Software Process Assessments: Issues and Lessons Learned

By

Mark C. Paulk
Watts S. Humphrey
George J. Pandelios
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213-3890
(412) 268-5794
Internet: mcp@sei.cmu.edu

Abstract
The software process assessment method developed by the Software Engineering Institute at Carnegie Mellon University is being used by a growing number of U.S. government and industrial software organizations. This paper describes the key organizational issues found by using this assessment method and relates them to traditional U.S. industrial practices. Some of the SEI’s experiences are described as well as the lessons learned from assessing over 60 software projects and interviewing approximately 2,300 software professionals and managers.

Keywords: software process, capability maturity model, quality management, process improvement, assessments.

1 Introduction
The Software Engineering Institute (SEI) at Carnegie Mellon University is chartered by the U.S. Department of Defense to improve the state of U.S. software practice. In addressing this goal, a key strategy is to motivate the rapid adoption of sound software engineering methods. The basic principles of the software process assessment method developed by the SEI (Humphrey89) are based on the concepts of statistical process control as espoused by Deming (Deming86), Juran (Juran88), Crosby (Crosby79), and others. The success of their approaches has been well demonstrated by the Japanese in their manufacturing industries, and there is growing evidence that these methods are equally applicable to software.

This paper reviews the basic issues in traditional U.S. software processes and why they are particularly troublesome. It also shows how an assessment helps identify those issues, describes common assessment problems and misconceptions, and describes how they can be handled. We begin with a brief summary of the SEI assessment method, the maturity framework, and the maturity questionnaire. We next outline the key software process issues that have been found most often in the organizations that the SEI has assessed. We present these issues as seen by senior management, middle management, and the software professionals.
respectively. We close the paper by addressing some more general assessment-related issues.

2 The Maturity Model and the Assessment Method
The principle behind the SEI capability maturity model is that the quality of a product stems, in large part, from the quality of the process used to create it. To consistently improve products, the process used for developing them should be understood, defined, measured, and progressively improved. Software products are built by such processes.

A software process under statistical process control can be managed and improved based on a statistically sound quantitative understanding. This greatly facilitates the control and improvement of such complex processes as software development and maintenance. A well-defined software process provides the basis for orderly productivity and quality improvement.

Five basic levels of process maturity, illustrated in Figure 1, have been defined to describe this progression from an ad hoc software process to one that is under statistical control (Humphrey87, Humphrey88). The typical level 1 (initial)

![Diagram of the five levels of software process maturity](image)

**Figure 1.** The five levels of software process maturity.
organization has serious cost and schedule problems. At level 2 (repeatable), the organization has installed management controls and generally learned to manage its costs and schedules. At the repeatable level, the focus is on the product, and the organization typically follows a reactive management system. In level 3 (defined), the process is well understood and characterized. Here, the organization has introduced a structured framework for software development and has established dedicated process improvement resources. In level 4 (managed), the process is quantified, measured, and reasonably well controlled. Data is available, for example, to establish improvement priorities and to support tool and technology investment. In level 5 (optimizing), process data is used to progressively improve the process in response to new and evolving issues and capabilities. At the optimizing level the organization focuses on continuous measured improvement.

Figure 2. The key process areas in the capability maturity model.
2.1 The Capability Maturity Model
We have refined and elaborated the maturity framework first published by the SEI in 1987, based on our experience in applying the assessment method. The result of this refinement effort is the "Capability Maturity Model for Software" (Paulk91) and the "Key Practices of the Capability Maturity Model for Software" (Weber91). The capability maturity model (CMM) establishes a set of public criteria describing the characteristics of mature software organizations. These criteria can be used by organizations to improve their process for developing and maintaining software, or by government or commercial organizations to evaluate the risks of contracting a software project to a particular company.

The CMM characterizes the maturity levels by key process areas, which identify the most significant improvement needs of each maturity level. The key process areas are summarized in Figure 2.

The SEI provides briefings and presentations to acquaint the software industry with the assessment method. Assessment teams use the CMM to guide them in identifying and prioritizing findings on the organization's software process.

These findings, along with guidance provided by the key practices in the CMM, are used by staff management (e.g., a software engineering process group) to plan a process improvement strategy for the organization.

