DOI: 10.21917/ijms.2023.0259

AN EXPERIENCE OF IMPLEMENTING SOFTWARE DEVELOPMENT MANAGEMENT SYSTEM BY COMBINING CMMI WITH AGILE IN SMALL AND MEDIUM-SIZED ORGANIZATION

Il Nam Mun[1], Ki Son Ryang[2], Hyon A Kim[3] and Chol Hak Kim[4]

[1,2,3]Institute of Information Technology, Hightech Research and Development Center, Kim Il Sung University, DPR of Korea
[4]Korea Computer Technology Company, DPR of Korea

Abstract

Nowadays, the research on improvements of software development processes has been developing extensively in software organizations. Currently these researches are focusing on how to combine CMMI with Agile. In addition, most of the software development organizations all over the world are small or medium, not large. Such small and medium-sized organizations play a great role in the software industry. However, there is lack of case studies that small and medium-sized organizations implement CMMI process areas to establish their development management systems, especially practical and helpful case studies that such organizations may apply to combine with Agile for improving their management systems. Our goal is to share our experience in successful improvements of development process under the industrial environment from an empirical point of view. And these results cannot be generalized, but they point out that under some circumstances in small and medium-sized organizations, a combination strategy of CMMI and Agile can be a possible option for software professionals. In this paper we present our experience of implementing a DMS (Development Management System), in which CMMI is combined with Agile, for a small and medium-sized software development aiming at CMMI level 3. Moreover, we give the result that our organization applied this DMS.

Keywords:

Software Process Improvement (SPI), Agile Software Development (ASD), Capability Maturity Model Integration (CMMI)

1. INTRODUCTION

Today the most interesting subject of software process improvements is how to combine CMMI with Agile. CMMI, which is inherited in 2002 from CMM-SW developed by Software Engineering Institute (SEI) in order to describe principles and practices as the basis of software process maturity, is composed of the model, appraisal methods and training materials for software process improvements (SPI). Unlike standards that focus on process, planning, and documentation, such as ISO 9001 or CMMI, Agile is a development methodology that has recently gained popularity in the software industry, focusing on collaboration, repetition, and communication. Agile methods are lightweight methods that focus on the rapid delivery of valuable products, whereas SPI standards are too focused on quality and documentation and become heavy and overly bureaucratic [1].

Some researchers regard that Agile methods contradict the SPI standards, and some think of that these SPI standards are not reasonable or no longer useful [1]. But recent research has shown that software organizations can find and implement software development process that combine CMMI and Agile. Some scholars also assert that software organizations need both agility and discipline for them to win the success [2]. And different researchers agreed with that agile methods and the SPI standards can have complemented each other in software organizations following SPI standards such as ISO 9001 and CMMI if two approaches are combined on a proper base [1]-[7]. Much literature discusses about the combination of agile methods and the standards or the application of specific agile methods (XP or Scrum) to CMMI or ISO 9001 [4], [6], [7].

The rest of the paper is as follows. We briefly describe research results about the combination of CMMI and agile methods in Section 2. In Section 3 we describe the DMS of our organization, a Small and Medium-sized software development organization. In Section 4 we give the results of applying the proposed DMS, and Section 5 concludes the paper.

2. BACKGROUND AND RELATED WORK

International standards and models, which extensively used worldwide to improve software process, include CMMI, ISO 9001, ISO 15504 (also called SPICE) and so on [8]-[10]. In addition, the Agile approach is welcomed in small and medium-sized software development and has become a key development paradigm, and in recent years large organizations also tend to accept the ASD philosophy.

Traditional agile methods include XP (eXtreme Programming), Scrum, Crystal, DSDM (Dynamic Systems Development Model), FDD (Feature-Driven Development), Lean, the agile version of RUP (Rational Unified Process). The most well-known and the most widely used agile methods are XP and Scrum [2], [11].

Roger [12] highlights those 4 properties, such as peoples, products, processes and projects, should be focused on in order to make software development managements effective. Ambler [13] states that 5 properties, such as peoples, principles, practices, products and processes, should be dealt with, and explains their importance.

Kevin [14] poses the question of whether Small-sized software organizations are able to implement CMMI under the lack of schedule, budget and resource, and asserts the implementation possibility through some case studies.

