What is CMM® and Why Should I Care?

As the wave of immense commercial spending in the “high tech” and software industry washes ashore, many firms were washed up for good. Firms with diverse customers and/or products may have bobbed up and down a bit as the wave passed by, but many (especially smaller) firms were caught in the undertow and are scrambling to stay above water. Yet there are those companies who were nearly unaffected – except for the onslaught of refugees’ resumes from the beached and the tumbling. These unaffected firms are steeped in government work. Companies – previously riding high on the water and now tumbling in the undertow – are looking at their more stable classmates for clues and have had a change of tune lately, now themselves seeking the stability of government work.

If this describes you and you’re a software company, you may be in for a rude surprise called “the CMM®.”

You may have heard something about the CMM®, and for any one of many reasons you’ve decided not to look into it. CMM® stands for “Capability Maturity Model.” And why you should care is the subject of this article: *You can’t build software for the Federal Government unless you follow the CMM®. (There are a few exceptions – but don’t try to squeeze into these exceptions, they’re not loopholes.)*

What is It?

The CMM® is a model for managing software development. It is not a software development life cycle, but rather a model for *managing* the development. The difference is that a *development* life cycle is a way for programmers to code, test, deploy, and build on their software. A *management* model is a way for software projects to plan, organize, and identify the activities necessary to know how to run the project. To gain insight into and management control over the development process, so you can predict project success and adjust when necessary.

In the 1980’s a well-publicized Standish Group study that found that over 30% of all software projects failed to be delivered and that of the remaining, nearly 80% failed to come in on time and budget. And these figures account for small and large companies combined – accounting for the evidence that the smaller companies success rates are nearly twice as high as the success rates of larger companies. Further research indicated that most of the money in the software development industry was spent on fixing software delivered with bugs, rather than on the initial development itself. Similar data finds that of the successful projects, many were successful only after the amount of functionality was reduced – likely a rough reflection of the amount of already delivered code. So in effect, about 16% of all software projects surveyed were delivered on time, on budget and with the expected functionality, with most of that software being developed by smaller companies. And this assumes the companies surveyed were even able to track and estimate their schedules and budgets.[1]

A this point, the Department of Defense (DoD) funded the Software Engineering Institute (SEI) at Carnegie Mellon University to find ways to help defense contractors build software more economically. If defense contractors could reap the business benefits of building software of high quality and predictable fiscal and schedule results, the DoD surmised, then overall the cost of software acquisition would drop.

A defense software project was typically very large, took a long time, and was often executed by several different companies spread across many locations. The SEI looked into a great number of similarly large and complex commercial and defense software projects. First they defined “successful project” in terms of how close a project was able to meet cost, schedule, and quality objectives. In the “quality” objective they included how many post-delivery corrections, fixes, and upgrades the product required before it finally worked like the customers expected. “Failed projects” were those that were either cancelled for cost, schedule or quality reasons, or those that significantly overran these objectives.

Tired of overpaying for software projects that never went anywhere the DoD sought to determine the factors that lead to successful projects. Sure, they noted what the failed or failing projects had in common, but instead of focusing on all the mistakes, overruns, slip-ups, and what *not* to do, it focused on what the successful projects had in common, and what they did that correlated with their success. The result became the Capability Maturity Model, or CMM®.

Simply put, the CMM® is a framework of processes. The research found that successful projects had specific key processes in common and organized those key processes into maturity "levels." There are 5 levels in all; each level has a specific set of key processes associated with it. Level 1 is actually where most companies are: using "ad hoc" processes, relying on heroics to complete projects and never doing the same thing the same way twice. The CMM® doesn't "define" level 1, it simply describes the characteristics common to companies who demonstrate process capability *immaturity*. There are several other symptoms of process immaturity, and the more complex your business or product, the more likely you will display these symptoms.

Level 2 is the first level in which the processes are actually defined by the CMM®. This level is called the "repeatable" level because companies at "level 2" are *mature* enough that their processes can be repeatedly used to plan, predict, and execute projects. Though they may not yet hit all project performance targets, their processes support their objectives and they can successfully repeat them from project to project. There are a total of 18 key process areas in levels 2-4, six of them fall into Level 2.

In the '90's, the CMM® came into widespread DoD use and thanks to enormous success and enthusiasm quickly spread to other agencies. At first, a company had to demonstrate that it was following the CMM® only in order to do software development for the DoD, but just as quickly, the other agencies began making the same requirement. As a result of its credibility (as well as the fact that many commercial software firms were also DoD/Federal Government contractors), the CMM® took hold in the commercial world. Today, the CMM® is the *de facto* standard for software management, and is internationally recognized as a very powerful business tool and differentiator.

