ISSN No. 0976-5697

Volume 5, No. 2, March 2014 (Special Issue)

International Journal of Advanced Research in Computer Science

REVIEW ARTICLE

Available Online at www.ijarcs.info

The Process Improvement Model: CMM

CH.V.Phani Krishna
Computer Science Engineering
KLUniversity
Guntur, India
phanik16@yahoo.co.in

S.Gayatri Anusha
Computer Science Engineering
KLUniversity
Guntur, India
gayatrianusha@gmail.com

G.Lahari
Computer Science Engineering
KLUniversity Guntur, India
lahari.3013@gmail.com

***Abstract*—** The Capability Maturity Model for Software is a model for building organizational capability that has been widely adopted in the software community and beyond. The methodology of this model focuses on determining current process maturity in an organization and then identifying issues that are critical to the software quality and process improvements. Therefore, the CMM supports the theory that focusing on key issues and activities can steadily improve the software process capabilities throughout an organization. This framework is composed of 5 maturity levels that measure the maturity of an organizations software processes and evaluates the software. The Software CMM is a five-level model that describes good engineering and management practices and prescribes improvement priorities for software organizations. This paper stresses the need for a process maturity framework to prioritize improvement actions, describes the process maturity framework of five maturity levels and the associated structural components

Keywords: capability maturity model, CMM, process maturity framework, software process improvement, maturity level, key process area.

I. INTRODUCTION

The Software CMM is intended to be:

A common-sense application of process management and quality improvement concepts to software development and maintenance -- the CMM practices are not rocket science (even the statistical process control concepts at Levels 4 and 5 have been successfully applied in other industries for decades) a community-developed guide -- input from hundreds of software professionals was solicited in developing the current release of the CMM a model for organizational improvement – which implies a set of priorities that may differ from those of any specific project, but which have been proven effective in organizational transformation The underlying structure for reliable and consistent CMM-based appraisal methods -- assessments and evaluations based on the Software CMM are widely used by software organizations for improvement and customers for understanding the risks associated with potential suppliers.

The Capability Maturity Model for Software provides software organizations with guidance on how to gain control of their processes for developing and maintaining software and how to evolve toward a culture of software engineering and management excellence[2]. The CMM was designed to guide software organizations in selecting process improvement strategies by determining current process maturity and identifying the few issues most critical to software quality and process improvement. By focusing on a limited set of activities and working aggressively to achieve them, an organization can steadily improve its organization-wide software process to enable continuous and lasting gains in software process capability[3].

II. THE FIVE LEVELS OF SOFTWARE PROCESS MATURITY

A maturity model can be viewed as a set of structured levels that describe how well the behaviors, practices and processes of an organization can reliably and sustainably produce required outcomes. A maturity model can be used as a benchmark for comparison and as an aid to understanding - for example, for comparative assessment of different organizations where there is something in common that can be used as a basis for comparison[6]. In the case of the CMM, for example, the basis for comparison would be the organizations' software development processes. *A maturity level* is a well-defined evolutionary plateau toward achieving a mature software process. Each maturity level comprises a set of process goals that, when satisfied, stabilize an important component of the software process. Achieving each level of the maturity framework establishes a different component in the software process, resulting in an increase in the process capability of the organization

III. BEHAVIORAL CHARACTERIZATION OF THE MATURITY LEVELS

A. Level 1 - The Initial Level:

At the Initial Level, the organization typically does not provide a stable environment for developing and maintaining software. Such organizations frequently have difficulty making commitments that the staff can meet with an orderly engineering process, resulting in a series of crises. In spite of this ad hoc, even chaotic, process, Level 1 organizations frequently develop products that work, even though they may be over the budget and schedule. Success

CONFERENCE PAPER
Two day National Conference on Advanced Trends and Challenges in Computer Science and Applications
Organized by: Shree Vishnu Engineering College for Women, Bhimavaram A.P.
Schedule: 18-19 March 2014

in Level 1 organizations depends on the competence and heroics of the people in the organization3 and cannot be repeated unless the same competent individuals are assigned to the next project. Thus, at Level 1, capability is a characteristic of the individuals, not of the organization. It is characteristic of processes at this level that they are undocumented and in a state of dynamic change, tending to be driven in an *ad hoc*, uncontrolled and reactive manner by users or events. This provides a chaotic or unstable environment for the processes.

