

Agile SW-Development within a QM System

Abstract

Agility is IN, heavy methods are OUT. There is a big hype about Extreme Programming, SCRUM, Crystal only to mention a few of the new paradigms. Many software developers are convinced that the new attitude solves most of the problems of the past and is adequate to the challenges of today, especially in the fast world of internet with internet time.

On the other hand there is experience with CMMI that shows that customer satisfaction is growing with maturity level. There are many calls for tender that explicitly ask for ISO 9001 certification or for a certain CMMI level.

Many companies not only are able to show an ISO 9000 certificate but really believe in quality management and successfully manage their business with total quality management (TQM) or are at least on the way to organise their business according to TQM Models such as the model for business excellence of the European Foundation for Quality Management (EFQM). How do this both worlds fit together?

*Siegfried Zopf, Dipl.-Ing., SIEMENS AG Austria, Program and Systems
Engineering (PSE) Gudrunstr.11, A-1100 Vienna, Austria
phone: +43 51707 46303, email: Siegfried.Zopf@siemens.com*

Processes

The idea of writing down how the work should be done by everyone in department or company (I will call it organisation in the rest of the text) is a basic concept of quality management. Whenever you find out how to do it better you change the description and everybody who follows the new description does his job better. That is the idea behind the famous Deming Wheel (Fig.1). It is the kernel of quality management systems. Continual improvement, Kaizen or corporate learning are built on this

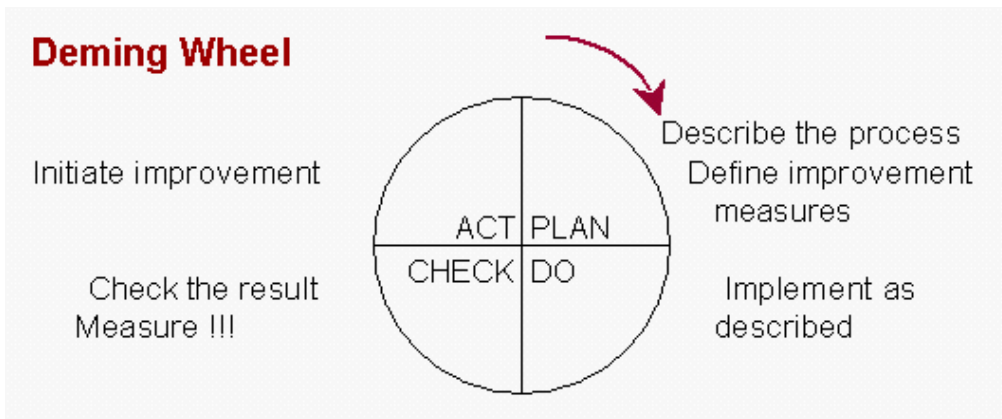


Figure 1: Deming Wheel

This scheme became perfected. Software development processes were worked out in detail for the development on the whole as well as for the sub processes. Those process descriptions tended to become voluminous because the authors mostly had large projects in mind and did not want to ignore any possible necessity in a project. Thus the processes became heavy. Tailoring to the project needs was always recommended but therefore you had to study and understand the whole process to know what to take and what to leave away.

Now enter managers and dull developers who do not take the time to understand the intention of the processes but simply follow the rules literally. The worst case is managers without software development experience who get an incentive for managing the implementation of a process.

Processes applied in this way destroy everything that is necessary for successful projects.

Agile software development

As a reaction to the unproductive and dull development bureaucracy the **Manifesto for Agile Software Development** had to come:

We are uncovering better ways of developing software by doing it and helping others do it.
Through this work we have come to value:

- Individuals & interactions over processes & tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

That is, while there is value in the items on the right,
We value the items on the left more.

www.agilealliance.org

Figure 2: Manifesto for Agile Software Development, Agile Alliance, 2001

That reminds me to the time when we wrote down our System Development Method SEM at Siemens Program- and System Engineering (PSE) at Vienna, Austria in the 1980ies. Nobody thought that processes were a substitute for motivated developers working in teams, or that documentation would be more important than working software. But we saw that even if we usually had a trustful relationship with our customers we needed some kind of contract. When we were writing our method we knew we had to keep our flexibility and we had signed the agile manifesto without hesitation.

