Software Process Improvement by Changing Roles in Small Size Organizations

Gi-Son RYANG (1), Il-Nam MUN (2)

(1) Institute of Information Science, Kim Il Sung University (D.P.R. of Korea)
E-mail: ryanggison@126.com
(2) Institute of Information Science, Kim Il Sung University (D.P.R. of Korea)
E-mail: munilnam@126.com

ABSTRACT

Agile methodology has become more popular in small size software organizations and many case studies and best practices taking the advantage of agile principles have been presented. Research for developing effective methods which make it possible to achieve continuous process capability improvement complying with CMMI in small size software organizations is being continued. In this paper we propose an organization structure where a Process Manager and Process Management Supporters play important roles in organizational-level processes. Case studies are presented to demonstrate the effectiveness of the proposed method.

Key words: **Agile Software Development (ASD), Capability Maturity Model Integration (CMMI), Software Engineering, Software Process Improvement (SPI)**

1- INTRODUCTION

Today small size software organizations claim a sizable share and take considerable position in the software industry. Most of these organizations apply agile methodology in consonance with the realities of their organizations. In such organizations the most pressing task is to have high process capability maturity through systematic process improvement. Specially, strategic objectives of most software organizations are to appraise their own organizations' process capability maturity against CMMI and achieve higher level of CMMI. From these needs, major subject in the software industry is the relation between agile methodology and CMMI, coexistence of two methods and practical solution to make use of advantages of both these methods [1]-[5].

From its nature, software development needs discipline to manage complexity, and flexibility to deal with various changes timely. While agile methods, such as XP and Scrum, intend flexibility and high speeds, CMMI ensures discipline. Thus software organizations are recommended to accept agile methodology to achieve high speed and efficiency and should comply with CMMI to establish well-ordered management systems for their organizations. In particular, for small software organizations, an important development strategy is to be appraised at higher level of CMMI and get more recognition for their

capability and position. Studies and applications about agile methodology get more extensive in order to maintain high productivity and efficiency, reduce software development cost and improve the quality [1]-[11].

In this paper we introduced an organization structure for small size software organizations and specified each individual's role in this structure for complying with process areas of CMMI without affecting their own agility.

The rest of the paper is as follows. We briefly describe agile methodology and CMMI in section 2. In section 3 we explain main results of the paper, i.e., a change of roles for complying with process areas of CMMI without affecting their own agility in small size software organizations. In section 4 we present four case studies which demonstrate the effectiveness of our method.

2- BACKGROUND AND RELATED WORK

2.1 AGILE SOFTWARE DEVELOPMENT

Agile Software Development emerged in February 2001 when a group of software consultants signed the Agile Software Development Manifesto. Agile methods focus on the challenges of unpredictability of the real world by relying on people and their creativity rather than processes. The main theme in agile methods is to promote and speed up responses to changing environments, requirements and meeting the deadlines. The agile manifesto states the main focus of the agile development as the following:

1) Individuals and interactions over processes and tools.
2) Working software over comprehensive documentation.
3) Customer collaboration over contract negotiation.
4) Responding to change over following a plan.

The most common methods are extreme Programming (XP), Dynamic Software Development Method (DSDM), Scrum, and Crystal [3].

The characteristics of agile methods are elaborately defined in the twelve principles behind the agile manifesto:

- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- Business people and developers must work together daily throughout the project.
- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

• Working software is the primary measure of progress.

• Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

• Continuous attention to technical excellence and good design enhances agility.

• Simplicity - the art of maximizing the amount of work not done – is essential.

• The best architectures, requirements, and designs emerge from self-organizing teams.

• At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly [3].

Table 1. Description of Main Agile Development Methods with References [12]

Agile Method	Description	Reference
extreme Programming	Focuses on best practice for development. Consists of twelve practices.	[11]
SCRUM	Focuses on project management in situations where is difficult to plan ahead, with an importance on feedback mechanisms	[8]
Crystal methodologies	Crystal Clear focuses on communication in small teams developing software that is not life-critical.	[6]
Dynamic Software Development Method (DSDM)	It divides projects into three phases: pre-project, project life-cycle, and post project	[8]
Feature driven development	Combines model-driven and agile development with emphasis on iterative design.	[9], [13]
Lean software development	It consists of seven principles: eliminate waste, amplify learning, decide as late as possible, deliver as fast as possible, empower the team, build integrity, and see the whole	[10]

As you can see, Scrum is suitable for self-organized teams, and for organizations that use Scrum it might be best reasonable to initiate their process improvement based on CMMI.