B. Level 2 - The Repeatable Level:

At the Repeatable Level, policies for managing a software project and procedures to implement those policies are established. Planning and managing new projects is based on experience with similar projects. Process capability is enhanced by establishing basic process management discipline on a project by project basis[4]. An effective process can be characterized as one which is practiced, documented, enforced, trained, measured, and able to improve.

The software process capability of Level 2 organizations can be summarized as disciplined because planning and tracking of the software project is stable and earlier successes can be repeated. The project's process is under the effective control of a project management system, following realistic plan based on the performance of previous projects. It is characteristic of processes at this level that some processes are repeatable, possibly with consistent results. Process discipline is unlikely to be rigorous, but where it exists it may help to ensure that existing processes are maintained during times of stress.

C. Level 3 - The Defined Level:

At the Defined Level, the standard process for developing and maintaining software across the organization is documented, including both software Engineering and management processes, and these processes are integrated into a coherent whole. This standard process is referred to throughout the CMM as the organization's standard software process. Processes established at Level 3 are used (and changed, as appropriate) to help the software managers and technical staff perform more effectively. The organization exploits effective software engineering practices when standardizing its Software processes.

The software process capability of Level 3 organizations can be summarized as standard and consistent because both software engineering and management activities are stable and repeatable. Within established product lines, cost, schedule, and functionality are under control, and software quality is tracked. This process capability is based on a common, organization-wide understanding of the activities, roles, and responsibilities in a defined software process.

D. Level 4 - The Managed Level:

At the Managed Level, the organization sets quantitative quality goals for both software products and processes. Productivity and quality are measured for important software process activities across all projects as part of an organizational measurement program. An organization-wide software process database is used to collect and analyze the data available from the projects' defined software processes. Software processes are instrumented with well-defined and consistent measurements at Level 4. These measurements establish the quantitative foundation for evaluating the projects' software processes and products.

The software process capability of Level 4 organizations can be summarized as being quantifiable and predictable because the process is measured and operates within measurable limits. This level of process capability allows an organization to predict trends in process and product quality within the quantitative bounds of these limits. Because the process is both stable and measured, when some exceptional circumstance occurs, the "special cause" of the variation can be identified and addressed. When the known limits of the process are exceeded, action is taken to correct the situation. Software products are of predictably high quality[5]s.

E. Level 5 - The Optimizing Level:

At the Optimizing Level, the entire organization is focused on continuous process improvement. The organization has the means to identify weaknesses and strengthen the process proactively, with the goal of preventing the occurrence of defects. Data on the effectiveness of the software process is used to perform cost benefit analyses of new technologies and proposed changes to the organization's software process. Innovations that exploit the best software engineering practices are identified and transferred throughout the organization.

Software project teams in Level 5 organizations analyze defects to determine their causes. Software processes are evaluated to prevent known types of defects from recurring, and lessons learned are disseminated to other projects

