An Approach for Using CMMI in Agile Software Development Assessments: Experiences from Three Case Studies

Minna Pikkarainen and Annukka Mäntyniemi
VTT Technical Research Centre of Finland, P.O. Box 1100, FIN-90571 Oulu, Finland
Minna.Pikkarainen@vtt.fi, Annukka.Mantyniemi@vtt.fi

Abstract

Software development organizations are increasingly interested in the possibility of adopting agile development methods. Organizations that have been employing the Capability Maturity Model (CMM/CMMI) for making improvements are now changing their software development processes towards agility. By deploying agile methods, these organizations are making an investment the success of which needs to be proven. However, CMMI does not always support interpretations in an agile context. Consequently, assessments should be implemented in a manner that takes the agile context into account, while still producing useful results. This paper proposes an approach for agile software development assessment using CMMI and describes how this approach was used for software process improvement purposes in organizations that had either been planning to use or were using agile software development methods.

1. Introduction

Over the past years, the Capability Maturity Model® (CMM) and Capability Maturity Model® Integration (CMMI℠) [1] have been broadly used for assessing organizational maturity and process capability throughout the world [2]. Many organizations are now used to regular CMMI assessments and appraisals. They have confidence in CMMI because of its extensive descriptions of how the various good practices fit together [3].

Lately, a new approach to software development has aroused wide interest among software development organizations. Agile software development methods, practices and techniques promise to increase customer satisfaction [4, 5], to produce high quality software and to accelerate development times [6].

Companies that have made a huge effort on improving their processes based on CMMI have now realized that agile approaches can provide improvements as well. On the other hand, agile practices have been criticized for a lack of discipline and argued of being suitable only for some specific types of projects (e.g. small teams and applications) [4, 7].

As organizations may be dithering between their old plan-driven and new agile methods, using CMMI as a framework for assessing the current state of development could help building trust in agile methods or provide safer ground for starting agile-based improvements. After all, CMMI represents the traditional and familiar way of getting things done.

This paper proposes an approach for assessing agile software development using CMMI, and presents how assessments were performed in a total of seven projects carried out in three organizations using or planning to use agile software development methods. While the initial purpose of CMMI is to provide objective assessment of organizational maturity or process capability, the goal in these cases was to use CMMI as an assessment framework in software process improvement and thus, case organizations' processes were not rated against CMMI. The approach was developed concurrently with its actual performance and it evolved from case to case.

The paper is composed as follows: Section 2 presents the background and related research of agile software development and CMMI; Section 3 introduces the research context; Section 4 proposes the approach; Section 5 focuses on describing the case studies; and in Section 6, the results of these cases are discussed. The last section concludes the paper with the final remarks.

2. Background

This section describes the background of agile software development, CMMI, and the relationship between these two.

2.1 Agile software development

Several agile software development methods have been suggested in the literature, e.g. Extreme Programming (XP) [10], Scrum [11], Crystal methodologies [12], and Mobile-D™ [8, 13] (agile.vtt.fi/mobile-d). All of these methods employ agile principles, such as iterative cycles, early delivery of working software and simplicity as defined in Agile Manifesto [14].

Plan-driven software development (e.g. processes compliant with CMMI or ISO 15504) and agile software development methods are often seen as opposites to each other [15, 16]. Plan-driven approaches suppose that software development is a repeatable and predictable process. Agile developers do not believe that these assumptions are valid for projects involving any degree of exploration [17].

Agile methods define how the work should be carried out under agile values and principles (Agile Manifesto [14]) to answer the challenges of rapid development and changing requirements. Agile practices can be described as activities within the agile methods (e.g. the Planning Game, an iteration planning practice in XP). Agility, as characterized by Highsmith [5], is "the ability of to both create and respond to change in order to profit in a turbulent business environment".

2.2 CMMI

Capability Maturity Model® Integration (CMMI^{SM}) [1] is a widely known appraisal approach for determining organizational maturity and process capability. CMMI has four disciplines to choose from: systems engineering (SE), software engineering (SW), integrated product and process development (IPPD) and supplier sourcing (SS). The model itself has two representations: staged and continuous. The staged representation focuses on a set of process areas, which are organized by maturity levels (1-5), while in continuous representation each process area is rated in terms of capability level (0-5).

