels of the SEI *capability maturity model* (CMM) are defined. Possible uses and future directions of the model are provided.

Background and Definitions

The Software Engineering Institute was established at Carnegie Mellon University in Pittsburgh, Pennsylvania in 1984, under a Department of Defense contract. Its mission is to provide leadership in advancing the state of the practice of software engineering to improve the quality of systems that depend on software. The software process work began two years later. One of the results was a *software process maturity model*. In 1987, the SEI and MITRE Corporation produced the first *maturity questionnaire*, a set of yes-no questions that address organization and management issues, as well as the technical software development process. Over the next few years, the SEI developed two methods for using the questionnaire to appraise an organization’s software process.

This document is taken from the SEI educational materials package “Lecture Notes on Software Process Improvement” by Laurie Honour Werth, document number CMU/SEI-93-EM-8, copyright 1993 by Carnegie Mellon University. Permission is granted to make and distribute copies for noncommercial purposes.

After an extensive review process, the *capability maturity model* (CMM) for software replaced the software process maturity model in 1991. The CMM summarizes general software process practices for each of five maturity levels. Once the current level of operation is established using the maturity questionnaire, improving a company's software process involves implementing the software engineering practices needed to reach each of the five levels, in order, from lowest to highest.

What is Software Process and How Can It be Improved?

It is important to understand the vocabulary used in describing the software process and the maturity model. These terms are used in a particular way, and it is important to know their meaning in order to comprehend the model. It is also necessary to understand that a software process model is not a mathematical formula. In this context, it is a description of how to conduct the process of software development.

Software process is defined as a set of activities that begin with the identification of a need and concludes with the retirement of a product that satisfies the need; or more completely, as a set of activities, methods, practices, and transformations that people use to develop and maintain software and its associated products (e.g., project plans, design documents, code, test cases, user manuals).

Software process capability describes the range of expected results achieved from a software process. But capability is not the same as performance. Software process performance is the actual results achieved from following a software process. That is, results achieved (performance) differ from results expected (capability).

Many software process techniques, such as quality assurance, configuration management, inspections, and reviews, are described in most software engineering textbooks and will not be covered here. From a general problem-solving point of view, project management may be described simply as:

- identifying what is to be done
- deciding how to do it
- monitoring what is being done
- evaluating the outcome

Most managers try to do the first and second steps above, describing the *what* and *how* using plans and schedules. However, even though managers know that requirements will change, schedules will slip, and all the other typical problems will likely occur on the project, few attempt to build these dynamic events into their plans. In addition, managers need to monitor project activities and to adjust the plans as modifications occur. To improve the software development process, it is necessary to evaluate the success of the project and avoid repeating problems in the future. The CMM addresses these latter, less well-understood, issues in an effort to improve the software development process.

The scientific, closed-loop model of management can be explained most simply as follows. Project plans are used as hypotheses, and project results are evaluated to verify or validate these hypotheses. Statistical, or closed-loop, process management is based on measurement. First a baseline is determined. After improvements are instituted, measurements are repeated. Results are compared against the hypotheses or predictions to measure progress. This process comparison is repeated with the goal of reducing the differences between the predicted results and the actual results. Thus, the project is managed, but the management process is meta-managed.

W. E. Deming, one of the pioneers of applying statistical process control in industry, describes process improvement as a continuous cycle which follows these steps:

1. Understand the status of the development process.
2. Develop a vision of the desired process.
3. List improvement actions in priority order.
4. Generate a plan to accomplish the required actions.
5. Commit the resources to execute the plan.
6. Start over at step 1.

SEI Capability Maturity Model

The SEI capability maturity model is derived from the ideas of quality improvement applied to software development. The five-level improvement model is shown in Figure 1. The five stages are called maturity levels. Each represents an improvement in the software process. An organization's software capability can be improved by advancing through these five stages or levels. The CMM helps organizations to select improvement strategies based on current process maturity status and to identify critical issues in quality and process improvement.

The following descriptions outline the primary characteristics of the software process for each level of the CMM.

1. Initial

The initial software process is characterized as ad hoc. Typically, the organization operates without formal procedures, cost estimates or project plans. There are few mechanisms to ensure that procedures are followed. Tools, if they exist, are not well integrated. Change control is lax or nonexistent. Senior management neither hears about nor understands software problems and issues. Success generally depends on the efforts of individuals, not the organization. Not too surprisingly, most software organizations—86% in an early survey—were at the initial level.

