Capability Maturity Model

By

Ming Lei

Foundation of Software Engineering

April 22, 2018

Introduction

Today market has a very high demand for cheaper and better software which has to be delivered in a more timely manner. However, software analysis and design is a complex process. Many software projects are unsuccessful due to the low quality of the software, the failure to meet requirements, or the excessive cost to develop and maintain the software. Software companies, organizations, and governments have turned to software process improvement as a way of accelerating software development processes, lowering costs, and enhancing the overall quality (Sommerville, 2010). There are a few software process improvement frameworks are in used today, such as Capability Maturity Model(CMM), and Capability Maturity Model Integration(CMMI). This paper will focus on the Capability Maturity Model (CMM).

The purpose of this paper is to analyze how Capability Maturity Model(CMM) promotes quality. The paper first goes over the summary of CMM and explain the characteristics of each maturity level. Then the key process areas and how they relate to maturity levels are discussed in detail. Finally, the paper discusses the return on investment of advancing through the levels of the CMM by citing real evidence from organizations that have done it.

Capability Maturity Model

Capability Maturity Model, a measure of the effectiveness of a software process, was originally developed by the Software Engineering Institute of Carnegie Mellon University in the 1980s. The initial purpose of CMM is to assist the U.S. Department of Defense to evaluate the contractor performance in software acquisition. The CMM can be used to assess the maturity of software processes of an organization and determine the key practices that are required to increase the maturity of these processes (Kaur, 2014). The goal of CMM is to help software

organizations improve the maturity of processes from chaotic processes to optimized processes. Process improvement means recognizing the current processes' shortages and changing these processes to increase product quality, shorten development time and reduce costs (Sommerville, 2010).

However, CMM is not the only one for process improvement. In the book *Software Engineering (2010),* professor Ian Sommerville points out two different approaches to process improvement are used in industry. The first one is the agile approach. The agile methods are delivery of functionality quickly in response to customers' changing requirements. Professor Ian Sommerville believes agile methods may be the most cost-effective process improvement strategy for small to medium-sized project. But it may not work well for large projects. The large projects usually need many different developers in different departments from various companies. In general, the management issues are the very reasons why projects run into problems. The other one is the process maturity approach, which focus on process management and maturity-based process improvement. This paper will place emphasis on Capability Maturity Model (CMM) because it works well on large complex project. CMM consists of five maturity levels, initial, repeatable, defined, managed, and optimizing (Sommerville, 2010). The following section will discuss the detail of five maturity levels.

Five Maturity Levels

In the book *Introduction to Software Engineering (2016),* professor Ronald Leach at Howard University cites the source from Software Engineering Institute and summarizes the following characteristics for each maturity level. The first level is the initial level. Processes are ad hoc and chaotic in general. Few processes are defined and results are not predictable. The

success of the project primarily depends on individual heroic efforts. Organizations may produce products, but they most likely run over the budget and miss deadlines. The second level is the Repeatable level. Basic project management processes are established for tracking cost, schedule and functionality. Experience and observation of previous project is used for similar project. The projects of the organization have ensured that requirements are managed and processes are planned and controlled. The third level is the Defined level. Software development processes are well standardized, documented and consistent because management processes are established and repeatable. All projects use the well-defined standard and procedure for developing software. The forth level is the Managed level. Statistical and other quantitative techniques are used as criteria in processes. Detailed measures of software process and product quality are collected. Software processes are predictable. The fifth level is the Optimizing level. The major characteristic is the continuous process improvement to enhance process capability and performance. The organization has quantitative feedback systems in order to detect strengths and weakness of project (Leach, 2016).

Key Process Areas of the CMM

Key Process Areas (KPA) describe that functions that must be present to satisfy good practice at each individual maturity level. It indicates the areas that an organization should focus on to improve its process (Kaur, 2014). Besides level 1, all CMM levels contain a set of Key Process Areas as shown in next page Table 1. According to a published journal by professor Kirti Mathur at Devi Ahilya University, KPAs are used to attain prespecified goal for setting up process capability at that maturity level. Each KPA stabilizes an important aspect of the software process when it is satisfied.

Maturity Level	Key Process Areas
Initial (Level 1)	None
Repeatable (Level 2)	Software Configuration Management (SCM)
	Software Quality Assurance (SQA)
	Software subcontract management (SSM)
	Software project tracking and oversight (SPTO)
	Software project planning (SPP)
	Requirements management (RM)
Defined (Level 3)	Peer reviews (PR)
	Inter-group communication (IC)
	Software product engineering (SPE)
	Integrated software management (ISM)
	Training program (TP)
	Organization process definition (OPD)
	Organization process focus. (OPF)
Managed (4)	Software quality management (SQM)
	Quantitative process management (QPM)
Optimizing (5)	Process change management (PCM)
	Technology change management (TCM)
	Defect prevention (DP)

Table 1. The Key Process Areas

The above table summaries all KPAs in relate to each maturity level. Professor Matthew Dwyer at Kansas State University cites the source from Software Engineering Institute and concludes the following key points. At the second level, the KPAs place emphasis on establishing basic project management controls. At the third level, the KPAs focus on project and organizational issues. An infrastructure is established to institutionalize effective software engineering and management processes in all projects. At the forth level, the KPAs cover on establishing a quantitative understanding of the software process being built. At the fifth level, the KPAs focus on the issues that the organization and the process must address to implement continual, measurable software process improvement (Dwyer).

