Molding the CMM to Your Organization[1]

Karl E. Wiegers

Process Impact
716-377-5110
www.processimpact.com

Religious wars wage continuously in the software industry. One current conflict is over the merits of the Software Capability Maturity Model (CMM), developed by the Software Engineering Institute (SEI). Inside the CMM fortress are hundreds of organizations that have found the CMM to be a useful tool for helping them improve the quality, timeliness, and predictability of their software products. Besiegers are arrayed outside the walls. They loudly proclaim that their organization wasted tons of money fruitlessly attempting to do the CMM thing, that the CMM emphasizes dogmatic perfection rather than realistic approaches to quality, or that the CMM doesn't apply to them for any number of reasons.

As in many wars, both sides have a point. Application of the CMM as a framework for software process improvement has clearly been of value to many companies. Others have struggled with the CMM, trying to follow its guidance, but achieving less than stellar results. Few organizations that fail in their improvement attempts publish their sad tale so the rest of the world can learn from the experience. The wise process improvement practitioner will keep eyes and ears open to what those around her have tried, to avoid rediscovering painful lessons.

Set the Right Goals

Someone came up after I gave a presentation at a conference recently and asked me a question about his company's process improvement program, whose goal was to achieve CMM Level 2 (Repeatable). I asked why they wanted to reach Level 2. His blank stare suggested that his company had not yet held that important discussion. Striving for Level 2 (and beyond!) is just what you do if you're playing the CMM game, right?

I don't think so. Your goals for any improvement effort should be primarily *business* goals, not maturity level goals. Appropriate business goals include shortened product cycle times, fewer defects at all development steps, fewer schedule overruns, and less unplanned overtime work. View the push for higher process maturity as one means to achieving these worthy business ends.

A successful software process improvement initiative doesn't require that a team of official assessors give you the rubber stamp of approval with a big "Level 2" or "Level 3" on it. Despite the step-function nature of the CMM's five maturity levels, process capability is a continuum. You may fall a bit short of officially achieving the next maturity level, but does that mean you're still stuck in the mud-filled trenches of process chaos? Of course not.

[1] This article was originally published in *Software Development* magazine May 1998. It is reprinted (with modifications) with permission from *Software Development* magazine.

Some organizations, particularly government contractors, feel the pressure of external motivations for achieving higher CMM maturity levels. But what should impress a potential prime contractor more: an official maturity level rating, or a track record of results that demonstrate your ability to make and keep realistic project commitments and deliver high-value products to satisfied customers? I vote for the latter.

Although you must have repeatable processes in place to achieve CMM Level 2, you don't need to be at Level 2 to work in a repeatable way. Your objective is to use the framework and concepts of the CMM to enhance those software engineering and management activities that are critical success factors for you. You won't go far wrong by focusing on the key process areas (KPAs) at Level 2 of the CMM: requirements management, software project planning, software project tracking and oversight, software configuration management, software subcontract management, and software quality assurance. Nearly every project I've seen has shortcomings in those areas, and fixing them has always led to better results. Use the CMM to help you select practices that will enable you to achieve your business objectives, project after project.

Why Not Just "Do the CMM"?

The CMM is available in the form of two huge technical reports from the SEI, and also as a 430-page book (*The Capability Maturity Model: Guidelines for Improving the Software Process,* Addison-Wesley, 1995). Simply declaring to your software community that you're going to follow the CMM is likely to generate immediate resistance: "I can't do all that!" "I don't understand it." "That doesn't apply to me."

Although improvement objectives like "Level 3 in 3 [years]" are concise and punchy, they may not be right for your organization. A better approach is to respect the substantial wisdom embedded in the CMM, but adapt it to your organizational culture, needs, and realities. Study the key process areas of the CMM, especially those at Levels 2 and 3, and think about the activities and practices that will help *your* organization achieve its goals. Focus on obtaining "Level 2 results" — predictability, stability, and visibility — more than on satisfying every detail of the CMM's expectations for Level 2.

Adapting the CMM: A Case Study

The rest of this article describes the approach that a large software division in a very large company took to mold the CMM to best meet its needs. I was a member of the software engineering process group (SEPG) that helped guide this division's process improvement activities. This division had about 500 software engineers working on dozens of projects to create both commercial applications and embedded software for complex equipment products. Our SEPG helped set strategic goals that aligned with business objectives, and provided an infrastructure of training, resources, expertise, and CMM knowledge.