CMMI-SE/SW describes 25 process areas. The process areas have specific and generic goals, the fulfilment of which is appraised through practices. The practices are further categorized as specific and generic. Generic goals and practices apply to multiple process areas, whereas specific goals and practices apply to individual process areas. The specific goals and practices of process areas describe what kind of activities need to be carried out. Generic goals and practices are aimed at finding out how well the activities are performed.

Through appraising organization practices, sub practices and work products against those defined in CMMI, the fulfilment of related goals can be assessed. However, the main purpose of the appraisal is to find out if the goals are achieved or not, instead of finding out if all the defined items exist as such. Thus, these items can be considered as tools for the appraisers when evaluating the fulfilment of the goals.

2.3 Agile software development assessment

At the moment, no methods exist for assessing agile software development. The approach of Boehm and Turner [4] provides a way for assessing the agile home ground of a software development project. However, this model maintains a strict focus on assessing the agile and plan-driven risks rather than finding the weaknesses and strengths of the used practices. Thereby it may not provide specifics on the state of the processes and may thus not give enough data on what needs to be improved.

Several studies have been conducted on CMM and CMMI as used for assessment or as a basis for improvements in organizations or projects employing agile practices [19-23]. In these studies, the agile practices used in organizations or projects were, in many process areas, found to fulfill the CMMI Level II and Level III goals. Anderson [19] even states that it is possible to develop a truly agile full life cycle process which meets the requirements of all the 5 levels in the CMMI model. Boehm and Turner [18], in turn, state that the Level 5 concept of constantly adapting to improve performance is in line with agile methods, while also maintaining that most agile methods do not support all elements for lower-level certification.

CMM or CMMI and agile methods have also been compared in several studies, and mappings or comparisons between agile and CMMI practices have been proposed [15, 24-26]. For example, Paulk [26] suggests that XP's use of stories, on-site customer and continuous integration fulfill the SW-CMM requirement management goals. On the other hand, Turner and Jain [15] found in their study that several of the CMMI components and agile methods were in conflict, most of them being those addressing organizational processes. Yet, many of them were also found to be supportive or neutral to each other, especially those focusing on project management.

There is also criticism towards these kinds of comparisons. These two models have not been found to be comparable as entities because the agile methods are development process descriptions while CMMI is a reference model for appraisals [4]. Thus, as it has also been suggested, for example, by Paulk [26] and Kähkönen and Abrahamsson [23], CMMI could be rather used as a tool when building up methods that combine agile and traditional elements. According to Paulk [26], "taken together, the two methods can create synergy, particularly in conjunction with other good engineering and management practices".

3. Research Design

This section describes the research goals and approach of this study. The case study contexts are also shortly introduced.

3.1 Research goals and approach

The goal of this research was to study how Capability Maturity Model® Integration (CMMI) [1] could be used in assessing agile software development or in a situation in which the organization is planning to change its processes towards agility. The research was designed to be conducted as a series of case studies.

The approach was developed concurrently with its actual performance and it was adapted and enhanced from case to case. The baseline of the approach was designed to be loose in the beginning, with no strict guidelines on how to proceed, so that it could be adjusted to the specific agile development context. The limitation of the taken approach was that the actual costs of the assessments could not be evaluated, because the effort made to develop the approach and the effort made for the assessments were found to be inseparable.

After the assessments had been conducted, a survey concerning the successes and weaknesses of the assessments was sent to the representatives of the case organizations in order to evaluate the approach. In addition, the actions taken after the assessments can be considered as further evaluation criteria for the approach.

3.2 Case study contexts

A total of three organizations and seven different projects were selected for case studies. All the case organizations were operating globally, two in the embedded software development branch, and one in the field of application software (mainly PC but also embedded SW), while the company size varied from medium to large. The general research purpose of the case studies was to collect experiences on using CMMI in agile software development assessments.

All the assessed organizations were familiar with CMMI and its utilization as a part of software process improvement. All the organizations also showed a strong interest towards improving their processes with agile methods.

The data collection methods used in the cases are described in Table 1.

Table 1. Data collection in the case studies

Case	Data collection
Case 1	5 interviews: a program manager, 3 project managers and a software engineer
Case 2	5 interviews, a project manager, three software engineers and a customer of the project; workshop
Case 3	Survey: comparable data from all three projects Scrum1: 2 workshops; 5 interviews: project manager, quality engineer and 3 software engineers; observation of a post-mortem meeting Scrum2: workshop Mobile-D: documentation study of the results from Post-Iteration Workshops; two interviews: PIW facilitator, project manager; participatory observation of a post-mortem meeting

4. An approach for assessing agile software development with CMMI

There is no reason why CMMI could not be used as such in assessing agile software development processes. Agile software development – as a part of other organizational practices - can be seen as one of the diverse environments [26, 27] for a CMMI appraisal. In this context SCAMPI [28], for example, could be used as an appraisal method and a trained team of professionals could perform the appraisal.