André [15] aims to identify primary factors that may influence success of agile projects. He identified 53 practices that could affect project success by reviewing the literature and asserts that only 15 practices of them are needed to agile projects earnestly and 6 practices are related to quality, 8 practices are related to scope, and 1 practice is related to time.

Anu [11] and Fantina [16] proposed a practical and easy-to-apply methods to efficiently improve the process in terms of cost in small-size software organizations and conducted several case studies in order to validate the methods. They complement a practise which is capable of supporting the inception and success of software process improvements in small-size organizations based on lightweight

techniques for practical process modelling and identification of process improvement objects.

Taking into account of the fact that currently many software projects accept agile methods, Konstantinos [17] carries out investigation to measure how much the projects are agile, and surveys questionnaires with measurement tools such as PAM (Perceptive Agile Measurement), TAA (Team Agility Assessment), OPS (Objectives Principles Strategies), but he doesn't find validity with any tools.

Mulju [18] shows the study result conducted by a software development company to improve the technical implementation process of CMMI to cope with future CMMI level 3 certification.

Gerard [10] mentioned process maps, forms and institutionalization that arise when implementing all 22 processes of CMMI in large size software organizations.

Paul [19] introduces several experiences of improving the software process by integrating CMMI and Agile. Firstly, he introduces techniques to increase agility in CMMI mature organizations and investigates common misapplications of CMMI. Secondly, he explains how CMMI, Agile and Lean can help each other, common challenges that agile organizations encounter with when trying a certain level of CMMI process maturities, and experiences in establishing process asset stores capable of supporting both Agile and CMMI. Also, he shows how implicated 11 processes can cover 18 processes of CMMI.

Table.1. Example of Processes of an Agile Organization and Coverage of CMMI Process Areas level 2 and 3 in small-size organizations (Table.1)

Processes of an Agile Organization	**Coverage of CMMI Level 2 and Level 3 Process Areas**
Organizational Process Focus	OPF
Organizational Process Definition	OPD
Organizational Training	OT
Integrated Management Process	PP, PMC, RSKM, IPM, DAR, MA
Supplier Agreement Process	SAM
Integrated Requirements Management / Development Process	REQM, RD
Design Process	TS, DAR
Implementation Process	TS
Integration, Test and Validation Process	VER, VAL, PI
Configuration Management Process	CM
Quality Assurance Process	QA

He also presents samples of PMP (Project Management Plan) templates, Organizational Process Assets Guidelines, Process Assets Approval and Release process, Organizational Process Focus process and Organizational Process Definition process, and proposes a structure of Process Tailoring and Process Assets (Fig.1).

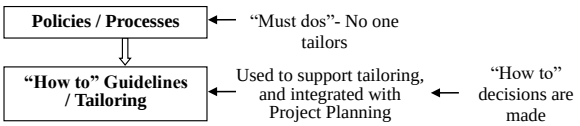

Fig.1. A Structure of Process Assets in Small and Medium-Sized Software Organizations.

As such, theoretical analysis, including consideration of CMMI and Agile, their combination has been profoundly conducted in previous research, however, there seldom are found practical, helpful to practitioners, case studies demonstrating how CMMI level 3 processes may be implemented, and especially how to incorporate Agile mode, if a Small and Medium-sized software organization target CMMI 3 level certification. In the following, we first describe the DMS that we have implemented.

3. DMS OF SMALL AND MEDIUM-SIZED SOFTWARE DEVELOPMENT ORGANIZATION

3.1 VALUES AND PRINCIPLES

The DMS of a small and medium-sized software development organization focuses on:

- Working system: To produce a working system regularly, typically at short and fixed boxes.
- Active participation of customer: To work with customer closely, every day if possible.
- Self-test: To perform walk-through without cease.
- Organization: To make teams fit for their characteristics.
- Improvement: To reconsider development practices regularly and take improvement measures timely.
- Evaluation: To evaluate teams at regular intervals and stimulate developers to greater enthusiasm.
- The DMS of a small and medium-sized software development organization follow the following principles:
- Reducing wastes – To consider any activities which do not contribute to a final product directly as waste.
- Taking care of quality problems early – To implement requirements by stages, validate their correctness and fix detected problems.
- Realizing and correcting in time –To realize timely what stakeholders want really and act based on that, although it is important to make a good plan.
- Coping with changes – To make a great point of flexible architectures which can bear up under changes and schedules which are able to cope with changes rapidly.
- Fast delivering – To deliver high-quality systems quickly and get feedback.
- Respecting developers' opinions – To respect active, careful developers' opinions and carry out the project with them as the core.
- Accompanying monitor and control – To make senior development managers grasp situations of projects through periodic inspection out of projects and take countermeasures and help them.