We tried to steer clear of the dogmatic quest for the next maturity level, although at times this was attractive to managers who favored simple responses to complex problems. Instead, we focused on adapting the CMM's practices to address the very real software-related problems facing this organization. We found that adapting the CMM in this way made model-based process improvement more palatable to the client communities than a more rigid, dogmatic application of the CMM.

Process Assessment

Any process improvement effort should begin with some kind of assessment, to establish a baseline understanding of current practices and problem areas. One approach to process baselining is to perform a full organizational appraisal based on the CMM or another improvement framework. An alternative that works well in small groups is a simple, facilitated brainstorming session. We chose an intermediate approach: we developed a flexible process mini-assessment method based on the CMM that we could apply to a wide variety of projects.

Over the span of a year, our SEPG conducted about 20 of these mini-assessments on diverse projects. Some project teams were happy to see us bring more structure to their improvement activities, while other teams clearly resented our presence. This is par for the process improvement course. The mini-assessments typically addressed the six Level 2 KPAs, plus occasionally the peer reviews and intergroup coordination KPAs from Level 3.

The deliverables from each mini-assessment included both relative strengths of the project and a list of findings that pointed out relative process weaknesses. We augmented each finding statement with a list of current and potential consequences of that process shortcoming. There's no point in worrying about findings derived by comparison against a reference model if the team isn't experiencing (or likely to experience) problems as a result of that finding. We also proposed several realistically achievable recommendations for addressing each finding.

We collected our findings from all these projects in a database, then looked for patterns of recurring problems. Certain issues were prevalent across multiple projects. Often, we could cut-and-paste from the findings database to generate a portion of the findings on another mini-assessment. It seemed reasonable to focus our limited improvement energy on these common problem areas, using the CMM as a guide to suggest sensible remedies for those problems.

Molding the CMM

Once we had the common process weaknesses from across the organization in hand, we scrutinized the Level 2 KPAs and picked out individual practices that we felt were worth pursuing. The selected practices had to satisfy three criteria:

1. If properly implemented, they would make the findings — and the associated consequences — go away.

2. They must be realistically achievable.

3. They could be interpreted sensibly in the context of our organization: what level of managers were appropriate for a particular CMM-specified review, for example.

Each key process area of the CMM describes several key technical or managerial practices ("Activities Performed") that will generally satisfy the goals prescribed for that KPA. However, the CMM also recognizes that other key practices are necessary to enable an organization to consistently and routinely achieve those goals. These practices are found in components of each KPA called Commitment to Perform, Ability to Perform, Measurement and Analysis, and Verifying Implementation. We examined all of these CMM components for each KPA and chose the practices that we felt were both pertinent and realistic for our division's process improvement.

CMM maturity levels are achieved by demonstrating that specific sets of KPA goals are being satisfied, not by performing all of the key practices of each KPA. The practices we selected

Table 1. Examples of Evaluation Criteria for Requirements Management.

Item	Action or Deliverable	How Verified
A documented procedure for developing a software requirements specification has been adopted and followed.	D	See procedure; compare SRS to procedure
Software requirements are documented, with a preliminary version at Gate 1 and a baselined version at Gate 2.	D	See two versions of SRS at Gates 1 and 2
The baselined software requirements document is under version control and change control.	D	See that SRS versions are identified; change procedure in place
Team has been trained in requirements engineering and management, through course 6307 or equivalent.	A	Review training records

generally aligned with satisfying those KPA goals, but that alignment was secondary to the ability of the practices to address the division’s known process shortcomings effectively.

We documented exactly what an effective application of each selected KPA practice would mean in our organization. These individual “evaluation criteria” included a statement of the observable reality if the practice has been successfully applied, and also whether this observable was an action or a deliverable of some kind. Table 1 illustrates some of the evaluation criteria we developed around the requirements management KPA. The evaluation criteria set out the expectations of what should be different if our organization succeeded in following the CMM practices that we adapted.