However, CMMI compliant processes and agile practices are often seen as opposites [15] and it has been found that CMMI does not always support interpretations in an agile context [23]. Thus, an approach that takes agile context into account would be more useful, particularly if the purpose is not to rate processes, but rather to build processes based on CMMI and agile methods or to verify if current agile processes are effective. In an agile environment, using CMMI assessment items can be a demanding task as the practices and work products are quite different from traditional plan-driven software development. In addition, the processes can be combinations of plan-driven and agile practices, which may confuse the situation even more. Thus, the assessment team should be aware of the nature of all the factors affecting the development, such as agile practices (e.g. Planning game in XP) and plan-driven practices (e.g. Requirements elicitation). Help can be provided for this issue by drawing up mappings describing the connections between the used CMMI and agile practices.

The assessment process illustrated in Figure 1.

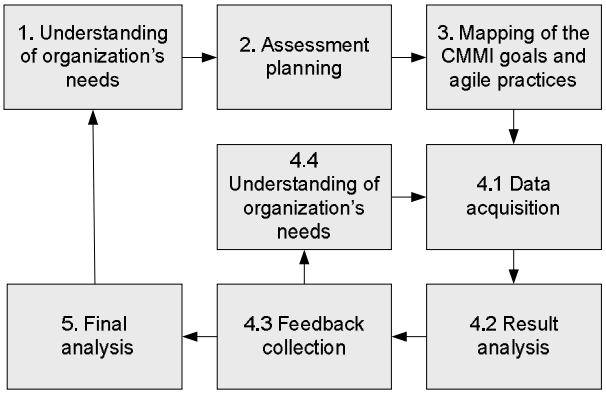

Figure 1. Assessment Process

The first step is to understand what the target organization's goals for conducting an assessment are. In the traditional assessments, the assessment goals are typically defined at the beginning of the assessment process. In the proposed approach, however, the idea is to bring the assessment team and the customer closer to each other and to observe the changes of the goals and data acquisition needs also during the assessment process.

In the second step, the assessment execution is planned together with representatives of the target organization. These representatives are expected to participate in the assessment process (e.g. as information providers, analysts and result interpreters).

In the third phase, a mapping between the selected assessment framework (CMMI in this case) and agile practices is carried out. The fourth step includes an iterative execution of the assessment, with phases of data acquisition, result analysis, feedback collection and understanding of the organization's needs. This step is iterated until enough data has been acquired. The decision of enough is made during the assessment process together with the organization's representatives. Data can be acquired, e.g. through interviews, workshops, observation, surveys, and document analysis.

The fifth step includes final analysis, along with a presentation and packaging of the results. The key purpose of this step is to address the improvement targets and their solution alternatives. The whole cycle can be iterated from the beginning.

5. Case studies

The assessments were conducted by an assessor team, comprising two to three persons, which was familiar with CMMI, had several years of experience on software process improvement and assessments, but which had also gained knowledge on agile software development methods and their use in industry.

In the following sections, the case studies are presented.

5.1 CASE 1

In the first case, the assessment process was gone through once. The assessment focus was defined together with a representative of the organization management and a quality co-ordinator, who also participated in the analysis phase. The goal of the assessment was to find the most suitable agile practices to be incorporated into the organization's processes. The assessment was limited to the CMMI Project Management process areas and the Requirements Management process area (under Engineering).

All the three assessed projects were small (2 - 5 members), representing the general project size in the company. It was found that, at the time of the assessment, all the three projects were following few agile principles or practices (e.g. Rapid 7 [9], which is a teamwork documentation practice), while no agile methods were used to their full potential.