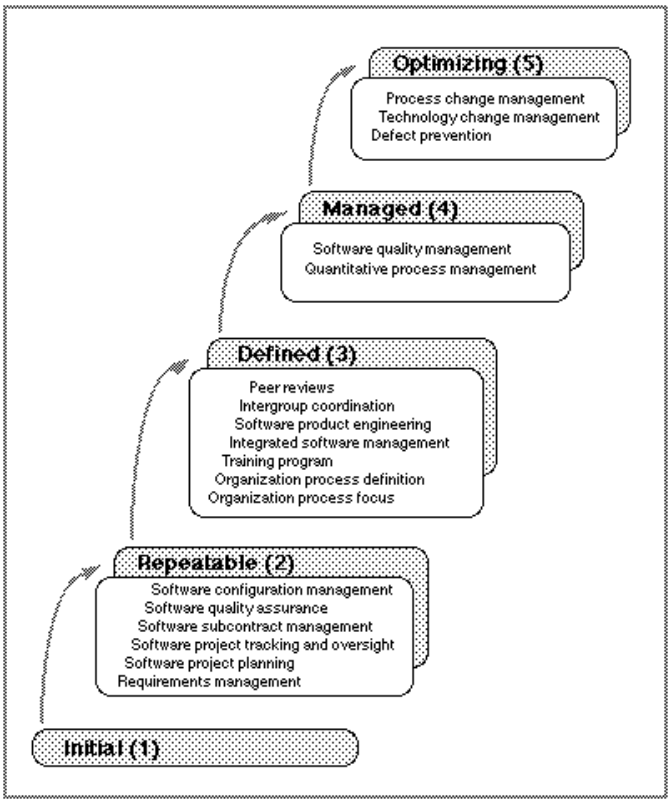

Figure: 1 The Key Process Areas by Maturity Level

IV. STRUCTURE OF THE MODEL

The model involves five aspects:

a. ***Maturity Levels:*** a 5-level process maturity continuum - where the uppermost (5th) level is a notional ideal state where processes would be systematically managed by a combination of process optimization and continuous process improvement.

b. ***Key Process Areas:*** a Key Process Area identifies a cluster of related activities that, when performed together, achieve a set of goals considered important.

c. ***Goals:*** the goals of a key process area summarize the states that must exist for that key process area to have been implemented in an effective and lasting way. The extent to which the goals have been accomplished is an indicator of how much capability the organization has established at that maturity level. The goals signify the scope, boundaries, and intent of each key process area.

d. ***Common Features:*** common features include practices that implement and institutionalize a key process area. There are five types of common features: commitment to perform, ability to perform, activities performed, measurement and analysis, and verifying implementation.

e. ***Key Practices:*** The key practices describe the elements of infrastructure and practice that contribute most effectively to the implementation and institutionalization of the area.

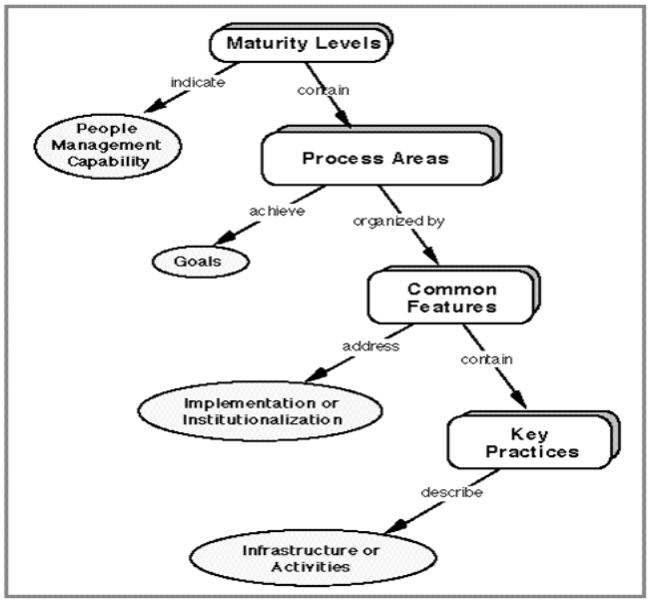

Figure: 2 The CMM Structure

A. *Key Process Areas:*

Each key process area identifies a cluster of related activities that, when performed collectively, achieve a set of goals considered important for enhancing process capability [7]. The path to achieving the goals of a key process area may differ across projects based on differences in application domains or environments. Nevertheless, all the goals of a key process area must be achieved for the organization to satisfy that key process area.

The key process areas may be considered the requirements for achieving a maturity level. To achieve a maturity level, the key process areas for that level must be satisfied.

The specific practices to be executed in each key process area will evolve as the organization achieves higher levels of process maturity. For instance, many of the project estimating capabilities described in the Software Project Planning key process area at Level 2 must evolve to handle the additional project data available at Level 3, as is described in Integrated Software Management.