3.2 DEVELOPMENT LIFECYCLE AND ROLE STRUCTURE

The life cycle of DMS is shown in Fig.2 below. To implement this life cycle, an organization needs to have the role structure shown in Fig.3.

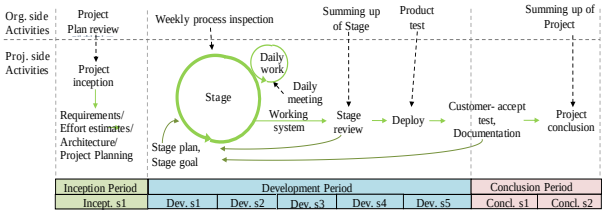

Fig.2. Development Lifecycle of DMS.

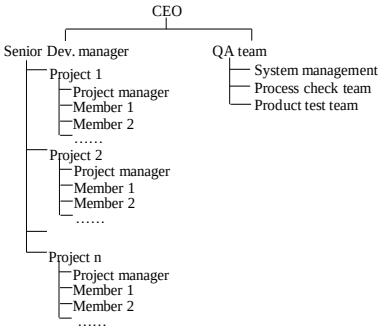

Fig.3. A Role Structure of Small and Medium-Sized Software Organization.

Responsibilities and authorities according to roles are shown in Table.2. Here, the QA team consists of a System management team, a Process inspection team, and a Product test team, which may overlap each other, and 2~3% of the total development staff of the organization is sufficient.

3.2.1 Project Side Activities:

Project side activities can be divided into Inception period, Development period and Conclusion period.

In Inception period, most of requirements are collected, the overall scope of project is defined as whole, the sufficient architecture of the system to ensure that the solution is possible is devised, and a project plan including effort estimate and work-items is made up. Inception period is composed of 1~2 stages.

Table.2. Roles, Responsibilities and Authorities

Roles	**Responsibilities and Authorities**
CEO	· Setting up General Direction of DMS · Taking charge of All work with the customers
Senior Development Manager	· Managing and reporting all projects · Managing development environment
QA Team	·Forming and executing Annual DMS plan ·Organizing Internal Audit
System Management Team	· Performing Internal Audit · Identifying and Process Executing Improvements
Process Inspection Team	Objectively monitoring, grasping and reporting the status of projects and processes
Product Test Team	Objectively testing and reporting product quality

In Development period, full-scale development is carried out, and most of efforts are consumed. Development period is composed of several stages, and a deliverable system is produced in each stage. In the beginning of Development period, functionalities of the suggested software are confirmed in more details to specify the elaborate scope of project, and the architecture is refined. In each stage of the development phase, under consultation with stakeholders, the goal of the stage is set by selecting the highest priority requirements among the entire unresolved requirements and assigning tasks to achieve it. Also, development and management of requirements, analysis and design, implementation, integrating and testing, deployment, configuration management and risk management are iterated, and a working subsystem is deployed in the field, if necessary (Fig.4).

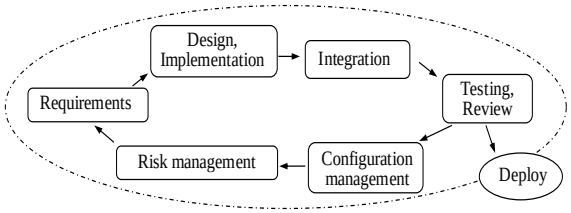

Fig.4. Activities Iterated in Each Stage.

Conclusion period is the period of deploying a produced system in the field on a full scale and it is composed of 1~2 stages. It includes acceptance testing of customer, supporting operation including customer education about product, documentation.

In the beginning of each stage of projects, the stage goal is corrected (correction of the goal of the stage in the early project plan) and stage plan is made up, in the end of the stage, stage review is performed, in which the subsystem, developed in the stage, is demonstrated to stakeholders.