A mapping between the CMMI specific goals and possible suitable agile practices was done. For example, the CMMI Goal of Manage requirements was mapped against the following agile practices: Product Backlog (dynamic list of requirements [11]); Planning Game/Sprint Planning Meeting (goals and requirements for the next iteration defined by the team and the stakeholders [10, 11]); Stories (definition of customer visible functionality + task estimation [29]); Sprint backlog (goals, tasks and estimates for a Sprint, i.e. iteration [11]); Daily scrums (daily status meetings [11]); Information radiators (project information on the wall [12]); Sprint & release reviews (review after an iteration/release in which the development team and the stakeholders participate [11]); On-site customer (a customer available to answer questions of development team [10]) participating in requirements definition and validation activities; and Self-organizing teams (authorized to make decisions [11]) in establishing commitment to the requirements.

The first one of the assessed projects was a new product development project without a customer interface, as no potential customers were yet involved in the project. The project was quite at the beginning of development, and probably due to the early phase of development, this project seemed not to have problems in requirements or project management.

The two latter projects showed situations that were totally opposite to the first one. These projects were concerned with the further development and tailoring of

an existing product based on customers' needs. In these projects, especially the management of requirement changes was found problematic. The project had initially well-established plans and requirements specifications, but the planning was done based on the needs of a single customer, and many new customers came along during the development. These new customers had extra needs, which proved difficult to prioritize. The new requirements turned out to cause problems to scheduling and task prioritization and especially to the task of release planning. As a consequence, project plans, requirements specifications and release plans could not be kept up to date.

In an analysis made together with the customer it was concluded that changes to the development process were needed when several customers could be expected to be involved in a project during the development. In a more stable situation, project and requirements management could also be well handled using the existing plan-driven approaches and longer iterations (e.g. two months).

In more turbulent projects, the improvements were planned to be achieved with a Mobile-D™ based iterative development process with shorter cycles (3 - 4 weeks). Each iteration would begin with a planning phase (requirements, tasks, estimates for an iteration) and end with a release and demonstration of working software followed by a post-iteration workshop (PIW) [30].

During the analysis, the improvement targets and agile-based solution proposals were identified and categorized using mapping. A representative of the case organization participated in this analysis work.

The results were presented to and discussed with the interviewees and management in a workshop. From among the proposed agile practices the case organization selected those that they thought would solve the identified problems, and improvements were incorporated into the organization's software development process model. As a result, the organization has now two separated process models: one for stable new product development and another for turbulent product tailoring. The agile-based improvements were deployed in the selected pilot projects. The first experiences collected so far indicate that the deployment has been quite complicated, mainly because of change resistance.

5.2 CASE 2

In the second case study, the assessment execution process was gone through twice. The goal, which was defined together with the project manager, was to evaluate the strengths and weaknesses of an agile software development project. The project used an integration of the Extreme Programming (XP) and Scrum methods. The focus was on CMMI level 2 process areas and level 3 engineering process areas, excluding Subcontractor Management. Mapping was done between the used agile practices and CMMI specific goals. Some examples of this mapping are described in Table 2.

Table 2. CMMI goals and used agile practices

CMMI goal	Agile practices
Manage requirements	Stories; Product Backlog; Planning Games; Information Radiator; Daily meetings; On-site customer; Self-organizing teams
Establish estimates	Planning Games; Tasks and effort estimations for one- to two-week iterations on information radiator
Develop a Project Plan	Planning Games; Tasks on information radiator; Product backlog
Obtain commitment to the Plan	Planning Games; Self-organizing teams; On-site customer; Reflection workshops

The purpose of the assessed project was to develop an internal reporting tool to be used in the product development of the case organization. The project team had four developers, a project manager and an internal customer representative.

Through an interview analysis, the strengths and weaknesses of the used agile practices were initially identified. During the analysis, it was noticed that not all the questions could be answered. Therefore, it was decided to run the second assessment execution cycle and to collect additional data by using a workshop (comparable to a group interview). For this, a template for guiding the workshop was prepared based on mapping. The assessment results were reanalysed based on the workshop results and discussed with the project management.

In the project, requirements were first captured, together with customer representatives and a project manager, as stories, which formed a product backlog. After this, the initial requirements were further elaborated together with the customer, project management and development team in a planning game. The developers found that the requirements in the product backlog were not understandable or clear enough in the first planning games, which made requirements analysis a laborious task. However, through iterations developers learned more about the needs of the customer and the product under development. As a consequence, the requirements analysis became easier.