2.2 CMMI

The Capability Maturity Model for Software (CMM) [14][15] developed by the Software Engineering Institute (SEI) has had a major influence on software process and quality improvement around the world [16]. Based on the first version released 1991, the Capability Maturity Model – Integrated (CMMI) has been presented in 2000, integrating CMM for Software (SW-CMM), the

Capability Model for Systems Development (EIA/IS 731) and the CMM for Integrated Product Development (IPD-CMM).

Software Process Improvement (SPI) assumes that a well-managed organization with a defined engineering process is more likely to produce software that consistently meets the users' requirements within schedule and budget than a poorly managed organization with no such engineering process. "In other words, a project failure is usually a process failure" [6]. CMMI – as SPI's "de facto method" [17] – describes managerial processes to attack software development difficulties at five maturity levels: CMMI for Development consists of best practices that address development and maintenance activities applied to products and services. It addresses practices that cover the product's lifecycle from conception through delivery and maintenance.

- initial
- managed
- defined
- quantitatively managed
- optimizing

It is important to note that the CMMI process models do not contain prescriptive processes that can be used right out of the box. Instead, CMMI provides a way to assess the state of an organization's ability to build software in a repeatable, predictable way [18].

Applying CMMI as a means to increase process capabilities is an organization-wide challenge. Herbsleb et al. showed that the average time for an organization to move up one level is between 21 and 37 months [19]. Over three quarters of the organizations reported that implementing any key SPI activity took longer than expected. But the effort pays off since "software process management maturity is positively associated with project performance" [20].

In order to reach a certain level, an organization has to fulfil all process areas of that level as well as those of lower levels. A process area is a summary of all requirements for a certain topic, e.g. project management, organizational training or Causal Analysis and Resolution. To satisfy a process area all of its associated goals – specific ones and generic ones – have to be met. Specific goals apply to a process area and address the unique characteristics that describe what has to be implemented to satisfy the process area. To meet a specific goal CMMI suggests a set of specific practices. A specific practice is an activity that is considered important in achieving the associated specific goal. Generic goals are called "generic" because the same goal statement appears in multiple process areas. In the staged representation, each process area has only one generic goal. To meet a generic goal, CMMI suggests a set of generic practices. Generic practices provide institutionalization to ensure that the processes associated with the process area will be effective, repeatable, and lasting [21].

2.3 COMBINATION OF AGILE METHODOLOGY AND CMMI

Many organizations demand CMMI compliance of projects where agile methods are employed. In this situation it is necessary to analyse the interrelations and mutual restrictions between agile methods and approaches for software process analysis and improvement. Martin Fritzsche and Patrick Keil analysed to what extent the CMMI process areas can be covered by XP and where adjustments of XP have to be made. Based on this, they described the limitations of CMMI in an agile environment and showed that level 4 or 5 are not feasible under the current specifications of CMMI (They analysed CMMI v1.2.) and XP [3].

K. Lukasiewicz and J. Miler proposed the CMMI–Scrum reference model, which maps Scrum practices onto 123 practices of CMMI staged levels 2 and 3. For 60% of CMMI practices, which are insufficiently covered by Scrum they added new practices that improve discipline while maintaining agility. The practices to improve an actual software development process were selected from the reference model with the P–Sel algorithm based on answers to a questionnaire with 25 single-choice questions. They have applied their approach to processes of two IT organizations, where on average 72% of the suggested practices were confirmed, 24.5% were mismatched and 3.5% were rejected [2].

Julio Ariel Hurtado Alegr´ıa and Mar´ıa Cecilia Bastarrica explored the possibility for software organizations of getting a CMMI certification of their processes by applying agile practices. For this purpose, starting with CMMI maturity level 2 generic goals and practices, they analysed the applicability of a series of agile methods, identifying their individual or combined contribution in the fulfilment of each process area. They presented an application case where a small organization applied a combination of XP and Scrum for implementing the requirement management area [2].