2. Repeatable

At the repeatable level, project controls have been established over quality assurance, change control, cost, and schedule. This discipline enables earlier successes to

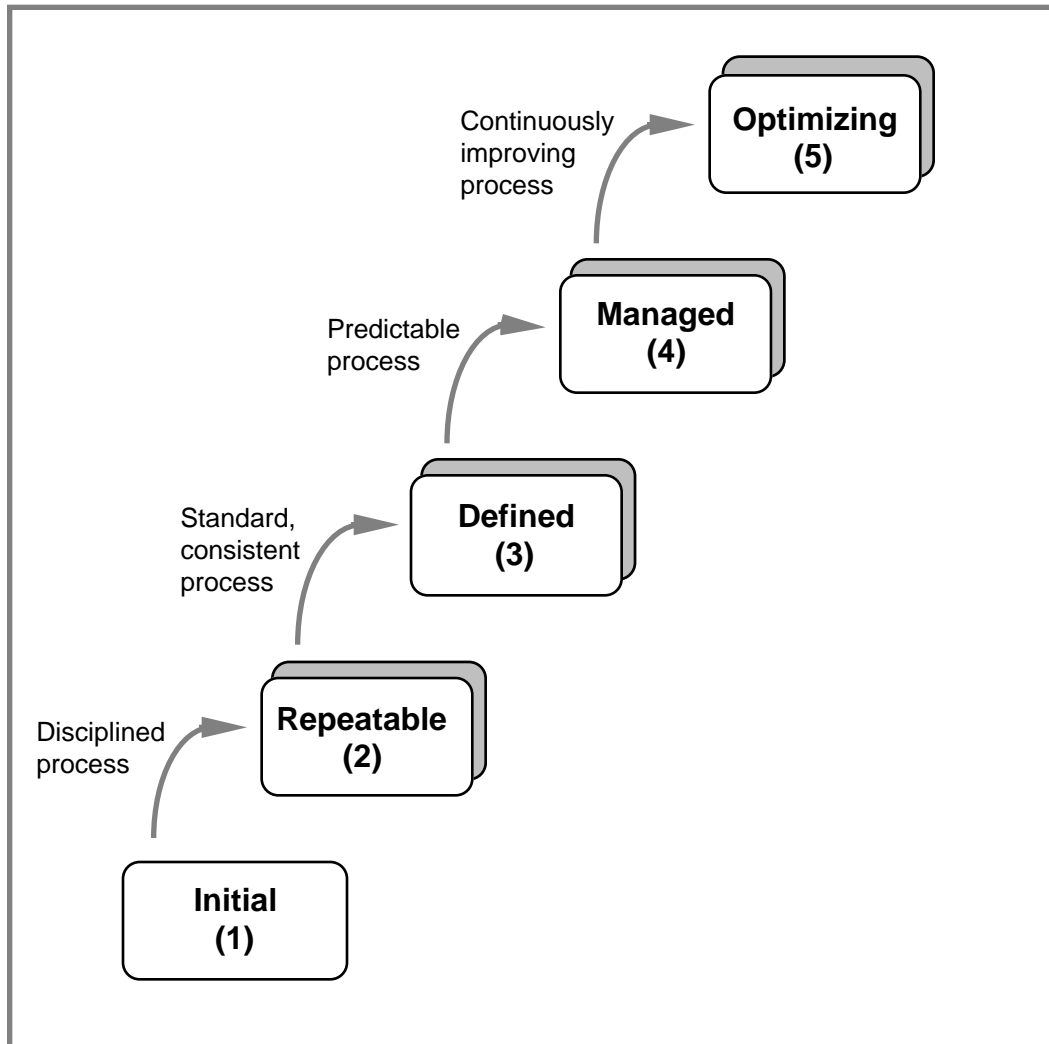


Figure 1. The five levels of software process maturity

be repeated, though the organization may have problems applying these techniques to new applications. An SEI survey found 16% of the companies at the repeatable level.