Requirements changes were discussed during iterations between the customer, other stakeholders and the project manager, and the product backlog was updated correspondingly. Requirement changes were

analyzed at the beginning of the iterations in planning games, in which also the development team participated. Although lots of new requirements and requirements changes appeared, the developers felt that they were working with stable requirements, because they were spared from continuously ongoing requirements negotiations during Sprints (i.e. iterations). The stakeholders validated the work products between the iterations, and thus the inconsistencies between the requirements, plans and project work were continuously followed.

Project planning was done iteratively during the planning games by defining the scope, goals, tasks and estimates for the next iteration. Overall budget, risks and scope were defined and maintained in separate steering group meetings. Tasks were published on an information radiator (i.e. information on the wall). The developers thought that the freedom to define tasks, establish estimates and make technical decisions by themselves (self-organizing teams) motivated them and reinforced their commitment to the project and to the requirements.

At the beginning, problems appeared due to too high level task definitions and effort estimation difficulties. In reflection workshops (team discussing periodically about the strengths, problems and improvements of the project [12]) held after the iterations, this problem was identified and task sizes were decided to be downsized. Consequently, the estimation accuracy improved.

During and after the assessment, the case organization had several agile projects and improvement initiatives going on. The assessment results were used for learning and selecting practices for the following agile projects.

5.3 CASE 3

The goal of the third case was to identify the most efficient agile practices from three different agile projects and to compare the practices used in two Scrum and one Mobile-D™ projects. The goal was defined together with the organization management. The assessment execution process was gone through twice. The CMMI focus was the same as in the second case and similar types of mappings between CMMI goals and agile practices were done. Examples of the mapping are given in Table 3.

Table 3. CMMI goals and agile practices

CMMI goal	Scrum Practices	Mobile-D practices
Manage requirements	Product and Sprint backlogs; Sprint planning, Sprint reviews; Self-organizing teams	Product backlogs; Stories; Planning days; Release days
Establish estimates	Sprint Planning; Tasks and effort estimations for 1- to 4-week releases	Planning Days; Task cards with effort estimates on information radiator
Develop a Project Plan	Sprint Planning; Product backlog; Sprint backlog	Planning days; Information radiator; Product backlog
Obtain commitment to the Plan	Sprint planning, Sprint Review; Self-organizing teams;	Planning days, Task cards on information radiator; Self-organizing teams; Release day

The Scrum1 project and the Mobile-D project developed products for mass markets. The products were considered critical in terms of organization's competitiveness. The Scrum2 project was an internal project the purpose of which was to research and implement new solutions to support the organization's core business activities. All the projects had four developers, a project manager, and a separate quality engineering team taking care of integration testing activities. At first, interviews, a documentation study, and workshops were selected as main data acquisition methods, but later, in the second assessment execution cycle also observation, survey and additional interviews were decided to be used.

All the projects followed an incremental and iterative life cycle model, with varying iteration lengths (from 1 to 4 weeks). The Mobile-D and Scrum1 projects had several external customers, the needs of which were collected by the product manager. The Scrum2 project had internal stakeholders.

In all the projects, the requirements were defined in product backlogs together with a product manager (i.e. product owner, responsible for the product [11]), (technical architect in Mobile-D), and a project manager. All the projects had iteration planning meetings (planning day/sprint planning), in which the requirements were defined at a more detailed level either as stories or in a sprint backlog.

One problem in the Scrum1 project was that the requirements were not understood all the time by the development team, because the sprint planning meetings were relatively short in view of the huge amount of requirements involved, especially at the end of the project. Due to the number of customers, a mass of new requirements and requirements changes were appearing all the time during the project. The team thought that new features were being too easily included in the product backlog, with too little consideration given to the technical constraints. The backlog was also getting so large that it became difficult to manage. Consequently, not all the feature sets could be completed during the project and it was difficult to keep the overall focus clear.

The overall budget, risks and scope of each project were defined in the initial project planning stage by management and updated during development. In all the projects, the teams specified the tasks and effort estimations for the iterations and recorded them electronically at the beginning of the iterations (sprint planning/planning day); The Mobile-D project also defined task cards on an information radiator. Due to iterative planning and daily status meetings the projects were able to keep their plans up-to-date. In sprint reviews/ on release days, inconsistencies between the requirements, plans and project work were checked together with the development team and stakeholders.