The key process areas at Level 2 focus on the software project's concerns related to establishing basic project management controls.

a. The purpose of *Requirements Management* is to establish a common understanding between the customer and the software project of the customer's requirements that will be addressed by the software project .This agreement with the customer is the basis for planning and managing the software project.

b. The purpose of *Software Project Planning* is to establish reasonable plans for performing the software engineering and for managing the software project. These plans are the necessary foundation for managing the software project.

c. The purpose of *Software Project Tracking and Oversight* is to establish adequate visibility into actual progress so that management can take effective actions when the software project's performance deviates significantly from the software plans.

d. The purpose of *Software Subcontract Management* is to select qualified software subcontractors and manage them effectively.

e. The purpose of *Software Quality Assurance* is to provide management with appropriate visibility into the process being used by the software project and of the products being built.

f. The purpose of *Software Configuration Management* is to establish and maintain the integrity of the products of the software project throughout the project's software life cycle.

***The key process areas at Level* 3** addresses both project and organizational issues, as the organization establishes an infrastructure that institutionalizes effective software engineering and management processes across all projects.

g. The purpose of *Organization Process Focus* is to establish the organizational responsibility for software process activities that improve the organization's overall software process capability.

h. The purpose of *Organization Process Definition* is to develop and maintain a usable set of software process assets that improve process performance across the projects and provide a basis for defining meaningful data for quantitative process management. These assets provide a stable foundation that can be institutionalized via mechanisms such as training.

i. The purpose of *Training Program* is to develop the skills and knowledge of individuals so they can perform their roles effectively and efficiently. Training is an organizational responsibility, but the software projects should identify their needed skills and provide the necessary training when the project's needs are unique.

j. The purpose of *Integrated Software Management* is to integrate the software engineering and management

activities into a coherent, defined software process that is tailored from the organization's standard software process and related process assets. This tailoring is based on the business environment and technical needs of the project.

k. The purpose of *Software Product Engineering* is to consistently perform a well-defined engineering process that integrates all the software engineering activities to produce correct, consistent software products effectively and efficiently. Software Product Engineering describes the technical activities of the project, e.g., requirements analysis, design, code, and test.

l. The purpose of *Intergroup Coordination* is to establish a means for the software engineering group to participate actively with the other engineering groups so the project is better able to satisfy the customer's needs effectively and efficiently.

m. The purpose of *Peer Reviews* is to remove defects from the software work products early and efficiently. An important corollary effect is to develop a better understanding of the software work products and of the defect that can be prevented. The peer review is an important and effective engineering method that can be implemented via inspections, structured walkthroughs, or a number of other collegial review methods.

The key process areas at Level 4 focus on establishing a quantitative understanding of both the software process and the software work products being built.

n. The purpose of *Quantitative Process Management* is to control the process performance of the software project quantitatively. Software process performance represents the actual results achieved from following a software process. The focus is on identifying special causes of variation within a measurably stable process and correcting, as appropriate, the circumstances that drove the transient variation to occur.

o. The purpose of *Software Quality Management* is to develop a quantitative understanding of the quality of the project's software products and achieve specific quality goals.

p. ***The key process areas at Level 5*** cover the issues that both the organization and the projects must address to implement continuous and measurable software process improvement.

The purpose of *Defect Prevention* is to identify the causes of defects and prevent them from recurring. The software project analyzes defects, identifies their causes, and changes its defined software process.

q. The purpose of *Technology Change Management* is to identify beneficial new technologies (i.e., tools, methods, and processes) and transfer them into the organization in an orderly manner .The focus of Technology Change Management is on performing innovation efficiently in an ever-changing world.

r. The purpose of *Process Change Management* is to continually improve the software processes used in the organization with the intent of improving software quality, increasing productivity, and decreasing the cycle time for product development.