Throughout the project, teams have daily meetings for less than 30 minutes. In the daily meetings, teams understand what was achieved by each member since last meeting, what is going to achieve, and which problems are posed, and so on are figured out, and actions are taken.

3.2.2 Organization Side Activities:

Organization side activities include project proposal audit, project plan review, weekly process inspection, summing up of stage, product inspection and summing up of project. Project proposal audit decides whether to execute the proposed project or not. Result of the audit comes out based on gain from the project versus resource to invest in the project, and the feasibility of the project. Project plan review is composed of initial review and stage review. Initial review is to examine the project plan at the early days of the project. The correctness of resource estimates including members to participate in the project and effort estimates to be budgeted, the possibility to attain the project goal, and the reality of stage goals as milestones of the project, and so on, are reviewed. In stage review, focus is placed on validating the correctness of goal of the current stage. Weekly process inspection is performed by the process inspector in charge. The process inspector in charge shall examine the current status of the project according to the process inspection index and notify the CEO. Summing up of stage is composed of the assessment of the goal attainment of the stage and quality assessment and is performed by unit development manager and process inspector in charge (Fig.5).

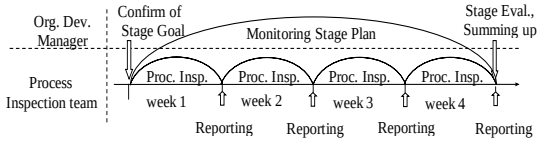

Fig.5. Activities of Process Inspection Team and Senior Dev. Manager in Each Stage

- Assessment of the goal attainment of the stage is an assessment of the implementation of requirements selected as the stage goal. That is, an objective assessment is made by examining whether the requirements are implemented and whether the functions work properly. In addition, we estimate the progress of the project by computing the ratio of the implemented requirements among the entire requirements of the project.
- Quality assessment is an assessment of the process of the project. That is, an objective assessment is made by analyzing the requirements and analyzing the specifications, whether the detailed specifications have been drawn up, implemented and integrated, whether a peer review of the written codes has been carried out, whether a self-test (unit test, integration test) has been performed for all functions, whether risk management is performed, and whether the configuration management is conducted properly.

Product inspection is an objective inspection performed by product inspector of the organization before products of the project are delivered to customer or are deployed in the field.

3.3 ADAPTATION FACTORS

In applying the above lifecycle, it is important to grasp the complexities of the conditions that projects might encounter with and adapt the projects to the conditions.

Five factors, such as Team size, Geographical distribution,

Organizational distribution, Domain complexity and technical complexity, should be adapted. If all the 5 factors are simple, the project can be managed with the above lifecycle. But, if any factor is complicated, then the lifecycle should be tailored fitting to the situation, and in some cases, new additional practices should be introduced to cope with possible risks.

For example, if a team has many developers of dozens, obstacles might arise in face-to-face communication among them, and daily meetings might be burdensome. It seems be pertinent to cope with such cases by a sub-team encapsulation method, as follows:

- A team is divided into several smaller sub-teams.
- Each sub-team doesn't need to know how the other sub-teams are composed of. Each sub-team has a "mediator", and all the sub-teams communicate through only mediators.
- Mediators are always available. They should be as accessible as any other member of a sub-team. This means each mediator is a member of two or more sub-teams.
- A mediator is hot a leader of development and serves merely as a communicator. He does not tell his colleagues what to do. Information does not travel up horizontally, but vertically.
- Each sub-team has a well-defined responsibility area, which overlaps with the other sub-teams' responsibilities as little as possible.

3.4 IMPLEMENTATION OF CMMI PROCESS ARWAS

To reach the CMMI 3 level, the development organization needs some more activities than the activities of the above development life cycle (see Fig.6).

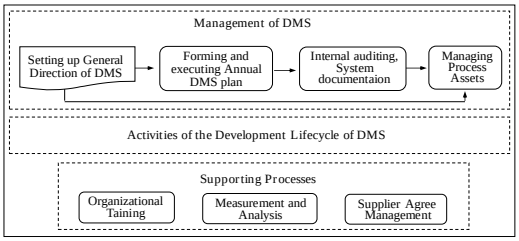

Fig.6. Overall DMS of Small and Medium-Sized Software Organization Complying with CMMI Level 3.