The CMM® was not designed to be a checklist government bureaucrat's standard. In fact, it was intended to essentially be a broadly applied DoD "gimme." It was the DoD's effort to collect the best business practices from around the software industry, then tell the world (for free) what they are. In addressing level 1 symptoms of immaturity, companies have found that they've also cured many of their other problems including employee retention, customer satisfaction, and repeat business. If it wasn't for the business value achievable via the CMM®, then most commercial enterprises wouldn't even bother, would they?

What Does "process maturity" or "capability maturity" mean?

When all the CMM® key processes are looked at collectively, it starts to paint a very interesting picture of information. When executed as part of a coordinated management effort, the key processes provide software developers and managers with insight into and control over the product as its being developed. This immediately translates into improved product quality, lowered cost, and better adherence to the schedule. These benefits allow companies to more accurately manage projects, leading to an improved ability to predict the outcome of their projects.

Project Planning, and Project Tracking and Oversight are fundamental benefits to formal project management, which comprise two of the CMM® Level 2 key process areas.

Often, project overruns are absorbed by the development company. This usually eats at their profit margins, or worse, results in losses. The immediate effect of implementing these processes is that the companies can track their software projects with confidence. Furthermore, they can better determine the cause of slips and stops, which can often be traced back to something the customers have asked for. However, without a formal *process* to track the source of requirements, a company is left holding the bits. So to speak. Thus,

Requirements Management is a CMM® Level 2 key process area.

Ultimately, the idea is to create such processes that are self-correcting and always improving. Thereby reducing the cost of producing software and being able to retain more of the profit margin built into what they charge their clients. Or, when developing software in house, these efficiencies are simply better products for less time and money.

Quality Assurance: also a Level 2 key process area.

Why You Should Care.

There are two reasons to care about the CMM®.

1. You already read that you can't provide custom software to the Federal government unless you are at least appraised to CMM® Level 2, and for most projects Level 3.

2. Your competition may already be or are working towards applying the CMM®.

You may have honed in on the word "custom" and say, "but I don't plan to provide *custom* software to the Feds, just shrink-wrapped or configurable software." Well, even though you've found the *exception* to the rule, to this I still say: Prove it.

Prove that the software you're

selling the Federal government isn't custom built. Prove that the product they're getting is exactly the same as the product you're selling at Office Depot. For some of you this is easy. Your product is literally shrink-wrapped. But for the majority of you who perform custom installation and configuration of the software, proving that there is no new code in what the government is getting may be a challenge you don't want to face.

What would it mean that custom code is going into the government's product? Well, it means that a little clause in the government's contract with you may quickly turn all that "non-government property" into "purchased product." Especially if you *can't* demonstrate one of two things: (a) that *really*, the code isn't custom, or (b) that yes, the custom parts follow the CMM®. You see where we are back to.

Configuration Management. It's one of the Level 2 key process areas.

Besides, if the CMM® is such a valuable business tool, why resist it? Don't look at it just because the government and other companies say it's a good thing. The 2nd reason to care is: the competition. What does it say about your company if you don't institute best practices in any repeatable way and your competition not only does it, but reaps the benefits of doing it?

Even though the CMM® was developed based on "large and complex" software projects, its business value has been recognized by companies of every size and in every industry. If you aren't working towards it, what would it mean to your market position if your competition is? Companies are using their CMM® appraisals in their marketing strategies. Can you do that?

Here's another reason to consider implementing CMM®: if you are subcontracted to a company who is providing software to the government, they, of course, must be following CMM®. This means that all the parts of the software product that your piece fits into must also be following CMM®. This means that either your client is covering for you and your lack of CMM®, or they're going to ask you to follow it too. Why?

Vendor Management is the 6th of the Level 2 Key Process Areas. So if you were thinking you could get around the CMM® by either hiring or being hired by a company that is assessed to the CMM®, nice try, but no dice. Outsourcing won't get you around the CMM® either.

What Can You Do?

Ok, so now if you are one of the many companies that has not yet seriously looked at the CMM® – either as a value added business tool or as a necessary step towards providing the government with custom software – you may be wondering what you can do. That's actually the easy part.

Go onto the SEI's website (www.sei.cmu.edu) and read up on the CMM®. You can also get a book or two on the subject, starting with *The Capability Maturity Model: Guidelines for Improving the Software Process*, by Mark C, Paulk, et al, 1994. You may want to take an intro to CMM® class offered either by the SEI, or through one of many companies licensed to do so. Training in the CMM® is very important. It isn't like any other standard out there in the software world. One thing that throws off many companies is that there is no explanation of how to *implement* the CMM®. This is where a lead assessor/appraiser will be very helpful.