3. Defined

The defined software process, for both management and engineering activities, is documented, standardized, and integrated across the organization. The process is visible; that is, it can be examined and improvements suggested. Typically, the organization has established a *software engineering process group* (SEPG) to lead the improvement effort, keep management informed on progress, and facilitate introducing other software engineering methods.

An early SEI study found only one percent of organizations surveyed at the defined level, though more recently several large companies have achieved this stage for

some of their development groups. Organizations need to achieve the defined maturity level to consistently produce quality software on time and within budget.

4. Managed

Achieving the fourth, or managed, level requires that measures of software process and product quality be collected so that process effectiveness can be determined quantitatively. A process database and adequate resources are needed to continually plan, implement, and track process improvements.

5. Optimizing

At the optimizing level, quantitative feedback data from the process allows continuous process improvement. Data gathering has been partially automated. Management has changed its emphasis from product maintenance to process analysis and improvement. Defect cause analysis and defect prevention are the most important activities added at this level.

As maturity increases, differences between targeted and actual results decrease; costs decrease and development time shortens; and productivity and quality increase. The process becomes more predictable as rework is prevented and the risk level is reduced. See Table 1 for the specific process areas added at each maturity level. It is important to note that maturity levels cannot be skipped as each level is the foundation for the next.

Progress through the maturity levels requires high-level management backing and long-term commitment. It requires fundamental modifications in the way managers and software practitioners do their jobs, and this takes time to accomplish. Software process improvement should not even be started without considerable levels of corporate-wide encouragement and support.

One SEI report summarizes the capability maturity model in this way.

“The CMM provides a conceptual structure for improving the management and development of software products in a disciplined and consistent way. It does not guarantee that software products will be successfully built or that all problems in software engineering will be resolved. The CMM identifies practices for a mature software process, but it is not meant to be either exhaustive or dictatorial. While the maturity questionnaire samples key indicators of an effect, software process and the CMM identifies the characteristics of an effective software process, the mature organization addresses all issues essential to a successful project—including people and technology—as well as process.”

Level 2:	Repeatable
	Requirements Management
	Software Project Planning
	Software Project Tracking and Oversight
	Software Subcontract Management, if applicable
	Software Quality Assurance
	Software Configuration Management
Level 3:	Defined
	Organization Process Focus
	Organization Process Definition
	Training Program
	Integrated Software Management
	Software Product Engineering
	Intergroup Coordination
	Peer Reviews
Level 4:	Managed
	Process Measurement and Analysis
	Quality Management
Level 5:	Optimizing
	Defect Prevention
	Technology Innovation
	Process Change Management

Table 1. Key process areas (KPA) by maturity level

How Is the Maturity Model Used?

Several companies have already reported benefits from applying the capability maturity model. A process improvement investment of \$400,000 produced an annual savings of \$2,000,000 at Hughes Aircraft. Raytheon saved about \$9.2 million by eliminating rework on a base of about \$115 million in software development costs. At IBM Houston, the NASA space shuttle on-board software development group showed the results depicted in Table 2.

There are two major ways the maturity model can be applied: for software process assessment (SPA) and for software capability evaluation (SCE). Both methods are based on the capability maturity model and maturity questionnaire. Together, the model and questionnaire provide a way to identify and compare organizations' strengths

NASA	1982	1985
Early error detection (% errors found)	48	80
Reconfiguration time (weeks)	11	5
Reconfiguration effort (person-years)	10.5	4.5
Product error rate (errors per 1000 lines of code)	2.0	0.11

Table 2. On-board shuttle software improvements

and weaknesses. Figure 2 shows, at a high level, the common steps in the SPA and SCE methods.

There are differences between the two methods, SPA and SCE, as summarized in Table 3. These differences include the objectives, the make-up of the site visitation teams, the criteria for determining scope and for defining findings, and the ownership of the results. In view of these differences, the outcomes of assessments and evaluations are not likely to be interchangeable or directly comparable. The SCE team may conclude that a particular weakness has low risk associated with it for the acquisition and therefore discount it (for example, subcontract management in an acquisition that may not involve subcontracts), while a SPA team might recommend corrective action as a high priority for the same weakness (since other projects involve subcontractors). Both views would be valid for their respective purposes.

Software Process Assessment

A software process assessment is a means for organizations to identify their strengths, weaknesses, existing improvement activities, and key areas for improvement. It enables them to determine the current state of their software process and to develop action plans for improvement.