Here, explanations of activities in Fig.6 are omitted. However, to attain CMMI level 3, all the activities of the above DMS must be documented in the form of process definitions, procedures, guidelines, formats.

The Table.3 shows the relationship that activities of the DMS above mentioned are mapped into Specific Goals (Specific Practices) of CMMI (CMMI-DEV1.3) level 3.

Table.3. Mapping of CMMI Level 3 Processes into the Activities of the DMS

Process	Level	Specific Goal (Specific Practices)	Satisfaction Evidence for the Goal and Practices
Requirements Management	2	Manage Requirements (5)	Requirements collection, work-items, Requirements definition
Project Planning	2	Establish Estimates (4)	Effort estimate, resources estimate
		Develop a Project Plan (7)	Initial plan, stage plan, stage goal
		Obtain Commitment to the Plan (3)	Review and approval of project plan, stage goal and plan
Project Monitoring and Control	2	Monitor the Project Against the Plan (7)	Weekly process inspection, summing up of stage
		Manage Corrective Action to Closure (3)	Record of process inspecting, weekly and monthly summing up
Supplier Agreement Management	2	Establish Supplier Agreements (3)	Supplier Agreement Management
		Satisfy Supplier Agreements (3)	Supplier Agreement Management
Measurement and Analysis	2	Align Measurement and Analysis Activities (4)	Measurement and Analysis according to the metrics specified
		Provide Measurement Results (4)	Measurement and Analysis according to the metrics specified
Process and Product Quality Assurance	2	Objectively Evaluate Processes and Work Products (2)	Weekly process inspection, summing up of stage
		Provide Objective Insight (2)	Report of weekly process inspecting monthly quality evaluation
Configuration Management	2	Establish Baselines (3)	Configuration Management plan, Use of SVN
		Track and Control Changes (2)	Configuration item Management, change management
		Establish Integrity (2)	Project repository, repository management
Requirements Development	3	Develop Customer Requirements (2)	Agree with stakeholders
		Develop Product Requirements (3)	Requirements definition, Requirements specification

		Analyze and Validate Requirements (5)	Review of Requirements definition, Requirements specification
Technical Solution	3	Select Product Component Solutions (2)	Design, analysis
		Develop the Design (4)	Develop structural design, detailed design and design review
		Implement the Product Design (2)	Coding, Coding standards
Product Integration	3	Prepare for Product Integration (3)	Using product integration servers
		Ensure Interface Compatibility (2)	Daily or weekly product integration
		Assemble Product Components and Deliver the Product (4)	Daily or weekly product integration
Validation	3	Prepare for Validation (3)	Product inspection by organization, acceptance testing
		Validate Product or Product Components (2)	Product inspection by organization, acceptance testing
Verification	3	Prepare for Verification (3)	Test plan, test cases, review, Pair programming
		Perform Peer Reviews (3)	Review of Requirements specification, design and code, Collective code ownership
		Verify Selected Work Products (2)	Unit test, integration test, system test
Organizational Process Focus	3	Determine Process Improvement Opportunities (3)	Internal audit and corrective actions, Annual DMS plan
		Plan and Implement Process Actions (2)	Devise and execute Annual Improvement plan
		Deploy Organizational Process Assets and Incorporate Experiences (4)	Manage Process Assets
Organizational Process Definition	3	Establish Organizational Process Assets (7)	System documentation
Organizational Training	3	Establish an Organizational Training Capability (4)	Organizational Training
		Provide Training (3)	Organizational Training
Integrate-d Project Management	3	Use the Project's Defined Process (7)	Proposal audit, project planing and conclusion of Project
		Coordinate and Collaborate with Relevant Stakeholders (3)	Customer communication plan
Risk Management	3	Prepare for Risk Management (3)	Risk management plan
		Identify and Analyze Risks (2)	Weekly, monthly process inspection, stage summing up
		Mitigate Risks (2)	Mitigate Risks
Decision Analysis and Resolution	3	Evaluate Alternatives (6)	Project proposal audit

4. APPLICATION EVALUATION

In the following, for projects carried out in our organization from 2018 to 2021 we analyze the results in terms of schedule, quality, and cost, which are three key indicators of project success judgment. Since 2020 we have adopted the DMS proposed in this paper.