The agile manifesto does not oppose methods but the wrong application of methods.

Predictive vs. adaptive processes

It is true that a creative process like software development is not predictive and planable as routine work is. James A. Highsmith goes into details in his book "Adaptive Software Development" /2/

But there is also fundamental difference in understanding the meaning of planning. Especially technicians tend to be very precise. The more decimal places a value has the better. People are not simple objects like stone or steel. People have individual behavior. Within frame conditions it is to a certain extent predictable, but not precisely.

There is a big difference in having no plan at all and in having a plan that can be changed whenever circumstances demand a change.

Concentration on the project

Working in teams that concentrate on the project is emphasized in most literature about agility. The developers should build a team, organize the work themselves

and should not do things that don't contribute to the goal of the project or to the next release of the project.

This may be good for a single project but not for an organization.

Documentation of agile processes

A wide spread opinion or misunderstanding about agility is that you are only agile when you don't care about processes at all.

But on the contrary agile methods are described (documented) and there are key practices and sets of rules for Extreme Programming and other methods.

Values and not bureaucracy are guiding the application of the rules and the collaboration in teams. Professionalism of developers should allow describing the processes without going into much detail.

In the discussions of agile development few is said about the management of an organization in which several agile projects are executed. We find nothing about management responsibility, selecting and training the right people, tracking and control of projects. Shall each project select an agile method or define how a selected method is applied or shall each project define its own process, templates and rules? What about corporate learning and synergy?

Quality Management

Quality Management is a very successful management philosophy. Its key tenets are

- Success on the market through customer satisfaction
- Product improvement through process improvement
- Productivity increase through avoiding errors
- Continual improvement process
- Statistical methods

Because of the importance of quality management (QM) for the success of economies many institutions all over the world built models for QM-Systems and worked on these models and improved and perfected them.

ISO 9000ff Standards on Quality management systems

The ISO 9000ff standards on Quality management systems are the most successful standards of ISO.

In ISO9000:2000 /6/ the quality management principals are stated as shown in Fig.3

ISO 9000 Quality management principals	EFQM Fundamental concepts of excellence
	Results orientation
Customer focus	Customer focus
Leadership	Leadership & constancy of purpose
System approach to management	Management by processes & facts
Process approach	
Factual approach to decision making	
Involvement of people	People development & involvement
Continual improvement	Continuous learning, innovation & improvement
Mutually beneficial supplier relationships	Partnership development
	Public responsibility

Figure 3: Quality management principals and fundamental concepts

The minimum requirements for a quality management system according to ISO9001:2000 /7/ are grouped in chapters as follows

- 4 Quality management systems (general requirements, documentation)
- 5 Management responsibility
- 6 Resource management
- 7 Product realization
- 8 Measurement, analysis and improvement

A QM system shall guaranty that the expected quality will be delivered without waste.

Agility and ISO 9001

The first three chapters: “4 Quality management systems”, “5 Management responsibility” and “6 Resource management” concern the organizational framework or infrastructure in which the development processes are executed but have no direct requirements for the development process but one: The process has to be documented. The extent of the documentation can differ due to the competency of the personnel (4.2.1 Note 2a). Certainly there are implications on developers, such as audits or participation in improvement projects but even this can be done in intervals between projects. Until now nothing hinders agility.

The chapter “7 Product realization” is directly connected with the execution of a project. It has the subchapters:

- 7.1 Planning of product realization
- 7.2 Customer related processes
- 7.3 Design and development
- 7.4 Purchasing
- 7.5 Product and service provision and
- 7.6 Control of monitoring and measuring devices

In ISO 9001 requirements concern the realization of products or services of any kind be it pastries, steel mills or software. Therefore it gives a rather abstract description.

In 2004, ISO/IEC 90003 Software engineering – Guidelines for the application of ISO 9001:2000 to computer Software /8/ was released.

Concerning “7.1 Planning of product realization” it explains in chapter 7.1.1 Software life cycle:

“Processes, activities and tasks should be performed using life cycle models suitable to the nature of a software project, considering size, complexity, safety, risk and integrity. ISO9001:2000 is intended for application irrespective of the life cycle models used and is not intended to indicate a specific life cycle model or process sequence.