Paul E. McMahon exposed characteristics of agile misapplications common in growing “Agile-like” organizations and shared how the CMMI can help these organizations. He also shared how the CMMI can help even successful growing organizations that are applying fundamental Agile practices as intended and provided numerous options to traditional “how-to” approaches to implement CMMI practices [5].

3- CHANGING ROLES IN SMALL SIZE SOFTWARE DEVELOPMENT ORGANIZATIONS

3.1 COVERAGE OF CMMI PROCESS AREAS BY AGILE METHODOLOGIES

The 22 process areas in CMMI can be divided into Project Management Process Area, Support Process Area, Engineering Process Area and Process Management Process Area [15].

Martin Fritzsche and Patrick Keil showed in detail that some of the CMMI

process areas are supported by XP and Scrum, and some are in conflict. And they also gave a summary on the coverage of CMMI process areas by XP and Scrum [3].

Table 2. Coverage of CMMI-DEV process areas by XP and Scrum (Project M: Project Management, Process M: Process Management)

Process Area	Category	XP	Scrum
Requirements Management	Project M	+++	+++
Project Planning	Project M	+++	+++
Project monitoring and control	Project M	+++	+++
Supplier Agreement Management	Project M	0	0
Measurement and Analysis	Support	+	+++
Process and Product Quality Assurance	Support	+	0
Configuration Management	Support	+++	0
Requirements development	Engineering	++	++
Technical Solution	Engineering	+++	0
Product Integration	Engineering	+++	0
Verification	Engineering	+++	0
Validation	Engineering	+++	+++
Organizational Process Focus	Process M	-	-
Organizational Process Definition	Process M	0	0
Organizational training	Process M	++	+
Integrated Project Management	Project M	++	+++
Risk management	Project M	+++	+++
Decision Analysis and Resolution	Support	-	-
Organizational Process Performance	Process M	-	-
Quantitative Project Management	Project M	-	-
Organizational Performance Management	Process M	-	-
Causal Analysis and Resolution	Support	0	0

For the coverage of specific goals, process areas and generic practices, a rating system is applied:

- Conflicting (–)
- Not addressed (0)
- Partially supported (+)
- Supported (++)
- Largely supported (+++)

As you can see, Organizational-level Process Areas(OPAs) such as Process Area “Supplier agreement management”, “Organizational Process Focus”, “Organizational Process Definition”, “Integrated supplier management”, “Decision Analysis and Resolution” and Level 4, 5 process areas are not covered by agile methods like XP and Scrum. The major cause is that these process areas focus on organizational management, not purely related with project management. Therefore, in order to establish and execute these process areas, organization’s management system should be improved, and it

is important to note that the advantage of agile development shouldn't become weakened when implementing CMMI process areas in small size software developments.

Table 3: Coverage of CMMI generic practices by XP and Scrum

Generic Practices	XP	Scrum
2.1 Establish an organizational policy	0	0
2.2 Plan the process	0	0
2.3 Provide resources	+	+++
2.4 Assign responsibility	+++	+++
2.5 Train people	+++	+++
2.6 Manage configurations	++	0
2.7 Identify and involve relevant stakeholders	+++	+++
2.8 Monitor and control the process	++	++
2.9 Objectively evaluate adherence	+	0
2.10 Review status with higher level management	++	+++
3.1 Establish a defined process	0	0
3.2 Collect improvement information	-	

3.2 PROCESS MANAGER

In small size software organizations, development and management systems are usually structured as follows. (Fig.1)

In order to implement organizational-level processes smoothly in small size software organizations with above organizational structure it is reasonable to assign a Process Manager. The Process Manager carries out process management activities professionally.

The Process Manager is responsible for:

- measuring and analysing current processes and identify their advantage and weakness
- planning and implementing process improvement programs
- selecting outcomes for analysis, analysing causes, and recording causal analysis data
- guiding decision-and-analysis
- collecting and analysing quantitative data on projects

Like Scrum teams, Project teams that use agile methods perform quality assurance activities such as review and testing in accordance with their own strict order. If QA director holds with a Process Manager concurrently in the organization, the organization would be able to implement organizational-level processes by itself without hiring a new person from the outside.

