HealthManagement
Promoting Management and Leadership

Volume 2 / Issue 1 2007 - Management

Capability Maturity Model (CMM)

Author

Catalina Ciolan

is Project Director of the European Association of Healthcare IT Managers in Brussels,

Belgium.

The Capability Maturity Model (CMM) is a process capability maturity model which helps in the definition and understanding of an organisation's processes. Initially known as Humprey's CMM (name given after IT guru Watts

Humprey), it has been actively developed by the SEI (US Department of Defense- backed Software Engineering

Institute at Carnegie-Mellon University) since 1986.

When conceived, the CMM model was intended to act as a tool for assessing the ability of government contractors' processes to perform a contracted software project. Besides this initial goal, CMM has been - and is – applied as a model to assist in understanding the process capability maturity of organisations in various areas such as software engineering, system engineering, project management, software maintenance, risk management, system acquisition, information technology (IT), personnel management.

The CMM model was first released in 1990. Although many organisations found these models to be useful, they faced problems caused by overlap, inconsistencies, and integration. Many organisations also confronted conflicting demands between these models and ISO 9001 audits or other process improvement programs.

Structure of the CMM

The CMM presents the following characteristics: 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.

Key Process Areas: A Key Process Area (KPA) identifies a cluster of related activities that, when performed collectively, achieve a set of goals considered important.

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.

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.

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

Five Levels of Software Process Maturity

The CMM defines five levels of software process maturity, based on an organisation's support for certain key process areas (KPAs).

Level 1 (initial) describes an organisation with an immature or undefined process. This provides a chaotic or unstable environment for the

processes. As a result, process performance in such organisations is likely to be variable (inconsistent) and depend heavily on past practices and traditions, the institutional knowledge, etc.

Level 2 (repeatable). It includes requirements management; software project planning; software project tracking and oversight; software subcontract management; software quality assurance; software configuration management.

Level 3 (defined). It focuses on organisational process definition, training programs, integrated software management, software product engineering, intergroup coordination, peer reviews. At this level, process management starts to occur using defined documented processes, with mandatory process objectives, and ensures that these objectives are appropriately addressed.

Level 4 (managed) includes process measurement and analysis; quality management; defect prevention. It is characteristic of processes at this level that, using process metrics, management can effectively control the ASIS process (e.g., for software development).

Level 5 (optimizing) describes organisations with successively higher levels of software process maturity and includes technology innovation, process change management. If at maturity

Level 4, processes are addressing special statistical causes of process variation and are also providing statistical predictability of the results, at maturity

Level 5, processes are addressing the underlying causes of process variation and are changing the process to improve process performance. For most organisations the key goal is to achieve a Level 3 maturity. One tool for assessing an organisation's current maturity level is a software capability evaluation (SCE), that evaluates its software process (usually in the form of policy statements) and project practices. However, one of the key issues with the CMM is that it overemphasizes peer reviews, inspections, and traditional Quality Assurance "policing" methods. On the other hand it is also believed that there is no emphasis on the architecting/design process, assessment process, or deployment process, all of which have proven to be key discriminators for project success. Some of these are known to be driven by artisanal/ 'feel' factors, which often produce the breakthroughs that really count.

From CMM to CMMI

Although the CMM model proved useful to many organisations, the use of multiple models has been problematic. Applying multiple models that are not integrated within and across an organisation could be costly in terms of training and improvement activities. As a consequence, the CMM Integration (CMMI) project was formed to sort out the challenge of using multiple CMMs. The source models that served as the basis for the CMMI include:

-CMM for Software V2.0 (Draft C) -EIA-731 Systems Engineering, and -IPD CMM (IPD) V0.98a while the old CMM was renamed to Software Engineering CMM (SE-CMM) and organisations accreditations based on SECMM expired on 31 December 2007. The combination of these models into a single improvement framework was intended for use by organisations in their pursuit of enterprise-wide process improvement. These three source models were selected because of their widespread adoption in the software and systems engineering communities and due to their different

approaches to improving processes in an organisation.

Although the CMMI remains an activity- based approach (and this is a fundamental flaw), it does integrate many of the industry's modern best practices, and it discourages much of the default alignment with the waterfall mentality. CMMI consists of best practices that address product development and maintenance. It addresses practices that cover the product's life cycle from conception through delivery and maintenance. There is an emphasis on both systems engineering and software engineering and the integration necessary to build and maintain the total product.

The Relationship Between Six Sigma to CMMI

The relation of Six Sigma for Software to CMMI/PSP/TSP can be best understood as a difference in level of abstraction. Six Sigma for Software might be used to objectively evaluate the overall effect of CMMI on software product quality, cost, and cycle time as compared to an alternative approach, perhaps one of the 'agile' process definitions such as Extreme Programming or Ken Schwaber's "Scrum" (Schwaber and Beadle 2001).

The relation of Six Sigma for Software to CMMI might also be characterised as a difference in goals, in which the goals of CMMI may be a subset of those associated with Six Sigma for Software. Therefore:

-The primary goals of CMMI are continuous improvement in the performance of software development teams in terms of software product cost,

cycle time, and delivered quality;

-The goals of Six Sigma for Software may include the goals of CMMI, but do not specify any particular process definition to achieve those goals. In addition, Six Sigma for Software may be used to achieve many other business objectives (e.g. improved customer service after delivery of the software, or improved customer satisfaction and value realization from the software product delivered). Six Sigma for Software applies to the software process, he software product, and to balancing the needs of the customer to the needs of business in order to maximize overall business value resulting from processes and products.

-An additional difference is that Six Sigma is being applied to selected projects, while CMMI is intended for all projects. Six Sigma may, for example, be used to plan and evaluate pilot implementation of CMMI, while CMMI can provide a defined tool to institutionalise the lessons learned from Six Sigma projects.

CMM and Six Sigma address a challenge posed specifically by hospitals (rather than many other businesses), where every department has specialised needs and working practices – as do different actors and users (nurses, physicians, pathologists and administrators).

Conclusion

For IT managers, the advantage of using these structured methodologies lies in optimizing both existing business and IT processes as well as producing a tangible roadmap for ‘systemically’ inspiring best practices. Meanwhile, the in-built metrics of CMM and Six Sigma allows hospital administrators to identify and ameliorate organizational bottlenecks and measure (continuous) improvements in process efficiencies over time.

Published on : Fri, 1 Feb 2008