Design and development can be an evolutionary process and procedures may therefore need to be changed or updated as the project progresses, after consideration of changes to related activities and tasks.”

From an ISO 9001:2000 point of view I see no principal incompatibility with agile development as long as the procedure is “established, documented, implemented and maintained” (ISO 9001:2000, 4.2.1 Note1) and the organization in which the project is done fulfils the requirements on organizational level. A working quality management system in the organization will strongly foster agile projects.

EFQM Model for Business Excellence

More ambitious models were built to guide companies to business excellence. Competitions for quality systems awards were organized. In Japan the Deming prize, in USA the Malcolm Baldrige award and the European Foundation for Quality Management /10/ defined the EFQM Model for Business Excellence /11/ and advertised the European Quality Awards.

The fundamental concepts of excellence are more or less the quality management principals of ISO 9000 enlarged by public responsibility and the consideration of results (Fig. 3).

As you see in the graphic of the EFQM model (Fig. 4) processes play a central role and contribute 140 points out of 500 points for enablers. The other enablers are about the management of the organization.

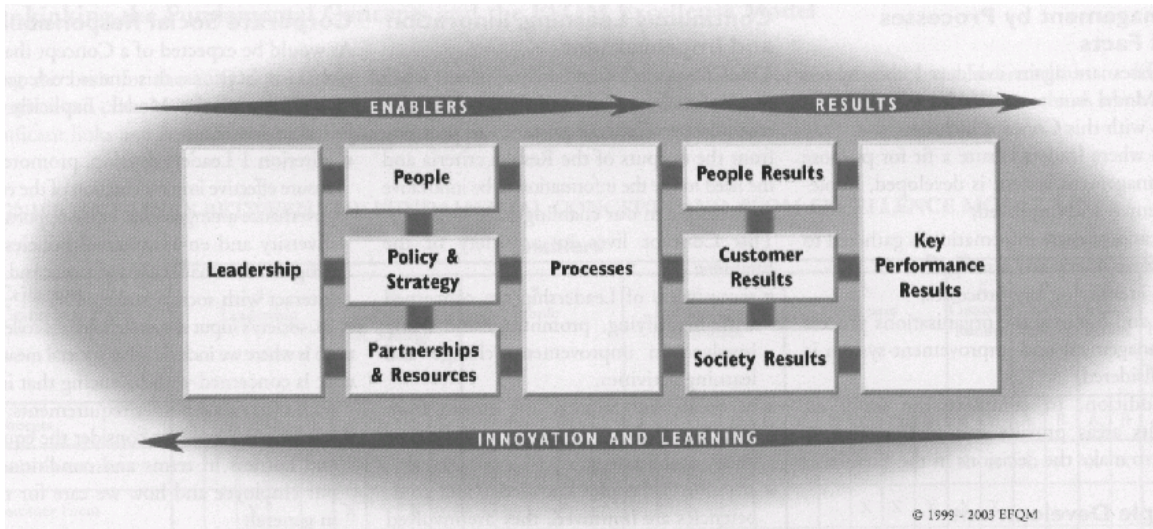


Figure 4: EFQM Model for Business Excellence

As the model for business excellence is a non descriptive model, it does not tell you what to do. You can define a process as you think is appropriate for the business and then apply the quality management philosophy. The organization monitors the execution of the process. It measures and checks if the process delivers the expected results and turns the Deming wheel for continual improvement.

In a (self)assessment you check the elements of the management system with the RADAR logic.

Results, Approach, Deployment, Assessment and Review.

For every result that you have planned you check the approach (the documented procedure is part of the system), the deployment (the execution follows the procedure) and assessment and review (someone looks how it works, measures, learns and improves). You also check, if the results are the consequence of your goals and activities and not only windfall profits.

For being excellent you have to show improvements of your results over several years.

Processes have to be systematically designed, managed and improved. This can certainly be done in agile development.