4.1 SCHEDULE EVALUATION

Schedule evaluation was determined by the period compliance of projects carried out from 2015 to 2020 (Table.5). The calculation basis is as follows (See Table.4).

For each project, the start date, the plan end date, the real end date, the plan days, and the real days are denoted by DS_i, $DEplan_i$, $DEreal_i$, $NDplan_i$, $NDreal_i$ (where i is the index representing the project), and for each year, the number of projects, the plan days, and real days of project are denoted by n_k, $NDYplan_k$, $NDYreal_k$ (k=1,2,3,4) (where k is the index representing the year, indicating the years 2018, 2019, 2020, and 2021, respectively). Then, the period compliance RSA_k is calculated as following.

$$RSA_k = 100 \times NDYplan_k / NDYreal_k \ (k=1,2,3,4),$$

$$NDYplan_k = \sum_{i}^{n_k} NDplan_i$$

$$NDplan_i = DEplan_i - DS_i,$$

$$NDreal_i = DEreal_i - DS_i.$$

In the Table.5, the improvement curve is an approximate curve of the type $y=ax+b$ for the period compliance, representing the improvement rate from 2018 to 2021.

Table.4. Duration of the Projects

No	Project name	Start date	End date		Number of days	
			Plan	Real	Plan	Real
1	P2018-1	2017.5.21	2017.12.31	2019.6.30	225	771
2	P2018-2	2017.1.10	2017.12.30	2018.12.24	355	714
3	P2018-3	2017.2.1	2017.5.30	2018.4.15	119	439
….						
25	P2018-25	2018.11.29	2019.5.31	2019.7.31	184	245
2018 Sum					4740	8106
1	P2019-1	2018.8.20	2019.6.21	2019.9.21	306	398
2	P2019-2	2019.1.4	2019.6.30	2019.6.30	178	178
3	P2019-3	2019.1.1	2019.6.30	2019.7.5	188	185
…						
23	P2019-23	2019.11.1	2019.12.30	2020.1.14	60	75
2019 Sum					3895	4693
1	P2020-1	2019.11.1	2020.6.30	2020.7.1	243	244
2	P2020-2	2019.12.25	2020.11.30	2021.1.21	342	394
3	P2020-3	2020.1.15	2020.6.30	2020.7.1	168	169
						…
17	P2020-17	2020.9.17	2020.12.15	2021.1.21	90	127
2020 Sum					2899	3773
1	P2021-1	2021.1.1	2021.12.15	2022.1.4	349	370
2	P2021-2	2021.1.4	2021.12.20	2021.12.30	351	361
3	P2021-3	2021.1.4	2021.12.20	2021.12.30	351	361
…						
16	P2021-16	2021.11.1	2021.12.15	2021.12.30	45	60
2021 Sum					3847	4622

4.2 QUALITY EVALUATION

Quality was determined by evaluating the number of defects found in the product test for projects carried out from 2018 to 2021 (Table.7). The calculation basis is as Table.6.

Table.5. Schedule Evaluation

Index \ Year	2018	2019	2020	2021
Number of Projects Carried out n_k	25	23	17	16
Plan Days of Project $NDY\,plan_k$	4740	3895	2899	3847
Real Days of Project $NDY\,real_k$	8106	4693	3773	4622
Period Compliance (%) RSA_k	58.48	83.00	76.83	83.23
Improvement Curve	65.17	71.98	78.79	85.60
Improvement Rate = 31.34(%)	a=-6.808		b=58.361	

For each project, the man-days and defects in the product test are denoted by PE_i, PF_i (where i is the index representing the project) and for each year, the man-days and defects in the product test are denoted by PEY_k, PFY_k (k=1,2,3,4). NFP_k (the number of defects per project) and NFE_k (the number of defects per 300 man-days) are calculated as follows:

$$NFP_k = PFY_k/n_k,$$

$$NFE_k = 300\times PFY_k/PEY_k \ (k=1,2,3,4),$$

$$PEY_k = \sum_{i=1}^{n_k} PE_i \text{ and } PFY_k = \sum_{i=1}^{n_k} PF_i .$$

In the Table.7, the improvement curve is an approximate curve of the $y=ax+b$ type for the number of defects per 300 man-days, representing the improvement rate from 2018 to 2021. The above evaluation is graphically expressed as Fig.8.