2.2 The Software Process Assessment Method
Software process assessments are used by organizations to help identify the status of their software process and to identify a prioritized list of areas to address for process improvement. A software process assessment involves reviewing the software development process of several (4-5) key projects in an organization. The key components of the process are:

- analysis of project-level responses to the SEI maturity questionnaire
- in-depth interviews with the project managers
- discussions with key software practitioners throughout the organization (not just from the 4-5 projects)
- prioritization of issues provided by the CMM
- training, organizational knowledge, and professional judgement of the assessment team

One of the fundamental prerequisites to doing software process assessments is to obtain senior management buy-in. If senior management is unwilling or unable to act on the recommendations resulting from an assessment, the organization should not do the assessment. Assessments set expectations for the organization

---

1 To support software process assessments, the SEI has developed a maturity questionnaire. The current (preliminary) version was published as an SEI technical report in September 1987 (Humphrey87). While this questionnaire is preliminary, it has already been used by the majority of large U.S. aerospace and defense contractors, many of the major computer systems suppliers, several military and commercial organizations, and a few groups in Canada, Europe, and Japan.
and staff; when senior management does not support the assessment, it lowers morale and damages senior management credibility.

The assessment method, illustrated in Figure 3, guides an assessment team in using the CMM as a structured framework for determining the organization's key strengths and weaknesses. It also helps the organization establish effective improvement plans. It is important to note, however, that the capability maturity model and the maturity questionnaire are aids to assessment. With their help, the assessment team conducts a structured examination of the organization to identify key strengths and weaknesses and to determine the highest priority areas for improvement. The objective is to identify those issues that the assessment team and the organization agree are most important. Almost all of the issues in SEI assessments have been indicated by the questionnaire responses, but where other issues have been found, the team is trained to pursue them and address them in their findings and recommendations. The key is to address the issues in the organization whether or not they precisely fit the CMM.

2.2.1 Selecting and Training the Assessment Team
Selection of the assessment team (1) is a critical component of the assessment process. The team members must be respected, competent, and have the ear of management. The assessment team should have about six to eight members. This is small enough to reach consensus fairly easily, but large enough to permit a reasonable mix of experience and skills. Team members should have ten or more years of software engineering and/or management experience. The team members should have ten or more years of software engineering and/or management experience. The team leader should possess a minimum of twenty years experience.

The assessment team must then be trained. The SEI has competitively selected, trained, and licensed assessment vendors to assist organizations in training for and conducting software process assessments. These vendors provide training in the maturity model and the assessment method. In addition, they participate in, or may lead (at the client's discretion), the assessment. If an organization is a strategic SEI client, the SEI may participate directly in the training and the assessment.

---

2 It should be noted that the capability maturity model, though reasonably comprehensive, is not exhaustive. It does not, for example, address systems requirements or recruiting and staffing issues. Similarly, the maturity questionnaire, which has about 100 questions, can only sample the issues covered in the CMM. We estimate that a questionnaire to exhaustively cover the maturity model at a reasonably abstract level would require over 6,000 questions.

3 Contact the SEI at (412) 268-7700 for further information and vendor assessment training.

4 The SEI has only limited resources available to participate directly in assessments. The majority of assessments involve licensed vendors of the assessment method.
The assessment team attends a multi-day training course[^5] that presents the fundamental concepts of the CMM and the specifics of the assessment method. Training provides a common vision of what software process maturity means. Reliability and consistency can only be achieved if all team members operate by the same rules. Learning by experience, via role plays in the training, helps the team members recognize their individual biases. Training helps the team develop skills for investigating the organization's software process and reinforces the concept of team building, leading to a team which acts from a common vision and appreciates one another's abilities. Depending on the experience of the team, training in managing technological change and consulting skills may also be desirable.

![Figure 3. Steps in the Software Process Assessment Method.](image-url)

[^5]: The information presented in this paper is not sufficient by itself for readers to conduct an assessment. Anyone wishing to do a software process assessment should receive training, since an undisciplined application of the assessment method may serve an organization poorly.
2.2.2 Preparing for the Assessment

An essential part of preparing for the assessment is deciding the scope of the organization (or site) being assessed. The guidelines for determining the scope of an assessment include such characteristics as:

- a single management sponsor, who will sponsor the assessment and improvement activities
- a common process culture, to provide an infrastructure for the improvement effort
- a single geographical location, which is usually a prerequisite for commonality of management and culture

The organization being assessed will frequently be part of a larger parent organization. It may be one of several groups at a single location.