Capability Maturity Model Integration CMMI

CMMI is different from ISO9001 and EFQM. The Capability Maturity Model Integration is meant to be the application of Total Quality Management to software developing organizations. Other than the EFQM Model it is a prescriptive model. It defines exactly what to do in software development and calls these different activities Process Areas (Fig.5). For every Process Area it

gives a lot of information how to do it. For each Process Area specific goals and generic goals are defined.

<p>Level2 Requirements Management Project Planning Project Monitoring and Control Supplier Agreement Management Measurement and Analysis Process and Product Quality Assurance Configuration Management</p>	<p>Level3 Requirement Development Technical Solution Product Integration Verification Validation Organizational Process Focus Organizational Process Definition Organizational Training Integrated Project Management Risk Management Decision Analysis and Resolution</p>
<p>Level4 Quantitative Process Management Software Quality Management</p>	<p>Level5 Organizational Innovation and Deployment Causal Analysis & Resolution</p>

Figure 5: CMMI Process Areas

The Generic Goals reflect the organizational embedding of the processes. The Common Features

- Commitment to perform
- Ability to perform
- Directing implementation
- Verifying implementation

remind strongly to the ISO 9001 requirement that procedures are established, documented, implemented and maintained”.

CMMI is a waterproof model for software development with the aim to foresee every possibility that may occur in a project. It is a collection of experience and best practise and complies with the highest requirements even on level 2 and 3. The CMMI is also used as one of the reference models in Spice Assessments /9/.

Measurement at Siemens PSE /3/ showed that higher CMM-Levels improve customer satisfaction (Fig.5).

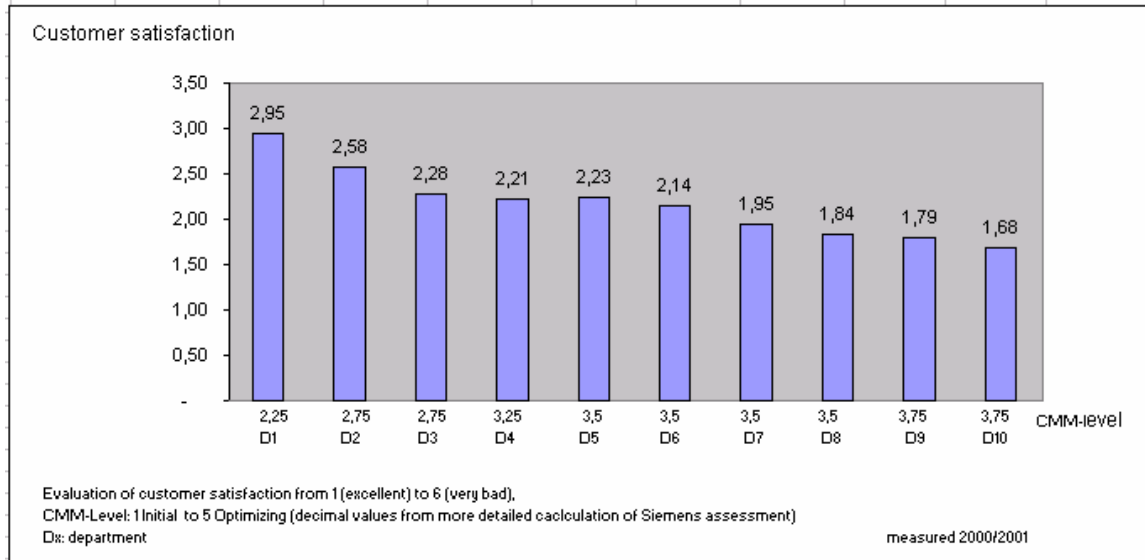


Figure 5: Customer satisfaction vs. CMM-Level

CMMI is an excellent coherent system of process areas that are interwoven perfectly. But its complexity brings people to follow letter by letter instead of agile, based on values and goals.

CMMI suggests Tailoring

In Chapter 6 of the CMMI Version 1.1 Staged Representation /13/ the **model tailoring** of the CMMI model to the need of an organization is encouraged.

“Tailoring a CMMI model is a process whereby only a subset of a model is used to suit the needs of a specific domain of application.”