The assessment results were presented to the management including a comparison of the used agile practices in the selected CMMI process areas, along with an analysis of the strengths and weaknesses of the agile software development practices, and an analysis of the improvement targets and proposals at the organizational level. As a result of the final analysis, it was concluded that the piloted agile process models required improvements. For example, it was found that agile methods that are thought to especially fit turbulent environments seemed not to provide an unambiguous solution for requirements change management problems in Scrum1 project. This is in line with what Boehm and Turner [18] have noticed: in view of the fact that agile requirements tend to be primarily functional and reasonably informal, it might be necessary to strengthen the agile requirements approach. Thus, a more plan-driven approach along with a more in-depth requirements analysis were identified as one solution to this problem, which would make it unnecessary to add all customers' needs into the product backlog.

The organization used the assessment results to define an agile-based development model, which was used alongside of their plan-driven software development processes. Later on, agile practices have also been taken into use even in larger and more complex projects and the tendency is to utilize the defined practices throughout the organization.

6. Discussion

The discussion of conclusions on using CMMI in assessing agile software development bases on qualitative data involving subjective opinions of an assessment team and on opinions that were collected through a short survey of the two case organizations. Yet, the actions taken after the assessments can be considered as success criteria of the taken approach even though the implications of the process changes could not be followed during the time scale of this study.

In all the cases, a mapping between CMMI goals and agile practices was done. In the first case, the mapping was theoretical because the organization was mainly employing plan-driven processes. In the other two cases, CMMI goals were mapped against the used agile practices. In all the cases, the mappings were found to facilitate the categorization of agile-based improvement goals and proposals because these could be presented in terms of CMMI goals and corresponding agile practices. CMMI also made agile practices more understandable for the projects not using agile methods (all the three projects in the first case), and managers were able to understand the connection between the organization's agile practices and the CMMI compliant processes used in other projects or in their earlier projects. However, it should not be assumed that following the CMMI-corresponding agile practices would inevitably lead to success. For example in the Case 3, the Scrum1 project followed several agile practices suggested to correspond to CMMI requirements management specific goal, but the project had problems in managing the requirements - not due to the fact that the practices were not followed, but mainly because of following them.

The mappings in these cases were done at the CMMI specific goal level. The mapping could be further elaborated so that a CMMI sub practice would correspond to a specific agile practice. Because in these cases the purpose was not to give any official ratings to specific process areas, but rather to find weaknesses and strengths in current processes, this mapping was only done when considered necessary and, thus, not systematically in all the cases.

Problems with CMMI in the agile environment are likely to arise in situations requiring written evidence on the used practices. This is due to one of the agile values, “Working software over comprehensive documentation”, meaning that the level of documentation is to be kept as low as possible and done late during the process. It is possible that many of the CMMI practices are in fact performed, but without any documentation through communication between the relevant stakeholders. It may thus seem that, for example, the design phase is neglected even though it happens all the time during the development. An appraisal team should, therefore, be aware of the nature of agile development – or, quoting the words of Boehm and Turner [18]: "It's possible that enlightened appraisers can find ways to include agile methods as alternative practices in many instances".

Additionally, as agile methods emphasize adaptable working practices, iterative development and late documentation, a "snap-shot" image of the project may not correspond to the situation in later iterations or may not be in line with the general working practices of the organization. This can be seen clearly in Case 2, in

which, for example, estimation improved during the project, while the project focus became fuzzier due to changed practices. Thus, as development practices may vary in different projects and during a single project, an overall picture of the capabilities of the different process areas or of organizational maturity can be difficult to form. On the other hand, this adaptability is in line with the CMMI level 5 concept of constantly adapting to improve performance [18], denoting that an experienced appraisal team should be aware of how adaptability and change should be taken into account in any appraisal, not only in the evaluation focusing on agile development.

In these three cases, the assessment team felt that workshops and observation were productive ways to collecting assessment data from the agile projects, although neither of these methods would fulfil a strict confidence principle of CMMI [31]. The workshops provided data that was already prioritized and categorised. Additionally, project members appreciated the workshops as a knowledge sharing event between different projects and project parties. The observation provided a good general view of the projects (i.e. successes and problems) and of the plans for the following projects (i.e. what is planned to be changed/improved).

The case organizations agreed but did not strongly agree, on a five-step scale, that the objectives for the assessment had been well achieved. The effort made in the assessment was as great as expected, reporting was considered sufficient but not too arduous, and the expertise of the assessment team was appreciated (all these points were strongly agreed or agreed with).