Next, we determined what would constitute suitable evidence that each selected practice was being applied as intended. This verification evidence should be simple, realistic, and objective. I have found that responses to questionnaires about current process performance are often overly optimistic. People are tempted to give themselves generous partial credit, “rounding up” in their favor. The verification methods did not address *who* would be doing the evaluation or the exact procedure to be used, thereby leaving latitude for doing whatever makes the most sense on each project.

A fundamental philosophy behind our approach was that we did not worry about whether our organization would achieve Level 2 even if every project satisfied all the evaluation criteria. That was not important to us. What was extremely important was to make sure that the CMM practices we selected really would rectify some known shortcomings in the way this division currently ran its software projects. Being “substantially Level 2” was fine with us.

Tracking Progress

Tracking progress in software process improvement is not trivial. The ultimate indication of success is that the team is working in some new ways that yield results superior to those previously achieved. Results-oriented measurements around quality and productivity are often lagging indicators of success in process improvement.

An alternative is to use activity measurements, on the faith that if we perform the right software engineering and management practices effectively, we should eventually achieve better results. However, we need finer-grained tracking than doing a full-scale process assessment every couple of years to see if we've reached the next CMM maturity level. The evaluation criteria we devised permit just such fine-grained tracking. Each KPA has an average of 21 individual evaluation criteria. Therefore, we can set goals in terms of the percentage of evaluation criteria that are satisfied over time for each KPA, and we can track progress toward those goals.

We judge each evaluation criterion in a binary way: either satisfied, or not. However, there is still opportunity for evaluator judgment as to what constitutes satisfaction. For example, certain evaluation criteria involve sending the whole project team to a specific training course around such topics as requirements engineering and management, or software project planning and tracking. For a team of 20 people, how many have to be trained to satisfy this criterion? All 20? Just 10? How about 13, or 18? We expect the evaluators to be reasonable when making such judgments. Formulaic approaches to process improvement cannot replace common sense.

What We Learned

The notion of adapting the CMM to fit the needs and realities of our organization was not uniformly accepted. We encountered resistance from one member of the SEPG, who felt we were attempting to reinvent the CMM, wasting time and confusing practitioners in the process. My opinion, though, is that the CMM is simply a model that recommends ways to address the kinds of problems that affect many software development organizations. Practitioners are expected to apply this model in a sensible, relevant, and practical way, which may mean adjusting it to fit local situations and objectives.

We also received resistance from certain of the client project teams. Their initial reaction was that our approach was still too CMM-ish, and there were far too many items in the evaluation criteria for them to take seriously and address. "We'll go through your list and cut it down to the size that makes sense for us," they said. So they took a close look, and they kept every single one of the evaluation criteria our SEPG had proposed!

I had predicted that this would happen. This outcome reinforces my contention that you should apply most of the practices in the CMM because they represent sound approaches to software engineering and management, not simply because they are in the CMM. Even if the CMM was never conceived, these practices describe things that most projects ought to be doing. It's easy to be overwhelmed when confronted with the contents of the entire CMM, or any other improvement model. When examined item by item, though, you won't find too many things that a reasonable developer or manager will pooh-pooh as silly or useless. If you do find some elements that don't fit your world, don't do them!

As with most things in life, neither extreme position is sensible. Don't throw out the CMM simply because there are parts that don't apply to you and other important components seem to be absent or misplaced. But neither should you apply it verbatim without considering how best to make it work for your organization.

The Bottom Line

I would love to be able to report that our entire organization followed this approach to software process improvement and achieved fabulous results. For several reasons, I can't present such a glowing outcome. Because of reorganizations and changing improvement goals, not all of this method has yet been applied. Several projects, though, created and began to implement action plans to satisfy the evaluation criteria and track their process improvement progress in this way. These projects chose several different methods for monitoring progress. Some wanted to have the SEPG do the evaluation through a formal mini-assessment, while others preferred to use the evaluation criteria as checklists for progress tracking on their own.

Despite the lack of resounding results to date, I'm confident that our approach represents a sensible way to apply the CMM to a large, diverse organization. By first getting a handle on the real problems and issues facing the projects, we can thoughtfully judge how selected elements of the CMM can help us better achieve our technical and business objectives. Focus on what the experience embodied in the CMM can do for you, rather than on racing to reach the next maturity level.