Assessments are performed by a team of 6-10 experienced software professionals. The majority are from the organization being assessed. In the past, individual teams have been trained by the SEI. More recently, the SEI has trained and licensed SPA associates to provide these services commercially. A SPA associate often works collaboratively with team members. In selected cases, the SEI does so.

An organization spends two to six months (elapsed time) preparing for an assessment, beginning with management commitment to the process improvement effort. Other preparations include selecting the assessment team, and selecting the representatives of software projects and functional areas (such as testing and quality assurance) who will participate in the on-site assessment activities.

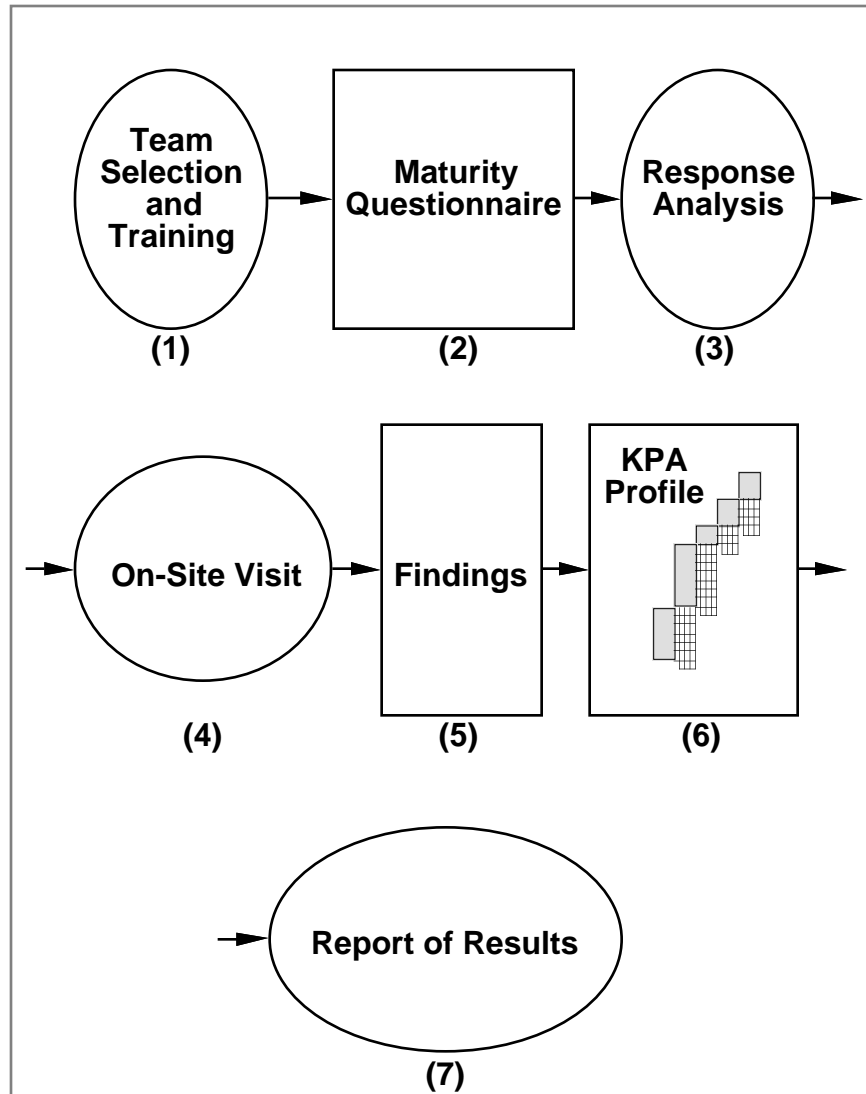


Figure 2. Common steps in SPAs and SCEs

The assessment team helps to prepare the rest of the organization for the assessment and for the ensuing process improvement activities. In addition to making detailed plans and a schedule, they tell everyone what to expect and concentrate on building support for process improvement and the assessment. The team spends five days on site.

They analyze the information they get from the maturity questionnaire, individual interviews with project leaders, and group discussions with practitioners (functional area representatives). To encourage interviewees to be open, the SPA team treats the information as confidential; they report only composite findings and give no attribution to individuals or projects. Occasionally, the team looks at documents to clarify information from the discussions.