As an organization progresses up on the five-level ladder, it is important that maturity levels should not be skipped. The KPAs in previous maturity level provides a foundation for effective implementation of processes at the next level. In order to move to next level, all organizations have to achieve all the specific and generic goals of the process areas assigned to

the previous levels. In other words, any organization willing to reach certain level has to satisfy all the key practices of that level to make process lasting, effective and repeating (Mathur and Rangwala, 2012). Without satisfaction of KPAs from lower levels, higher level processes have low chance of success.

Return on Investment

Software process improvement is not easy work and it has massive needs for investment. Many companies are concerned about what financial benefits they are reaping from their investment in process improvement. According to Software Engineering Institute of Carnegie Mellon University, there are many ways to define or calculate the Return on Investment, such as Benefit/Cost ratio, Net Present Value and Internal Rate of Value. This paper uses Benefit/Cost ratio formula provided in the book *Software Engineering (2010)* by professor Pressman and Maxim. The return on investment formula is:

$$\text{ROI}=\frac{\sum(benefits)-\sum(costs)}{\sum(costs)}\times 100\%$$

In the above equation, benefits indicate all the cost saving related to higher product quality, such as less rework and reduced effort. Costs involve all direct and indirect software process costs, including but not limit to training employees, hiring new technical staff and changing management activities.

Higher CMM levels usually result in more benefits. First, it can generate a higher return on investment. Second, higher CMM levels make processes repeatable and predictable. Third, it can increase product quality and decrease rework levels. A lot of evidence has proved that large companies and organizations benefited from achieving higher CMM levels. This paper will provide two examples. The first example is based on a research paper published on the *Journal*

of Defense Software Engineering. Mike Diaz, the chief software engineer for the General Dynamics Decision Systems at Arizona, studied the process improvement of 360 software engineers in his company and concluded the following table. The following Return of Investment table (Table 2) shows more cost saving on rework and return on investment they get if they put more investment on process improvement. As you can see, the largest benefit is the transition from Level 2 to 3 with 167% ROI because they invest heavily. The ROI from Level 4 to 5 is not significant as the other level transitions due to the less investment on process improvement.

CMM Level Transition	Cost for SPI in hrs (2.5% of Base)	Cost Savings on Rework (hrs)	Return on Investment
Level 4 to 5	884	1,009	14%
Level 3 to 4	1,310	2,744	109%
Level 2 to 3	2,544	6,806	167%

Table 2: Return on Investment by Level Transitions

The second successful example is Raytheon Equipment Division. According to "Process Improvement and the Corporate Balance Sheet," Raymond Dion, the principal engineer with Raytheon's Equipment Division, points out Raytheon got $7.7 return on every dollar invested in process improvement and rework levels decreased from 41 % to 11% in the transition from Level 1 to 3. In addition, Raymond also found Raytheon saved $4.48 million over six projects in one year.

Conclusion

CMM describes the important discipline and essential management tasks that all organizations have to perform. CMM serves as a road map for all organizations to achieve improved product quality and schedule predictability. The aims of process improvement are reduced cost, higher quality and faster delivery. Before initiating process improvement activities,

it is important to consider the Return of investment as process improvement is not free. To achieve higher CMM levels, large investment on time and money are usually required. In addition, a commitment from management to support process improvement are necessary to be success.

Reference

Dion, R. (1993). Process improvement and the corporate balance sheet. *IEEE Software,10*(4), 28-35.

Diaz, M. (2002). How CMM Impacts Quality, Productivity, Rework, and the Bottom Line. *Journal of Defense Software Engineering.*

Dwyer, M. *Overview of the Capability Maturity Model.* Retrieved April 19, 2018, from Kansas State University website: http://people.cs.ksu.edu/~dwyer/courses/748/resources/cmm-tr25/tr25_o2.html

Kaur. J. (2014). Comparative Study of Capability Maturity Model. *International Journal of Advanced Research in Computer Science & Technology*, 2(1).

Leach, R.J. (2016). *Introduction to Software Engineering.* Boca Raton: CRC Press Taylor & Francis Group.

Mathur, K. and Rangwala, F. (2012). Tailoring KPA of CMM TO Accede with RUP. *International Journal of Advanced Research in Computer Science and Electronics Engineering*, 1(7).

Pressman, R. S. and Maxim, B.R.(2010). *Software engineering: a practitioners approach.* New York: McGraw-Hill Higher Education.

Sommerville, Ian (2010). *Software engineering (10th ed.).* Addison Wesley.