4.3 COST EVALUATION

Cost evaluation was determined by calculating the ratio of the plan man-days to the real man-days of projects carried out from 2018 to 2021 (Table.9).

Table.6. Man-days of the Projects and Number of Defects in the Product Test

No	Project Name	Project Man-days	Number of Defects
1	P2018-1	370.0	5
2	P2018-2	942.5	17
3	P2018-3	137.0	10
25	P2018-25	336.0	4
2018 Sum		13270	164
1	P2019-1	506.0	10
2	P2019-2	649.0	9
3	P2019-3	359.0	5
23	P2019-23	176.0	4
2019 Sum		12759.6	151
1	P2020-1	343.0	11
2	P2020-2	1167.0	6
3	P2020-3	396.0	7
17	P2020-17	150.0	3
2020 Sum		7096.3	97
1	P2021-1	742.0	9
2	P2021-2	2744.3	11
3	P2021-3	575.5	8
…	……		
16	P2021-16	79.0	6
2021 Sum		10766.8	104

Table.7. Quality Evaluation

Index \ Year	2018	2019	2020	2021
Number of Projects Carried out n_k	25	23	17	16
Project Man-days PEY_k	13270	12759	7096	10766
Defects PEY_k	164	151	97	104
Number of Defects per Project NFP_k	6.56	6.57	5.71	6.50
Number of Defects per 300 Man-days NFE_k	3.71	3.55	4.10	2.90
Improvement Curve	3.85	3.66	3.47	3.28
Improvement Rate = 14.66 (%)	a=-0.187 90		b=4.033 847	

The calculation basis is as Table 8.

Table.8. The Man-days of Projects

No	Project Name	Plan Man-days	Real Man-days
1	P2018-1	170	370
2	P2018-2	890	942.5
3	P2018-3	110	137
…	……		
25	P2018-25	190	336
2018 Sum		11120	13270
1	P2019-1	450	506
2	P2019-2	640	649
3	P2019-3	380	359
…	……		
23	P2019-23	160	176
2019 Sum		10800	12759.6
1	P2020-1	340	343
2	P2020-2	1010	1167
3	P2020-3	390	396
…	……		
17	P2020-17	130	150
2020 Sum		6190	7096.3
1	P2021-1	700	742
2	P2021-2	2500	2744.3
3	P2021-3	560	575.5
…	……		
16	P2021-16	70	79
2021 Sum		8760	10766.8

Project plan man-days and real man-days are denoted by $PEplan_i$, $PEreal_i$ (where i is the index representing the project) respectively, and for each year, the number of projects, plan man-days and real man-days are denoted by n_k, $NPYplan_k$, $NPYreal_k$ (k=1,2,3,4).

Then, the man-days-ratio RPA_k is calculated as following.

$$RPA_k=100\times NPYplan_k/NPYreal_k \ (k=1,2,3,4),$$

$$NPYplan_k = \sum_{i=1}^{n_k} PEplan_i \text{ , and } NDYreal_k = \sum_{i=1}^{n_k} PFplan_i .$$

Table.9. Cost Evaluation

Index \ Year	2018	2019	2020	2021
Number of Projects carried out n_k	25	23	17	16
Project Plan Man-days $NPYplan_k$	11420	10600	6010	8860
Project Real Man-days $NPYreal_k$	13270	12759	7096	10766
Man-days Ratio(%) RPA_k	86.06	83.07	84.7	82.29
Improvement Curve	85.48	84.51	83.54	82.58
Improvement Rate = -3.51(%)	a=-0.97		b=86.45	

In the above table, the improvement curve is an approximate curve of the $y=ax+b$ type for the man-days ratio, representing the improvement rate from 2018 to 2021. As the above evaluations show, we believe that applying the proposed DMS has resulted in significant improvements in the schedule and quality aspects of the projects. However, there was no improvement in terms of cost. From a practical point of view, this paper gives the following implications. First, it provides the basis for a framework that can help organizations to embrace both high maturity standards and agile practices. In addition, the results of this paper will be valuable not only for improving software development processes but also for other organizations to increase their maturity. However, this empirical study was carried out only in the organization seeking higher maturity based on ISO 9001 and CMMI. Also, in the application evaluation, we tried to analyze different projects of the organization, but the data collected are limited. As such, the complete observation of the actual phenomenon and the production of accurate conclusions may have been limited. More importantly, the improvements described above may be due to factors other than applying the proposed DMS.