When the scope of the assessment is determined, the maturity questionnaire (2) is completed by project representatives from the site being assessed. Each question relates to key practices in the CMM. The project representatives are typically project or software managers from various projects. Once this activity is completed, the assessment team performs a response analysis (3), which tallies the responses to the questions and identifies those areas where further exploration is warranted. Since each question in the maturity questionnaire is derived directly from the key practices in the CMM, the areas to be investigated correspond to the CMM key process areas. For each maturity level, the key process areas are listed, and the degree of implementation is noted. The response analysis focuses the attention of the team on the maturity level and key process areas deserving immediate investigation. It should be noted that the maturity questionnaire’s role is to serve as a springboard for discussion. It has minimal impact on subsequent assessment steps.

The team is now ready to visit the site.

2.2.3 Conducting the Site Visit

The site visit (4) is an essential part of the assessment method. Beginning with the results of the response analysis, the team conducts interviews and reviews documentation to gain an understanding of the software process followed by the site. The key process areas and key practices in the CMM guide the team members in questioning, listening, reviewing, and synthesizing the information received from the interviews and documents. The CMM provides guidance and examples of best practices from mature software organizations; the CMM is not a prescription for the "one right way" to a mature software process. The team applies their professional judgement to determine what the key issues in the organization are and to relate them to the key process area goals. When there are clear differences between the key practices in the CMM and the site’s practices, the team must document their rational if they decide the site has satisfied a key process area.

The goal of the site visit is to come to an understanding of the process that is followed by the projects, which may differ radically from what was planned, in
the organization’s official policies and procedures, or indicated on the maturity questionnaire response sheet. A great plan or a superb standard is of little value if it is not followed. By conducting interviews and reviewing documentation, the assessment team reaches an understanding of the organization’s actual software process.

The assessment team explores the software process through a series of structured interviews held with the project representatives or leaders. One member of the team acts as a discussion leader for each interview. The goal is to foster a dialogue, not conduct an interrogation; therefore, the questions asked by the discussion leader are open-ended and designed to elicit as much information as possible. During the response analysis exercise, the assessment team will construct a script for each discussion leader. This script contains questions for a project leader that are derived from the questions in the maturity questionnaire. For example, the questions shown below (2.1.1, 2.1.15, 2.1.16) may be grouped into a general category called Estimating. The project representative’s answers to each question as well as any written comments are also recorded by the team.

2.1.14 Is a formal procedure used to make estimates of size?
2.1.15 Is a formal procedure used to produce software development schedules?
2.1.16 Are formal procedures applied to estimating software development costs?

During the interview, the discussion leader may ask an open-ended exploratory question such as: Would you please describe your cost, size, and schedule estimation procedures?

If more information is desired, the discussion leader may ask a follow-up question such as: How do the procedures work for you?

All questions are designed to be as non-threatening as possible. If the project representative indicates that written procedures for estimating exist, the discussion leader will ask for copies of those procedures.

The team also conducts unstructured discussions with the software practitioners of the organization. Functional Area Representatives (FARs) are selected on the basis of their experience, influence and stature with respect to their peers, and direct involvement in developing software. Eight to ten FARs are selected from each of the general areas of quality assurance and release, software integration and test, coding and unit test, and requirements and design. They are asked to talk about issues facing them in performing their work. The assessment team typically does not comment during the session. At the end of the discussion, each FAR is asked what is called the "big question": If there was one thing you could change, other than your boss or your paycheck, what would it be? This quickly surfaces major issues and provides input for generating recommendations.

*These questions are taken from (Humphrey87). The updated maturity questionnaire is not being publicly distributed at this time.
The assessment team then takes the information it has gathered and creates preliminarily findings. Preliminary findings are first drafts of issues that the team believes affect the organization. These preliminary findings are then reviewed with each project representative. Feedback gathered from these meetings, as well as from prior sources, is considered by the team. The team comes to consensus on the issues and their consequences and then prepares assessment findings in language appropriate to the organization’s culture and the understanding of senior management. The findings are put together into a briefing and presented to all of the project representatives and to all of the FARs in separate dry-runs. Feedback and reactions from each group are considered by the team.