“The models were written for use by all types of organizations; however for small types of organizations a CMMI model must be interpreted.”

“Often the project plan also contains plans for other supporting functions, such as quality assurance and configuration management. A four-person project might expect to develop a project plan that is only a few pages long.”

CMMI is not necessarily as heavy as its is often applied.

Extrem Programming and CMMI

“Blending Agile Development Methods with CMMI” /1/ is the title of a report written by Glen Alleman in which he discusses how Extreme Programming practices and agile values fit into a CMMI environment, enabling the organization to become more agile while still meeting its compliance standards. He shows a table mapping the CMMI process areas to XP practices. Not surprisingly there is a match or at least a moderate match with level 2 process areas and with level 3 process areas concerning development projects but practically no match with process areas concerning organizational matters on level 3 or quantitative management on level 4.

In our organization we developed the iterative incremental process e-SEM based on our standard process stdSEM. It supports agile development wherever agility is appropriate.

The question is: how much fantasy is allowed when it comes to fulfil the specific goals of a process area.

For example: Can Story Cards and “Customer on site” satisfy the level 2 Process Area “Requirements Management”, Specific Practice SP1.1 “Develop an understanding with the requirements providers on the meaning of the requirements”?

Probably yes when there is only one stake holder. When there are more write down a list of the stakeholders and there their goals and define a simple change request process. In my opinion this does not contradict agility. The Typical Work Product “Criteria for evaluation and acceptance of requirements” may be part of the compulsory training of the developers.

It certainly is not so easy with every element of the model even on level 2. A workshop with people from SEI and the industry /15/ identified one “rough edge” on level 2: “Objectively monitor adherence to process and QA products/services”. At the higher levels organizational aspects and quantitative management are not met by agile methods.

EFQM for Software Intensive Organizations

The European Software Institute ESI and the EFQM are cooperating in the project “Software TQM” to build an “EFQM Excellence model for Software Intensive Organizations”. Work is based on CMMI. But in my opinion quality management in software development does not necessarily mean CMMI.

Conclusio

From a quality management point of view an organisation has to have a management system that works according to the philosophy or principals or fundamental concepts (or how ever you want to call it) of quality management. To be able to apply this management philosophy to software development projects, the projects have to follow a documented process. Data have to be collected, analysed and used for process improvement. The organisation with a quality management system with its focus on goals, people, training, resources and systematic management is the perfect environment for executing software development projects (agile or traditional).

The additional work load of process definition as well as data collection and analysis can be done by the organization outside the project. The project team can stay small and concentrate on the project with the exception of participating in process improvement. But that is not outside the realm of agility. Even in XP there is a rule called “Fix XP when it breaks” for process improvement.

On the other hand, sometimes quality management is done in a heavy handed bureaucratic manner. Perhaps we should call for agile quality management.

References

- /1/ Alleman, Glen, Blending Agile Development Methods with CMMI, Cutters IT Journal June 2004
- /2/ Highsmith, James A., Adaptive Software Development: a collaborative approach to managing complex systems, 1999
- /3/ Zopf, Siegfried, CMM-Level vs. Customer Satisfaction, EOQ-Conference on Software Quality, 1999
- /4/ <http://www.agilealliance.org>
- /5/ <http://www.extremeprogramming.org/>
- /6/ ISO9000:2000 Quality management systems – Fundamentals and vocabulary
- /7/ ISO 9001:2000 Quality management systems – Requirements
- /8/ ISO/IEC 90003 Software engineering – Guidelines for the application of ISO 9001:2000 to computer software
- /9/ ISO/IEC 15504 Information technology – Process assessment
- /10/ <http://www.efqm.org>
- /11/ EFQM Excellence Model, EFQM Brussels, 2003
- /12/ <http://www.sei.cmu.edu/>
- /13/ Capability Maturity Model Integration Version 1.1 Staged Representation, CMU/SEI-2002-TR-012, 2002
- /14/ <http://www.swtqm.net>
- /15/ Agile Methods and CMMI – Annual Research Review and Executive Workshop, 2002
<http://sunset.usc.edu/events/2002/arr/presentations/AGILE%20Methods%20and%20CMMI.ppt>