B. Common Features:

For convenience, the practices that describe the key process areas are organized by common features[8]. The common features are attributes that indicate whether the implementation and institutionalization of a key process area is effective, repeatable, and lasting. The five common features are:

a. Commitment to Perform: Commitment to Perform describes the actions the organization must take to ensure that the process is established and will endure. Commitment to Perform typically involves establishing organizational policies and senior management sponsorship.

b. Ability to Perform: Ability to Perform describes the preconditions that must exist in the project or organization to implement the software process competently. Ability to Perform typically involves resources, organizational structures, and training.

c. Activities Performed: Activities Performed describes the roles and procedures necessary to implement a key process area. Activities Performed typically involve establishing plans and procedures, performing the work, tracking it, and taking corrective actions as necessary.

d. Measurement and Analysis: Measurement and Analysis describes the need to measure the process and analyze the measurements. Measurement and Analysis typically includes examples of the measurements that could be taken to determine the status and effectiveness of the Activities Performed.

e. Verifying Implementation: Verifying Implementation describes the steps to ensure that the activities are performed in compliance with the process that has been established. Verification typically encompasses reviews and audits by management and software quality assurance.

The practices in the common feature Activities Performed describe what must be implemented to establish a process capability. The other practices, taken as a whole, form the basis by which an organization can institutionalize the practices described in the Activities Performed common feature.

C. Key Practices:

Each key process area is described in terms of the key practices that contribute to satisfying its goals. The key practices describe the infrastructure and activities that contribute most to the effective implementation and institutionalization of the key process area. Each key practice consists of a single sentence, often followed by a more detailed description, which may include examples and elaboration. These key practices, also referred to as the top-level key practices, state the fundamental policies, procedures, and activities for the key process area. The components of the detailed description are frequently referred to as sub practices. The key practices describe "what" is to be done, but they should not be interpreted as mandating "how" the goals should be achieved. Alternative practices may accomplish the goals of the key process area.

The key practices should be interpreted rationally to judge whether the goals of the key process area are effectively, although perhaps differently, achieved. The key Practices are contained in the "Key Practices of the Capability Maturity Model,

CONFERENCE PAPER
Two day National Conference on Advanced Trends and Challenges in Computer Science and Applications
Organized by: Shree Vishnu Engineering College for Women, Bhimavaram A.P.
Schedule: 18-19 March 2014

V. CONCLUSION

The CMM represents a "common sense engineering" approach to software process improvement. The maturity levels, key process areas, common features, and key practices have been extensively discussed and reviewed within the software community. While the CMM is not perfect, it does represent a broad consensus of the software community and is a useful tool for guiding software process improvement efforts.

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 adequately resolved. However, current reports from CMM-based improvement programs indicate that it can improve the likelihood with which a software organization can achieve its cost, quality, and productivity goals.[1]..The CMM identifies practices for a mature software process and provides examples of the state-of-the-practice (and in some cases, the state-of-the-art), but it is not meant to be either exhaustive or dictatorial. The CMM identifies the characteristics of an effective software process, but the mature organization addresses all issues essential to a successful project, including people and technology, as well as process

VI. ACKNOWLEDGMENT

We would like to acknowledge the support and guidance of our supervisor Ch.V.Phani Krishna. This paper has been possible because of his trust and confidence in our work. He has always encouraged, supported, corrected and guided us during the paper preparation. The paper has been learning and growing experience for us. We would like to thank the all the professors for all the support that they have provided us in response.

VII. REFERENCES

[1] Dion92 Raymond Dion, "Elements of a Process-Improvement Program," IEEE Software, Vol. 9, No. 4,July 1992, pp. 83-85.

[2] Fowler90 P. Fowler and S. Rifkin, Software Engineering Process

[3] Group Guide, Software Engineering Institute, CMU/SEI-90-

[4] TR-24, ADA235784, September, 1990

[5] Humphrey87a W.S. Humphrey, Characterizing the Software Process

[6] Maturity Framework, Software Engineering Institute,

[7] CMU/SEI-87-TR-11, ADA182895, June 1987. Also

[8] Published in IEEE Software, Vol. 5, No. 2, March 1988,pp.73-79.