CMMI versus XP (eXtreme Programming)

Orhan KALAYCI
President, Nitelik SW Process Consultancy
Orhan.kalayci@nitelik.net

Özet: Evet, CMMI modelinin yazılım süreçleri ile ilgilendiği doğrudur. Peki, yazılım süreçleri CMMI modeli ile ilgilenmeli midir? Cevap basitçe: "Hayır." Yazılım süreçlerinin ilgilenmeleri gereken tek şey iş sonuçlarıdır. Diğer bir deyişle, daha az hata, daha az yeniden aynı işlerin yapılması, daha az maliyet, daha fazla fonksiyonellik ve bütün bunların daha kısa zamanda yapılması böylece daha fazla müşteri memnuniyeti yaratılmasıdır.

Bu noktada, XP tanımlanırken göz alınan hedefler düşünüldüğünde çok şey vaad etmektedir: Proje risklerinin azaltılması, iş değişikliklerine hızlı uyum, sistemin yaşamı boyunca verimliliğin arttırılması, takım halinde yazılım geliştirmeyi eğlenceli hale getirilmesi. Bunlar çok iddialı hedefler olarak görülebilirler, ancak XP'nin hedefledikleri bunlardır.

CMMI ve XP tanımları ve pratikleri arasında bir çelişki var mıdır? Cevap: "çelişki şart değildir." Bu makale CMMI ile bir yazılım şirketinin iş hedeflerine ulaşmak için tasarladığı süreçler arasındaki ilişkiyi açıklamayı hedeflemektedir.

Abstract: Yes, it is true that CMM(I) is interested in SW processes. We may ask then whether SW processes should be interested in CMM(I). The answer is simply "No." What SW processes have to take into account should be nothing but business results. That is to say, Less defects, less rework, less cost, more functionality in shorter durations and finally satisfied customers. .

At this point, XP is very promising as it is defined with the following aim in mind: To reduce project risk, improve responsiveness to business changes, improve productivity throughout the life of a system, and add fun to building software in teams; all at the same time. It might be considered as a challenging target but it is what XP is targeted.

Is there any conflict with CMMI and the definition and practices of XP? The answer is "Not necessarily." This paper aims at explaining the relationship between CMMI and processes used (e.g. XP) by the SW organizations to reach business results.

1. Introduction

When DOD had experienced Software Crisis (as defined in a NATO conference) in 1970s. A solution was required to solve the problem. The problem was: Software projects were almost always late, over budget, less or none functional.

CMM then introduced early 1990s to overcome these obstacles by distinguishing capable SW companies from others. CMM was not aimed at describing a SW process for companies to follow. Instead it was aimed at being a Meta model to assess the capabilities (by defining the necessary attributes) of any SW organization. These attributes described in CMM do not imply and/or impose any SW lifecycle.

XP is a new way of SW development. It aims at reducing project risk, improving responsiveness to business changes, improving productivity throughout the life of a system, and adding fun to building software in teams. As being a kind of new SW lifecycle it might be questioned whether there are any contradiction between XP and the requirements of CMM. There may or may not contradiction between the two. It depends on the implementation of the XP, contradiction is not directly coming from definition.

2. XP

XP aims at overcoming the Basic Problem: Risk

Software development fails to deliver, and fails to deliver value. This failure has huge economic and human impact. We need to find a new way to develop software.

The basic problem of software development is risk. Here are some examples of risk:

- **Schedule slips**; the day for delivery comes, and you have to tell the customer that the software won't be ready for another six months.
- **Project canceled**; after numerous slips, the project is canceled without ever going into production.
- **System goes sour**; the software is successfully put into production, but after a couple of

years the cost of making changes or the defect rate rises so much that the system must be replaced.

- **Defect rate**; the software is put into production, but the defect rate is so high that it isn't used.
- **Business misunderstood**; the software is put into production, but it doesn't solve the business problem that was originally posed.
- **Business changes**; the software is put into production, but the business problem it was designed to solve was replaced six months ago by another, more pressing, business problem.
- **False feature rich**; the software has a host of potentially interesting features, all of which were fun to program, but none of which makes the customer much money.
- **Staff turnover**; after two years, all the good programmers on the project begin to hate the program and leave.