As the team zeroes in on the issues facing the organization, they must apply their professional judgement. The professional judgement of the team is crucial in determining the critical issues facing the organization and their relative priorities. The result of this consensus building process is an understanding by the team of the organization’s software process and the highest priority issues for improvement.

2.2.4 Using the Findings of the Assessment

At the end of the on-site period, the assessment team produces a list of findings (5) which identify the strengths and weaknesses of the organization’s software process. The findings relate to the key practices and the key process areas in the CMM. The findings become the basis for recommendations and can be used by the organization to prioritize its software improvement efforts.

The findings are summarized in a key process area profile (6) that shows the areas where the organization has, and has not, satisfied the requirements of the key process areas. Each key process area is classified as not satisfied (NS), partially satisfied (PS), fully satisfied (FS), or not applicable (NA). Each key process area within a maturity level must be fully satisfied to attain that level’s process capability.

Aside from the findings themselves, perhaps the most important product of an assessment is the buildup of momentum for process improvement that an assessment creates. Because an organization-wide consensus emerges on the issues, the outcome usually forges an informed, motivated, and management-supported team of professionals committed to process improvement.

Process improvement is not a quick fix. It is an investment in the organization for the long haul. Assessments serve as the initial focusing activity of the organization, but the road to improvement takes time and effort. Just the
assessment activity itself may require six to twelve months of preparation by the assessment team. Managers and staff at the assessment site also contribute their time and effort. The decision to perform an assessment should be made carefully, with due attention to the follow-through required to implement process improvements based on assessment findings. The process improvement activities resulting from an assessment may require a significant expenditure of time and resources.

The findings and the key process area profile are presented to the managers and software professionals who own the software process and who can act to improve it. After the site visit, the team works with the affected parts of the organization to recommend (7) what actions can be taken to correct the weaknesses identified in the findings. The organization then takes action (8) to improve its software process.

In summary, the software process assessment method:

- uses the maturity questionnaire as a springboard for the on-site visit
- uses the CMM as a map that guides the on-site investigation
- develops findings that characterize the organization's highest priority software issues
- derives a profile based on the degree of satisfaction of the goals within the key CMM process areas
- presents the assessment results to the appropriate audience
acts as a catalyst, motivating the organization to address the issues found
is only the first step on the road to process improvement

3 Key Software Process Issues
Perhaps the most surprising finding from the SEI assessments is that the key problems in the organizations are not new. Not only are they common problems of long standing, but they are typically well known within the organizations themselves. The most obvious question then is: If they already know their problems, why aren't they fixing them? While the answer to this question is the primary topic of the rest of this paper, a few general observations are appropriate at the outset.

First, and perhaps most important, industrial management style is primarily reactive in the U.S.. As a nation we have trouble giving adequate priority to problems until they reach crisis proportions. Witness, for example, the slow or nonexistent reaction of the U.S. electronics, steel, auto, and ship-building industries to loss of market share to foreign competition. Unfortunately, software is no different. The big promotions go to the heroes who rescue the organization from an impending disaster rather than to the quiet professional who stayed out of trouble in the first place.

Fortunately, there is growing recognition that reactive software process management is not effective. As multimillion-line software systems become commonplace and their quality becomes increasingly critical, the historical bug-driven quality strategy will not suffice. The need is to evolve toward processes that assure high quality. Strategic thinking is essential because a high-quality process necessarily involves increased skills, substantial investment, and sound technological plans.

Perhaps the best indicator of this need for strategic thinking is the experience of one group that the SEI assessed. They were developing a large, complex weapons system for the DoD. The last program they had developed of this type required 30,000 lines of embedded software. The software estimate for the newer, more advanced system was for 3,000,000 lines of code. While this size software system no longer seems extraordinary, it is worth noting that, for this organization, it represented a scale increase of 100 times.