In one of the cases, the assessment results were used for selecting practices for the following projects. In two cases, the assessment had also organizational level implications, as the organizations started to define and deploy new agile-based software development processes based on the assessment results. Thus, the taken approach seemed to provide useful information for starting agile-based improvements. It was also found that there was no need to change the approach frame during the case projects because the approach could be adjusted to every given situation.

To conclude, based on the case experiences it can be stated that CMMI is a valuable tool when assessing the strengths and weaknesses of an agile software development project due to the fact that CMMI allows agile development to be viewed from a perspective of generally known practices and that it can thus be assured that all the focal viewpoints of software development are taken into account. While CMMI does not always support interpretations in an agile context, as also found by Kähkönen and Abrahamsson [23], help can be provided by drawing up mappings describing the connections between CMMI and agile practices.

7. Conclusions and further work

This paper introduced an approach for assessing agile software development using CMMI. The assessments were performed in a total of seven projects in three case organizations using or planning to use agile methods. The purposes of the assessments were to identify strengths and weaknesses of the existing processes and to define improvement goals and agile practices that could be used in improvements. The research goal of the cases was to collect experiences on using CMMI in assessing agile software development.

Mappings between CMMI specific goals and agile practices were used as a central tool in the assessments. In the studied cases, CMMI was found to be useful in assuring that all focal software development viewpoints were taken into account in the assessment, whereas mappings were found to clarify the connections between the agile and plan-driven processes and thus to ease both analysis and understanding of the assessment results.

The assessment results have subsequently been used in improvement initiatives in which the case organizations' processes have been changed towards agility. Unfortunately, the effort that was made on assessments could not be extracted from the effort made to develop the process, and further, the implications of process changes in all the three cases could not be followed within the time scale of this study. Thus, due to the lack of cost and benefit figures, the return-of-investment could not be evaluated.

It is concluded that the proposed approach for assessing agile software development using CMMI produces useful results for starting agile-based improvement efforts. It can be used, for example, as in these three case studies, for selecting suitable agile practices to be incorporated into the organization's existing processes and to be deployed in forthcoming projects.

As a typical limitation of case studies, the results of this study are context-specific. Furthermore, the number of cases is too low for making generalizations. In addition, due to the lack of a reference model for agile practices (e.g. a standard) the mappings were done based on the assessment team's current knowledge gained from a literature study and personal experience. Thus, the mappings presented are also subjective and context-specific.

In the future, the study of assessment in the agile context will continue by addressing the views of other assessment reference models, such as ISO 15504. The purpose of the future studies is to improve and further evaluate the proposed assessment approach.

Acknowledgements: The research was conducted under the Agile ITEA project funded by the National Technology Agency of Finland (TEKES). We would like to express our gratitude to the Agile team of VTT, to the industrial partners that have participated in this research, and to the three anonymous reviewers of this paper.

References

[1] CMMI, "CMMI[SM] for Systems Engineering/Software Engineering/Integrated Product and Process Development/Supplier Sourcing, Version 1.1, Staged Representation (CMMI-SE/SW/IPPD/SS, V1.1, Staged.", 2002.

[2] D. Zubrow, "Current Trends in the Adoption of the CMMI Product Suite," *In Proceedings of the 27th Annual International Computer Software and Applications Conference*, pp. 126-129, 2003.

[3] M. Buch and D. Dunaway, *CMMI Assessments, Motivating Positive Change*, Addison-Wesley, 2005.

[4] B. Boehm and R. Turner, *Balancing Agility and Discipline -A Guide for the Perplexed*, Addison- Wesley, 2003.

[5] J. Highsmith, *Agile Project Management, Creating innovative products*, Addison-Wesley, 2004.

[6] D. J. Anderson, *Agile Management for Software Engineering, Applying the Theory of Constraints for Business Results*, Prentice Hall, 2003.

[7] P. E. McMahon, "Extending Agile Methods: A Distributed Project and Organizational Improvement Perspective," *CrossTalk, The Journal of Defense Software Engineering*, vol. 18, issue 5, pp. 16-19, 2005.

[8] T. Ihme and P. Abrahamsson, The Use of Architectural Patterns in the Agile Software Development of Mobile Applications, *International Journal of Agile Manufacturing*, vol. 8, issue 2, 97-112, 2005.

[9] R. Kylmäkoski ,"Efficient Authoring of Software Documentation using RaPiD7", *In proceedings of the 25th International Conference on Software Engineering*, pp. 255-261, 2003.

[10] K. Beck, *Extreme Programming Explained: Embrace Change*, Addison-Wesley, 2000.