Extreme Programming (XP), a software development discipline that addresses risk at all levels of the development process. XP is also very productive, produces high-quality software, and is a lot of fun to execute.

How does XP address the risks listed above?

- Schedule slips; XP calls for **short release cycles**, a few months at most, so the scope of any slip is limited. Within a release, XP uses one- to four-week iterations of customer-requested features for fine-grained feedback about progress. Within an iteration, XP plans with one-to three-day tasks, so the team can solve problems even during an iteration. Finally, XP calls for implementing the highest priority features first, so any features that slip past the release will be of lower value.
- Project canceled; XP asks the customer to choose **the smallest release that makes the most business sense,** so there is less to go wrong before going into production and the value of the software is greatest.
- System goes sour; XP creates and maintains a **comprehensive suite of tests,** which are run and re-run after every change (several times a day), to ensure a quality baseline. XP always keeps the system in prime condition. Cruft is not allowed to accumulate.
- Defect rate; XP tests from the perspective of both programmers writing **tests** function-by-function and customers writing tests program-feature-by-program-feature.
- Business misunderstood; XP calls for **the customer to be an integral part of the team.** The specification of the project is continuously refined during development, so learning by the customer and the team can be reflected in the software.
- Business changes; XP **shortens the release cycle**, so there is less change during the development of a single release. During a release, the customer is welcome to substitute new functionality for functionality not yet completed. The team doesn't even notice if it is working on newly discovered functionality or features defined years ago.
- False feature rich; XP insists that **only the highest priority tasks** are addressed.
- Staff turnover; XP asks programmers to accept responsibility for estimating and completing their own work, gives them feedback about the actual time taken so their estimates can improve, and respects those estimates. **The rules for who can make and change estimates are clear.** Thus, there is less chance for a programmer to get frustrated by being asked to do the obviously impossible. XP also encourages human contact among the team, reducing the loneliness that is often at the heart of job dissatisfaction by **pair programming**. Finally, XP incorporates an explicit model of staff turnover. New team members are encouraged to gradually accept more and more responsibility, and are assisted along the way by each other and by existing programmers.

3. CMMI

Since 1991, CMMs have been developed for a large number of disciplines. Some of the most notable include models for systems engineering, software engineering, software acquisition, workforce management and development, and Integrated Product and Process Development.

Although these models have proven useful to many organizations, the use of multiple models has been problematic. Many organizations would like to focus their improvement efforts across the disciplines within their organizations. However, the differences among these discipline-specific models, including their architecture, content, and approach, have limited these organizations' ability to focus

their improvements successfully. Further, applying multiple models that are not integrated within and across an organization becomes more costly in terms of training, appraisals, and improvement activities. A set of integrated models that successfully addresses multiple disciplines and has integrated training and appraisal support solves these problems.

The CMM-Integration$_{SM}$ project was formed to sort out the problem of using multiple CMMs. The CMMI Product Team's mission was to combine three source models—

(1) Capability Maturity Model for Software (SW-CMM) v2.0 draft C,

(2) Electronic Industries Alliance Interim Standard (EIA/IS) 731, and

(3) Integrated Product Development Capability Maturity Model (IPD-CMM)$_{SM}$

into a single improvement framework for use by organizations pursuing enterprise-wide process improvement.

3.1. Maturity Levels

The maturity level of an organization provides a way to predict the future performance of an organization within a given discipline or set of disciplines. Experience has shown that organizations do their best when they focus their process-improvement efforts on a manageable number of process areas that require increasingly sophisticated effort as the organization improves.

A maturity level is a defined evolutionary plateau of process improvement. Each maturity level stabilizes an important part of the organization's processes.

In CMMI models with a staged representation, there are five maturity levels, each a layer in the foundation for ongoing process improvement, designated by the numbers 1 through 5:

1. Initial
2. Managed
3. Defined
4. Quantitatively Managed
5. Optimizing

3.2. Maturity Level Details

Maturity levels consist of a predefined set of process areas. The maturity levels are measured by the achievement of the specific and generic goals that apply to each predefined set of process areas. The following sections describe the characteristics of each maturity level in detail.