3.3 PROCESS MANAGEMENT SUPPORTER

We select a Scrum team, a typical agile team consisting a ScrumMaster, a product owner, and developers.

The ScrumMaster is responsible for training the team about Scrum, ensuring

the members follow the practices established by the team.

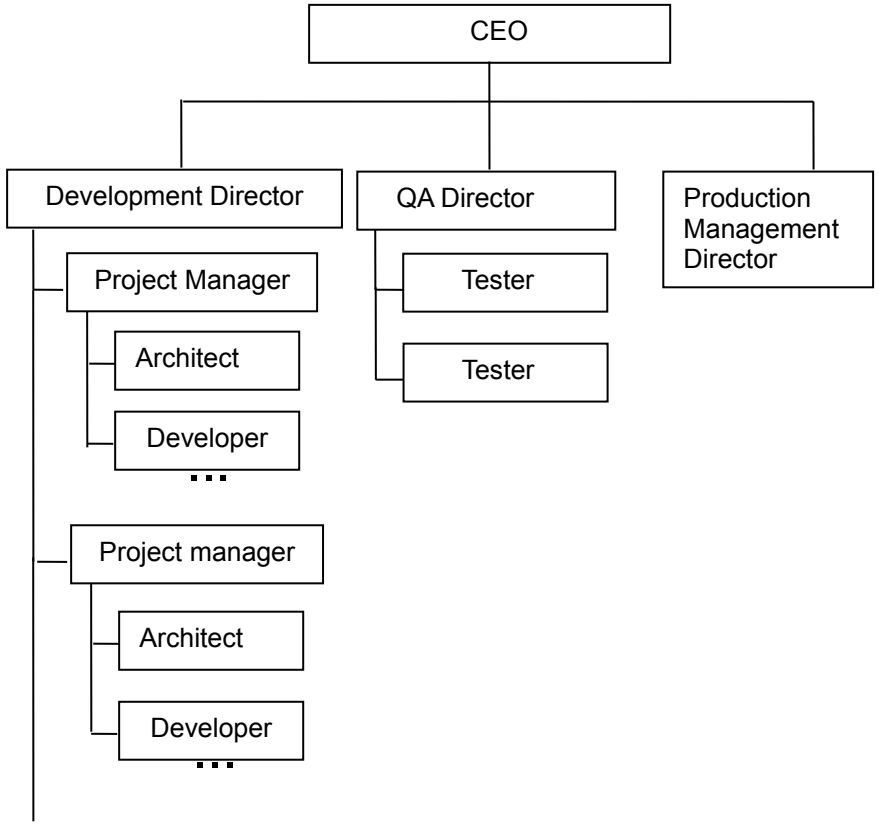

Figure 1 Development and management systems in small size software organizations.

The ScrumMaster is responsible for:

- Ensuring impediments are addressed.
- Monitoring progress.
- Facilitating planning, reviews, and retrospectives.
- Encouraging continual improvement.
- Helping stakeholders and teams communicate.

The product owner is responsible for communicating the vision of the product and maximizing the return on investment (ROI). The product owner maximizes ROI by establishing and prioritizing desirable features in the product backlog.

The product owner is responsible for:

- Managing the ROI for the product.

- Establishing a shared vision for the product among the customers and developers.
- Knowing what to build and in what order.
- Creating release plans and establishing delivery dates.
- Supporting sprint planning and reviews.
- Representing the customers, including the player who buys the product.

The team delivers a set of features from the product backlog every sprint. Developers are self-organizing and self-managing; they determine how much work they can commit to at the start of each sprint and take responsibility to deliver the completed work by the end. The team includes persons necessary to complete the goals that the team commits to for a sprint.

With above structure and roles, the Scrum team can manage its own project successfully, but it has a serious shortcoming in a sense of organizational need of pushing for higher level of CMMI. The shortcoming is that the team cannot be connected to organizational-level processes organically.