5. CONCLUSION

In this paper, we propose DMS suitable for small and medium-sized software development organizations and shows the results applied to our organization. The proposed DMS highlight the characteristics of small and medium-sized software development organizations, while taking advantage of both CMMI and Agile, but there are several shortcomings. We will further refine the proposed DMS by properly combining the advantages of agile approach and the characteristics of small and medium-sized software development organizations.

REFERENCES

[1] T. Stalhane and G.K. Hanssen, "The Application of ISO 9001 to Agile Software Development", *Proceedings of International Conference on Product-Focused Software Process Improvement*, pp. 371-385, 2008.

[2] B. Boehm and R. Turner, "Balancing Agility and Discipline: A Guide for the Perplexed", Addison Wesley, 2003.

[3] S. Akbarinasaji, B. Caglayan and A. Bener, "Predicting Bug-Fixing Time: a Replication Study Using an Open Source Software Project", *Journal of Systems and Software*, Vol. 136, pp. 173-186, 2018.

[4] H. Wei, H.C.Z. Hu, S.Y. Chen, Y. Xue and Q.X. Zhang, "Establishing a Software Defect Prediction Model Via Effective Dimension Reduction", *Information Sciences*, Vol. 477, pp. 399-409, 2019.

[5] A.S.C. Margal, B.C.C. De Freitas, F.S.F. Soares and A.D. Belchior, "Mapping CMMI Project Management Process Areas to SCRUM Practices", *Proceedings of IEEE Software Engineering Workshop*, pp. 13-22, 2007.

[6] M. Fritzsche and P. Keil, "Agile Methods and CMMI: Compatibility or Conflict?", *e-Informatica Software Engineering Journal*, Vol. 1, No. 1, pp. 9-26, 2007.

[7] C. Santana, C. Gusmao, L. Soares, C. Pinheiro, T. Maciel, A. Vasconcelos and A. Rouiller, "Agile Software Development and CMMI: What We Do Not Know about Dancing with Elephants", *Proceedings of International Conference on Agile Processes in Software Engineering and Extreme Programming*, pp. 124-129, 2009.

[8] CMMI Product Team, "CMMI-DEV", V2.0, Software Engineering Institute (SEI), Carnegie Mellon University, Available at https://resources.sei.cmu.edu/asset_files/technicalreport/2010_005_001_15287.pdf, Accessed at 2010.

[9] H. Turabieh, M. Mafarja and X.D. Li, "Iterated Feature Selection Algorithms with Layered Recurrent Neural Network for Software Fault Prediction", *Expert Systems with Applications*, Vol. 122, pp. 27-42, 2019.

[10] G. O'Regan, "Introduction to Software Process Improvement", Springer, 2011.

[11] A. Raninen, "*Practical Process Improvement: How to Initiate Software Process Improvement in a Small Company*", Ph.D. Dissertation, Department in Forestry and Natural Sciences, University of Eastern Finland, pp. 1-198, 2014.

[12] S.P. Roger, "Software Engineering-A practitioner's Approach", McGraw-Hill, 2001.

[13] S.W. Ambler, "Scaling Agile: An Executive Guide", IBM Rational Software, Agility at Scale Whitepaper, 2010.

[14] S. Kevin, "Capability Maturity Model Integration (Cmmi) for Small Organizations", Ph.D. Dissertation, Department of Management Studies, Regis University, pp. 1-178, 2019.

[15] H. Andre, "Agile Project Management. a Case Study on Agile Practices", Master's Thesis, Department of Business Administration, Arctic University of Norway, pp. 1-67, 2016.

[16] R. Fantina, "Practical Software Process Improvement", Artech House, 2005.

[17] C. Konstantinos, "Measuring Agility-A Validity Study on Tools Measuring the Agility Level of Software Development Teams", Master of Science Thesis, Department of Software Engineering, University of Gothenburg, pp. 1-99, 2015.

[18] T. Mulju, "Improvement of a Technical Solution Process in a Software Company", Master's Thesis, Department of Management Studies, Helsinki Metropolia University of Applied Sciences, pp. 1-89, 2013.

[19] E.M. Paul, "Integrating CMMI and Agile Development", Addison-Wesley, 2010.