3.3. Maturity Level 1: Initial

At maturity level 1, processes are usually ad hoc and chaotic. The organization usually does not provide a stable environment. Success in these organizations depends on the competence and heroics of the people in the organization and not on the use of proven processes. In spite of this ad hoc, chaotic environment, maturity level 1 organizations often produce products and services that work; however, they frequently exceed the budget and schedule of their projects.

Maturity level 1 organizations are characterized by a tendency to over commit, abandon processes in the time of crisis, and not be able to repeat their past successes.

3.4. Maturity Level 2: Managed

At maturity level 2, an organization has achieved all the specific and generic goals of the maturity level 2 process areas. In other words, the projects of the organization have ensured that requirements are managed and that processes are planned, performed, measured, and controlled.

The process discipline reflected by maturity level 2 helps to ensure that existing practices are retained during times of stress. When these practices are in place, projects are performed and managed according to their documented plans.

At maturity level 2, requirements, processes, work products, and services are managed. The status of the work products and the delivery of services are visible to management at defined points (for

example, at major milestones and at the completion of major tasks).

Commitments are established among relevant stakeholders and are revised as needed. Work products are reviewed with stakeholders and are controlled. The work products and services satisfy their specified requirements, standards, and objectives.

3.5. Maturity Level 3: Defined

At maturity level 3, an organization has achieved all the specific and generic goals of the process areas assigned to maturity levels 2 and 3. At maturity level 3, processes are well characterized and understood, and are described in standards, procedures, tools, and methods.

The organization's set of standard processes, which is the basis for maturity level 3, is established and improved over time. These standard processes are used to establish consistency across the organization. Projects establish their defined processes by tailoring the organization's set of standard processes according to tailoring guidelines.

The organization's management establishes process objectives based on the organization's set of standard processes and ensures that these objectives are appropriately addressed.

A critical distinction between maturity level 2 and maturity level 3 is the scope of standards, process descriptions, and procedures. At maturity level 2, the standards, process descriptions, and procedures may be quite different in each specific instance of the process (for example, on a particular project). At maturity level 3, the standards, process descriptions, and procedures for a project are tailored from the organization's set of standard processes to suit a particular project or organizational unit. The organization's set of standard processes includes the processes addressed at maturity level 2 and maturity level 3. As a result, the processes that are performed across the organization are consistent except for the differences allowed by the tailoring guidelines.

Another critical distinction is that at maturity level 3, processes are typically described in more detail and more rigorously than at maturity level 2. At maturity level 3, processes are managed more proactively using an understanding of the interrelationships of the process activities and detailed measures of the process, its work products, and its services.

3.6. Maturity Level 4: Quantitatively Managed

At maturity level 4, an organization has achieved all the specific goals of the process areas assigned to maturity levels 2, 3, and 4 and the generic goals assigned to maturity levels 2 and 3. Subprocesses are selected that significantly contribute to overall process performance. These selected subprocesses are controlled using statistical and other quantitative techniques.

Quantitative objectives for quality and process performance are established and used as criteria in managing processes. Quantitative objectives are based on the needs of the customer, end users, organization, and process implementers. Quality and process performance are understood in statistical terms and are managed throughout the life of the processes.

For these processes, detailed measures of process performance are collected and statistically analyzed. Special causes of process variation are identified and, where appropriate, the sources of special causes are corrected to prevent future occurrences.

Quality and process performance measures are incorporated into the organization's measurement repository to support fact-based decision making in the future.

A critical distinction between maturity level 3 and maturity level 4 is the predictability of process performance. At maturity level 4, the performance of processes is controlled using statistical and other quantitative techniques, and is quantitatively predictable. At maturity level 3, processes are only qualitatively predictable.

3.7. Maturity Level 5: Optimizing

At maturity level 5, an organization has achieved all the specific goals of the process areas assigned to maturity levels 2, 3, 4, and 5 and the generic goals assigned to maturity levels 2 and 3. Processes are continually improved based on a quantitative understanding of the common causes of variation inherent in processes.

Maturity level 5 focuses on continually improving process performance through both incremental and innovative technological improvements. Quantitative process-improvement objectives for the organization are established, continually revised to reflect changing business objectives, and used as criteria in managing process improvement. The effects of deployed process improvements are

measured and evaluated against the quantitative process-improvement objectives. Both the defined processes and the organization's set of standard processes are targets of measurable improvement activities.