In software, unlike most other technical fields, scale factors of 100 are taken quite casually. This is a key source of many software problems. The transition from 3 to 300 miles per hour requires several changes in technology. Similarly, compare a 1 to a 100 story building or a 10 to a 1000 foot boat. Software organizations face enormous new challenges when they make such scale transitions. What is not well recognized is that, based on history, those software organizations that survive for the next 20 years will undergo at least one such transition.
The purpose of assessments is to initiate planned change. This requires strategic planning, not reactive planning. This is perhaps the greatest challenge facing software today. To meet the needs of an increasingly competitive international environment, there is no other viable choice.

3.1 Senior Management Issues
As stressed by Deming in his fourteen points (Deming86), leadership is required to achieve improvement. Senior management’s support and commitment are essential for any sustained program of process improvement. The growing attendance at the quality programs offered by Deming, Juran, and Crosby attest to the increasing recognition of this fact by U.S. industry in general. Unfortunately, in many organizations this new quality focus has not been effectively applied to software. The connection between the software process and software quality is a direct one, and it is essential that management adopt the basic premise that the quality of software products largely depends on the quality of the processes used to build them.

Of the four basic senior management issues we have encountered, only one is software specific; there is a pervasive belief that traditionally effective management methods do not work with software. The other three issues concern the following: the need for immediate action, "shoot the messenger," and "the problems are technical." Each of these issues is treated further in the following paragraphs.

3.1.1 The Need for Immediate Action
With the increasing criticality of software, a software failure in a comparatively minor subsystem may lead to failure of the entire product with all the risks, costs, and attendant dissatisfactions. As has been observed by Crosby, Juran, and others, there is a large and rarely quantified cost of poor quality. The proof that poor quality has a major cost has been amply demonstrated in many other industries; we must now address these issues in the software industry. The next question then concerns the motivation for quality improvement. In military products, software quality may be literally a life-or-death tradeoff. For the space shuttle software, quality was the highest priority, with extraordinary steps taken to review, simulate, test, and retest the software before flight. As a result, no mission-critical software defects have been encountered in flight.

A less well recognized need for software process improvement is corporate profit and even competitive survival. An important indicator is the growing recognition by many U.S. Department of Defense acquisition groups that software process is important. They are increasingly using a SEI-developed method to evaluate the process capabilities of their software vendors during source selection. Organizations that do not focus on software process improvement will simply not get their business.

3.1.2 Shoot the Messenger
Another basic software quality principle is that the professional’s job is to follow the process, the manager’s job is to fix it. Quite obviously, however, to fix a
defective process, management must understand the problem and how to fix it. The typical U.S. management style, however, is often a serious problem.

It is human nature to get unhappy when told bad news. A common and highly destructive consequence is to behave as if the bearer of bad news is somehow responsible for it. While this is an extreme reaction, a more common one is to give the messenger the job of fixing the problem. Since software people are rarely willing to get blamed for other people’s problems, and they are almost universally overworked, they are rarely willing to carry problems upstairs.

While this is not a software-unique issue, it is often more severe for software because management is less familiar with the issues commonly needing their attention. The professionals also generally have had little experience with successful management problem solving. The SEI assessment process has been designed to provide an impersonal and non-threatening means for defining software organizations’ most critical process problems and describing them to management, together with some recommendations for their solution.

3.1.3 The Problems are Technical
The following is a common assessment experience. At the findings presentation, a few senior managers sit at the front of the room at one side while the rest of the room is filled with software professionals and junior managers. Since these software groups typically have serious cost, schedule, and quality problems, the executives expect to hear that these people have screwed up and how they should react. As the meeting unfolds, however, their expressions generally change from those of interested observers to directly involved participants.

They are usually surprised to learn that the lack of a commitment system, inadequate training, an ineffective software project planning system, and an ill-defined quality policy are fundamental management problems. When senior management sees this, it is encouraging to note that they almost universally accept responsibility and initiate appropriate action.

3.1.4 Software is Different
The reaction of one executive is typical. He said: "I don’t know anything about software so I can’t be of much help with these problems." The feeling generally is that since software is different, traditional management methods do not apply. The fact is, however, that software requires traditional engineering management, plus a lot more.