SCE	SPA
Used by acquisition organization for source selection and contract monitoring	Used by organization to improve software process
Results to the organization and the acquirer	Results to organization only
Substantiates current practice	Assesses current practice
Assesses commitment to improve	Acts as catalyst for process improvement
Analyzes contract performance potential	Provides input to improvement action plan
Independent evaluation—no organization members on team	Collaborative—organization members on team, with representative from licensed SPA associate or SEI
Applies to performance for a particular contract	Applies to organization overall, not individual projects

Table 3. Comparison between SCE and SPA

The team uses the maturity model to help them categorize data. They determine the current software process maturity, identify key findings (that is, they determine what will impede capability to produce quality software), and note strengths the organization can build upon. Decisions are made by consensus, which helps to counter possible bias in any individual's conclusions about the data. In addition, project leaders and practitioners who were interviewed review the team's findings and give feedback.

The team presents the results of the assessment to senior management and, often, to the entire organization that was assessed. They recommend what can be done to address their findings. Many teams brainstorm a preliminary list of recommendations and enlist others in the organization to help flesh it out. Later, the team writes a final report that presents findings and recommendations in detail. The SEI recommends that the report be widely available in the organization.

After the assessment, an action planning group (often a software engineering process group, under the guidance of a management steering committee) develops the strategies for accomplishing long-term process improvement and determines what improvements are achievable within a specific time frame. They work with many others in the organi-

zation to create an action plan and implement it. To be effective, they must combine improvement plans with other organizational plans such as the business plan. Members of the SPA team often become involved in these activities; their involvement is one of the significant differences between a process assessment and a capability evaluation.

As an organization's software process matures, periodic reassessments allow them to identify new priorities and strategies for further improvement. The SEI recommends reassessments approximately every two years.

Software Capability Evaluation

A software capability evaluation is an independent evaluation of an organization's software process as it relates to a particular acquisition. It is a tool that helps an external group (an "acquirer") determine the organization's ability to produce a particular product having high quality and to produce it on time and within budget.

Evaluations are performed by a trained and experienced team of 4-6 members. The members are not part of the organization being evaluated. Rather, they come from acquisition organizations, such as certain government agencies.

An SCE team looks at projects that perform work similar to that required by the new contract. Thus, the selected projects are usually similar to the acquisition in application domain, size, and life-cycle phases. They may not be representative of the organization as a whole.

During a planning period of several weeks, the organization being evaluated submits a choice of projects for selection, along with information about the projects and about the organization in general. The SCE team selects projects, reviews high-level documents from the organization, and makes detailed plans and a schedule for the site visit.

The SCE team spends three days on site interviewing project and organization personnel, one at a time. Team members may talk to as few as 10 or as many as 30 people. They also review documents from one to four projects, depending on the particular application of the SCE method (contract monitoring or source selection). As in the SPA method, the SCE team does not attribute information to individuals but holds their responses in confidence.

While still on site, the team identifies strengths, weaknesses, and improvement activities that they consider to be most relevant to performance on the acquisition contract. Consensus among the team plays a major role in countering possible bias in the way any individual might form conclusions about the data. The team does not assign a maturity level, but rather builds a profile of strengths and weaknesses relevant to the specific acquisition.

The SCE team presents their findings to the organization that was evaluated. To foster process improvement, the SEI strongly recommends that the presentation include all the detailed SCE findings that are delivered to the acquisition agency.

Ownership of the SCE findings is with the acquirer. A group in the acquisition agency transforms the findings of the SCE team into acquisition risks and/or technical ratings. To ensure objectivity and integrity, the SCE team deliberately avoids interpreting its own findings. The acquisition agency uses the SCE findings and other criteria to select a source or monitor a contract.

A Final Comment

Software process maturity requires a long-term commitment to continuous process improvement. The CMM does not address all issues important for successful projects. The CMM does not imply or limit the choice to a particular life-cycle model. Specific software technologies such as reuse or prototyping are neither required nor excluded. The use of DoD-STD-2167A is not dictated. Projects' documents and products do not have to match the CMM exactly. Particular organizational or project structures are not mandated. Suggested software practices are generic, intended to provide flexibility. Each project or organization must clarify these practices for its specific situation.