Process improvements to address common causes of process variation and measurably improve the organization's processes are identified, evaluated, and deployed. Improvements are selected based on a quantitative understanding of their expected contribution to achieving the organization's process-improvement objectives versus the cost and impact to the organization. The performance of the organization's processes is continually improved.

Optimizing processes that are agile and innovative depends on the participation of an empowered workforce aligned with the business values and objectives of the organization. The organization's ability to rapidly respond to changes and opportunities is enhanced by finding ways to accelerate and share learning. Improvement of the processes is inherently part of everybody's role, resulting in a cycle of continual improvement.

A critical distinction between maturity level 4 and maturity level 5 is the type of process variation addressed. At maturity level 4, processes are concerned with addressing special causes of process variation and providing statistical predictability of the results. Though processes may produce predictable results, the results may be insufficient to achieve the established objectives. At maturity level 5, processes are concerned with addressing common causes of process variation and changing the process (that is, shifting the mean of the process performance) to improve process performance (while maintaining statistical predictability) to achieve the established quantitative process-improvement objectives.

4. CMMI versus XP

XP is a new SW lifecycle developed to overcome vital obstacles of conventional ways of SW development. A SW organization can decide to use XP as its SW development lifecycle and at the same time, CMMI can be used to optimize the implementation of XP in the organization

XP is not an alternative to the any one of the CMMI process Areas. The process areas defined in the CMMI are there to develop an appropriate environment to take the SW lifecycle (e.g. XP) under control so that the SW lifecycle to be defined, measured, and optimized.

Reference (Paulk 2001) shows the relationship between CMM and XP explicitly. Although CMMI is a different from CMM, the main framework is the same so that comparing XP with CMM gives us a good insight about relationship between CMMI and XP.

XP satisfaction of key process areas, given the appropriate environment

Level	Key process area	Satisfaction
2	Requirements management	++
2	Software project planning	++
2	Software project tracking and oversight	++
2	Software subcontract management	—
2	Software quality assurance	+
2	Software configuration management	+
3	Organization process focus	+
3	Organization process definition	+
3	Training program	—
3	Integrated software management	—
3	Software product engineering	++
3	Intergroup coordination	++
3	Peer reviews	++
4	Quantitative process management	—
4	Software quality management	—
5	Defect prevention	+
5	Technology change management	—
5	Process change management	—

\+ *Partially addressed in XP*
++ *Largely addressed in XP (perhaps by inference)*
— *Not addressed in XP*

Table 1 – Comparing XP and CMM (Paulk 2001)

5. Conclusion

CMMI process areas describe how a SW organization can get to a CMMI level so that the processes used in the SW organization can be put under control by making the processes as defined, measured and optimized. The process areas defined in the CMMI help organization to reach the optimized processes in a way that independent from the processes used in the organization; be it XP, or any other SW lifecycle such as waterfall, spiral, RUP, etc. The paper "Extreme Programming from a CMM perspective" by Mark C. Paulk in 2001 shows that when we assess a SW organization uses just XP practices against CMM model we will find that it is very close to CMM level 3.

6. References

(Beck 1999) Kent Beck "Extreme programming explained," Addison-Wesley, 1999.

(SEI, 2002) CMMI Product Team , CMMI for Software Engineering (CMMI-SW, V1.1) Staged Representation CMU/SEI-2002-TR-029 ESC-TR-2002-029, SEI, Carnegie Mellon University

(Paulk 2001) Mark C. Paulk, "Extreme Programming from a CMM Perspective," IEEE Software, November 2001

(Kalayci, 2004) Orhan Kalaycı, et al, "Real life experience: CMMI and XP together, Pilot Project," PSQT North 2004, Minneapolis, USA.

Orhan KALAYCI

BS`91 in Computer Engineering from Boğaziçi University, MS`95 from Industrial Engineering from Boğaziçi University. MS Thesis title was "SW Process Assessment and Application in Turkish SW Industry." Work experience: TUBITAK MRC (NATO Project) as SW Engineer for two years, IEEE 730 applied in the project (total budget of 11 Million USD) Turkish Military of Defence R&D dept. (Military obligation), YKB Teknoloji (BILPA), Escort Yazilim, Alcatel (Helped company to reach CMM L3), Founder & President of www.nitelik.net