You can also get a list of lead assessors/appraisers and companies licensed to teach and perform CMM® assessments from the SEI's website. So look for their list of active appraisers (in some places referred to as "lead assessors") and find one that you want to talk to. You can look for them by name, company, or scan through the list and find one local to you. You can call or email one of them directly, or contact the SEI for help. Many, but not all, Lead Assessors are licensed to provide the training, and not all training is licensed by the SEI. You can arrange for the training to come to you with many of the companies offering on-site training. The SEI offers classes only in their locations.

Some lead assessors work for companies and are not available to do assessments or appraisals outside of their company. Others (the majority) are working for themselves or companies that specialize in this field. Speak to one or several and find a person or company that communicates well with you and you feel comfortable working with. You want to hear them care more about your business than about what they can do for you. You also want to get a sense for whether they will leave your company better off or whether they'll tell you what you want to hear for the purpose of getting the assessment done. It takes all kinds, right?

Here's what you can expect: regardless of what CMM® level you are working towards (each level builds on the previous level so, no, you can't skip levels) your company will probably first need to know where it is before it can know where it needs to go. You will want to perform a "gap analysis" against the CMM®. You could do this yourself, once you understand the CMM®, or you can spend some money and have it done for you. The objectivity and value of a 3rd party's experience will probably pay for itself compared to your own frustration and sweat equity.

After you know your starting point, most of you will need to come up with a plan to start developing, formalizing, and implementing your software management processes. This is often called "SPI" for "Software Process Improvement." Again, you can do this yourself. Again, there are experts that can help you. These people wouldn't be in business if there wasn't a need for them. There's probably a good reason they have jobs.

Once you've implemented your SPI plan and have demonstrable proof that it works, you can pursue the appraisal. Officially, this is called the CMM® Appraisal for Internal Process Improvement, or "CBA IPI." This you *cannot* do yourself. If you want the appraisal registered with the SEI and you really want value out of all this work, you need to go get yourself a Lead Assessor to perform the appraisal. Depending on the size of the organization, and the number of projects to be assessed, the appraisal can take about 2 weeks and *does not* guarantee that you will be appraised at the maturity level you seek.

This is why working through the entire CMM® and SPI effort with a company who can bring in a Lead Assessor can be so valuable. They can work with you to ensure a high degree of probability that when you purchase the appraisal service, you will be assessed to the maturity level you seek to demonstrate.

What's Left?

After all that, there's still a few words to be said. The more you understand about the CMM® the more you will see its value to your business. Beyond merely marketing or getting government work, this model is a success-based model. Think about that. *Successful projects do these things.* They are the software industry's "best practices." Whether you pursue a formal CMM® appraisal or not, does your company do these best practices? Do you do them well?

The last thing to say is this: you don't need the CMM® to develop software well. But you do need some form of process management in your business, especially where technology is concerned in order to develop your products well. It doesn't have to be the CMM®, but you do want to take advantage of the industry's best practices in some way.

Many companies are turned off by what seems to be a major bureaucracy required to implement CMM®. The problem isn't that the CMM® is a bureaucracy, it's that many of the companies who implement CMM® and many of the people who appraise companies to the CMM® are used to the bureaucratic approach taken by many "large and complex" software projects. You will want to work with a consultant that will help you implement what will work best for your company, whether it's CMM® or not.

Rest assured, if you don't have a large or complex software project, but you are still interested in just a little process discipline, there are ways to implement the CMM® (specifically) and disciplined processes (in general) that don't create a bureaucracy. After all even, best practices are of questionable value if they blunt your competitive edge.

When you decide that it's time to tighten up your insight into, control over, and predictability of your projects, your ability to deliver products, and your sources of problems that impact product quality, you're in essence looking to implement some form of process discipline. At that point, what you implement and how you implement it must align with your business goals. Don't let anyone tell you that your business positioning or objectives must be sacrificed on the altar of process.

A strong process infrastructure (whether CMM® or not) will allow your business to ride the tide and weather whatever waves pass through.

The author is an independent process management consultant. He brings a broad spectrum of experience in Process Engineering and Management. He is a student of the evolution of process-centered design, development and production from the introduction to the the DoD of TQM, Integrated Product and Process Development, and ISO 9000 through the emergence of CMM® and its subsequent migration to the private sector.

The focus of the author's career is on the issues of product integrity and technology management. He specializes in the management-driven engineering principles of quality, operations, risk, requirements, producibility, configuration and project management. In merging these disciplines with business and operations strategies he emphasizes the importance of thoroughly planned and integrated process management. He's successfully adapted and evolved these disciplines across the Internet, software, and manufacturing industries.

The author can be reached by writing to contact@entinex.com or visiting http://www.entinex.com .

Copyright Notice and Reproduction Permission

Entinex does not guarantee the accuracy or completeness of any information contained herein, nor shall EntinEx be in any way liable for any delay in keeping such information current. EntinEx specifically disclaims all warranties, express or implied, with respect to the use of this information or any results with respect thereto.