No self-respecting engineering manager would dream of letting his or her people work without estimates and schedules. Product releases should always be under configuration control and parts lists, process specifications, and test plans are generally well documented and reviewed. For some reason, the engineering managers do not even demand this level of discipline from their software people. With few exceptions, these are the key software process improvement needs that their organizations face.
3.2 Middle Management Issues

Andrew Carnegie once said, "As I grow older, I pay less attention to what men say. I just watch what they do." The middle managers are truly in the middle. On one hand they are responsible to senior management for getting the job done and on the other, they must lead, direct, and motivate the professionals' work. There is a school of thought which says managers should assiduously study their bosses to learn their priorities and then make them their own. Regardless of what managers say, most of us also soon learn that the only true guide to management's priorities is management's actions. Senior managers, unfortunately, rarely spend a noticeable amount of time on quality software or process improvement issues.

Middle managers thus generally have an extraordinarily difficult job. They need to balance the needs and pressures of the daily work against the need for organizational development. They must often do this under the leadership of an only partially informed senior management. The middle managers are, therefore, often answerable for the organization's performance without having the resources and authority to be fully effective. As a consequence, middle management is often highly skeptical and even strongly resistant to change. Their key issues are usually of four types: that can't be done, we are already handling that, product delivery is higher priority, and where do I get the resources?

3.2.1 That Can't Be Done

The case of one military officer is instructive. He was in charge of a large software organization and had just been given a recommendation to take a number of personnel actions. His immediate reaction was that it couldn't be done. "Regulations prevent it," he said, and "an act of Congress would be required." After some discussion, which included the possibility of obtaining an act of Congress, he was persuaded to have his people investigate the matter further. They found no regulation problem at all, and the actions were promptly implemented.

While there are many real inhibitors to organizational change, one of the most pervasive is the general belief that things can't be changed. Surprisingly, on examination these constraints often turn out to be imaginary. Since their effect is quite real, however, it is generally necessary to have change agents in the organization who will work to overcome both the real and imaginary roadblocks.

3.2.2 We're Already Handling It

This can be a tricky issue, as demonstrated by one example. The recommendation was to get better control of program changes by requiring that the design be updated every time the code was modified. The manager insisted this was being done and became irate when told it was not. His first demand, in fact, was, "Who told you that?"

Actions are often underway to address some of the key problems in an organization. Where such work is in place and appears effective, it must be recognized and reinforced. The tricky part is when the manager has directed some needed actions, and work was either not performed or not done effectively.
When such discussions focus on placing blame, the constructive and impersonal approach to process improvement often suffers.

While software professionals do occasionally make such mistakes, it is generally not because of incompetence. Either they lacked the training or experience, were given imprecise directions, or were faced with conflicting needs. The most common cause is that they have so many things to do that they can only address those with the highest priority. While the manager may believe the particular topic is critical, if he or she has not made that priority clear, it will likely not be followed. The objective should be to reinforce the manager's existing convictions while exploring the problems with the previously initiated actions.

3.2.3 Product Development is More Important
Since the justification for improving the software process is to offer improved products, some managers suggest that, since their products are high-quality, their software process must be good. This is not necessarily true. When working on comparatively small projects with a few software engineers, the dominant factor in product quality is the competence and dedication of the engineering team. With larger systems, where many people must work together effectively, the process becomes more important. With very large projects with mission-critical functions, a high quality software process is essential. While good people can overcome (and frequently have to) enormous obstacles to get their jobs done, energy, enthusiasm, and hard work are often not enough to overcome all the problems.

A mature software process enables routine work to be handled routinely. With an immature process, the software team often devotes much of their time to fire-fighting. One example was a group who was designing an enhancement to an existing communication system. While the upgrade was urgently needed, the software people had to devote several months to identifying, cataloging, and describing all the message types currently handled by the system. The lack of any routine way to handle this single system management task seriously delayed their project. When managers understand the impact of such issues on their work, they are often more willing to support process improvement.

3.2.4 Where Do I Get the Resources?
This is a real issue that often totally blocks progress. In one example, the managers involved with the assessment were very concerned about the vice president's reaction to the need for special process improvement resources. They knew his budget was tight and that he probably could not provide the few people needed. When the time came for the assessment presentation, the vice president invited the corporate CEO to the meeting, and he came. At the meeting's close, the CEO made a few comments about the assessment and how impressed he was at the organization's evident commitment to improvement. He then added that he knew resources were tight, but that he would see what could be done. He subsequently authorized the additional staff.