[11] K. Schwaber and M. Beedle, *Agile Software Development With Scrum*, Prentice Hall, 2002.

[12] A. Cockburn, *Crystal Clear, A Human-Powered Methodology for Small Teams*, Addison-Wesley, 2004.

[13] P. Abrahamsson, A. Hanhineva, H. Hulkko, T. Ihme, J. Jäälinoja, M. Korkala, J. Koskela, P. Kyllönen, and O. Salo, "Mobile-D: An Agile Approach for Mobile Application Development," *In proceeding of the 19th Annual ACM Conference on Object-Oriented Programming, Systems, Languages, and Applications*, pp. 174-175, 2004.

[14] AgileManifesto, *Manifesto for Agile Software Development http://agilemanifesto.org/* (1.3.2006)

[15] R. Turner and A. Jain, "Agile Meets CMMI: Culture Clash or Common Cause," *In proceedings of the Second XP Universe and First Agile Universe Conference on Extreme Programming and Agile Methods - XP/Agile Universe*, pp. 153-165, 2002.

[16] M. Lycett, R. Macredie, C. Patel, and R. J. Paulk, "Migrating Agile Methods to Standardized Development Practice," *Computer*, vol. 36, issue 6, pp. 79-85, 2003.

[17] J. Highsmith, "What Is Agile Software Development?," *CrossTalk, The Journal of Defense Software Engineering*, vol. 15, issue 10, pp. 4-9, 2002.

[18] B. Boehm and R. Turner. "Management Challenges to Implementing Agile Processes in Traditional Development Organizations," *Software*, vol. 22, issue 5, 30-39, 2005.

[19] D. J. Anderson, "Stretching Agile to fit CMMI Level 3 - the story of creating MSF for CMMI® Process Improvement at Microsoft Corporation " *presented at Agile2005Conference*, http://agile2005.org/, 2005. (1.3.2006)

[20] S. W. Baker, "Formalizing Agility: An Agile Organization's Journey toward CMMI Accreditation " *presented at Agile2005Conference,* http://agile2005.org/, 2005. (1.3.2006)

[21] E. Bos and C. Vriens, "An agile CMM," *In proceedings of 4th Conference on Extreme Programming and Agile Methods - XP/Agile Universe*, pp. 129-138, 2004.

[22] C. Vriens, "Certifying for CMM Level 2 and ISO9001 with XP@Scrum," *In proceedings of the Agile Development Conference*, pp. 120-124, 2003.

[23] T. Kähkönen and P. Abrahamsson, "Achieving CMMI Level 2 with Enhanced Extreme Programming Approach," *In proceedings of the 5th International Conference on Product Focused Software Process Improvement*, pp. 378-392, 2004.

[24] D. Kane and S. Ornburn, "Agile Development: Weed or Wildflower?," *CrossTalk, The Journal of Defense Software Engineering*, http://www.stsc.hill.af.mil/ crosstalk/2002/10/kane.html, 2002. (1.3.2006)

[25] J. Nawrocki, W. Bartosz, and A. Wojciechowski, "Toward Maturity Model for eXtreme Programming," *In proceedings of the 27th Euromicro Conference*, pp. 233-239, 2001.

[26] M. C. Paulk, "Extreme Programming from a CMM Perspective," *Software*, vol. 18, issue 6, pp. 19-26, 2001.

[27] D. L. Johnson and J. G. Brodman, "Applying CMM Project Planning Practices to Diverse Environments," *Software*, vol. 17, pp. 40 - 47, 2000.

[28] SCAMPI, "Standard CMMI® Appraisal Method for Process Improvement (SCAMPI[SM]), Version 1.1: Method Definition Document."

[29] K. Beck and C. Andres, *Extreme Programming Explained, Embrace Change, Second Edition*, Addison-Wesley, 2004.

[30] O. Salo, K. Kolehmainen, P. Kyllönen, J. Löthman, S. Salmijärvi, and P. Abrahamsson, "Self-Adaptability of Agile Software Processes: A Case Study on Post-Iteration Workhops," *In proceedings of the 5th International Conference on Extreme Programming and Agile Processes in Software Engineering*, pp. 184-193, 2004.

[31] "Appraisal Requirements for CMMI[SM], Version 1.1. (ARC, V 1.1.), Technical Report CMU/SEI-2001-TR-034.", 2001.