Few middle managers have the flexibility to make such resource allocations. That is why senior management support is essential for software assessments.
While the middle managers and professionals can make a lot of progress in identifying problems and recommending actions, little if anything will get done without resources.

3.3 Software Professional Issues
Typically, the software professionals are enthusiastic about the assessment process and strongly support the need for process improvement. They know the problems of a poor process since they face the need to fix some oversight or overcome some daily administrative inconvenience. In one organization, for example, the procedure for scheduling conference rooms turned out to be a serious impediment to conducting design and code inspections.

The professionals’ issues thus typically concern their understanding of the assessment process and the software process in general.

3.3.1 The Key is to Get a Better Score
In taking a test, it is natural for people to focus on the resulting score. The assessment process uses a questionnaire of about 100 yes/no questions to identify some key areas for initial discussion. In one assessment, one of the organization’s professionals who was on the team was so concerned with the final score that she consumed a good deal of the team’s time in detailed debates about every "no" response.

While it is proper to probe in detail to assure complete understanding of an issue, the critical point for software organizations is not their static score but their improvement vector. Because software challenges inexorably increase, a strong but complacent organization will certainly deteriorate. With the software process, it is safe to say that if it is not improving, it is getting worse. Rather than merely concentrating on the score, the key is to maintain a focus on continuous process improvement.

3.3.2 The Assessment Only Addresses Software Issues
The capability maturity model and the assessment method only address software development and maintenance. During the course of an assessment, however, non-software issues invariably arise. Some of the recommendations coming out of an assessment are quite likely to involve, for example, system engineering and its relationship to software. One of the values of an assessment is that it surfaces the predominant problems facing an organization developing software. Software tends to pervade organizations, and it should be recognized that the solution to a problem may lie outside the software group.

Requirements and systems engineering are common problem areas for the software organizations that the SEI has assessed. In one organization the lack of a systems group meant that the software engineers had to address the systems issues before they could do their jobs. This was generally adequate except where multiple hardware/software boxes had to interface. At the time of the assessment, a series of ad hoc product decisions had resulted in a serious overall systems compatibility problem. Unless a significant redesign was undertaken, the organization would not be able to make their new product compatible with the
existing product line. Since this would cause added customer expense, it was a very important problem. It took the software assessment to surface this issue in a way that management could recognize and address it.

3.3.3 Process is Only for Big Software Projects
While it is true that the SEI’s capability maturity model and maturity questionnaire emphasize large project issues, process improvement concepts apply equally well to very small projects and to maintenance as well as to new development. In one case, the data gathered on small maintenance updates for a large communication product demonstrated an extraordinarily high error rate for small changes. Changes of only a few lines of code had error rates of nearly 40 times those for new code. This data motivated a complete replanning of the maintenance change, review, and test processes.

3.4 General Issues
While the above sections cover most of the key issues commonly encountered, two final topics deserve special attention.

3.4.1 It’s Someone Else’s Problem
This is not a common assessment problem but when it does come up, it can be serious. It may be true that software organizations suffer from problems outside their control, such as poor and unstable requirements, an ineffective or non-existent systems organization, or an uninformed or bureaucratic program manager. It is always the case, however, that these same software organizations have many critical problems which are under their complete control. Until they face this and begin to address their internal improvement needs, any focus on the broader external issues is merely a distraction; substantial improvement is unlikely.

3.4.2 The Optimizing Level is the Goal
The optimizing level (level 5) of the SEI maturity framework is often viewed as either unachievable or the end objective of software process improvement. A few level 5 projects have been observed, and it is interesting to note that they universally recognize the need for continued improvement. In fact, as their process maturity improves, organizations are better able to understand their opportunities for further improvement. It is only the initial level organizations that don’t seem to know what to do to improve.

4 Conclusions
After conducting software process assessments for over four years, the SEI has gained an appreciation of the key process improvement issues software organizations face. The most encouraging finding is the rapidity with which senior corporate management groups endorse the assessment concept and support continued process improvement. While the issues we face are serious, they can be overcome in all of the organizations we have assessed.
References