In other words, the team cannot

- help the organization to manage organizational-level processes through measuring and reporting the status of project quantitatively.
- help the organization to verify objectively, practically how organizational-level processes affect project teams.
- offer evidences for validity and effectiveness of organizational-level processes.

From this, we suggest a Process Management Supporter in the organization. Process Management Supporter is responsible for performing quantitative measurement that is needed to manage and improve organizational-level processes.

Someone might think that an existing team member (Scrum master or architect) could take on responsibilities of a Process Management Supporter, or a role of a Process Management Supporter could be added to Scrum team. But we abandon such thoughts because any of the two ways might cause changing the structure of Scrum team, and might be against principles of agile development. We need members available for helping organizational-level process implementation, so we think it is more efficient to add Process Management Supporters to the organization. The better way may be that testers in a QA group take on responsibilities of Process Management Supporter, get training, and take charge of some projects.

A new organizational structure and roles of small size software organizations with a Process Manager and Process Management Supporters are presented in Fig.2.

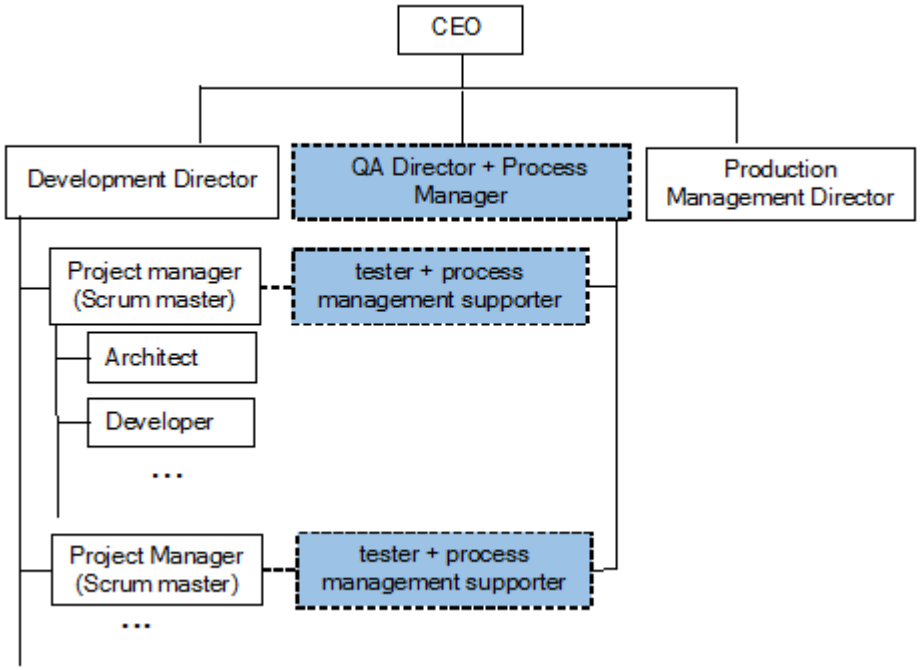

Figure 2 New Organization structure for CMMI-compiliant process improvement in small size software organizations.

3.4 PERFORMERS OF PROCESS IMPLEMENTATION IN THE ORGANIZATIONAL-LEVEL PROCESS AREAS

For each of the Organizational-level Process Areas, the performers are as follows:

Supplier Agreement Management

Table 4. Specific practices and performers of Supplier Agreement Management (PM: Project Manager, PMS: Process Management Supporter, CEO: Chief Executive Officer, PMD: Production Management Director)

Specific Practice	Main performer	Assistant performer
SP 1.1 Determine Acquisition Type	PM	PMD
SP 1.2 Select Suppliers	PMD	
SP 1.3 Establish Supplier Agreements	CEO	PMD
SP 2.1 Execute the Supplier Agreement	PMD	
SP 2.2 Accept the Acquired Product	PM	PMD
SP 2.3 Ensure Transition of Products	PMD	

Organizational Process Focus

Table 5. Specific practices and performers of Organizational Process Focus (P.M: Process Manager, DD: Development Director)

Specific Practice	Main performer	Assistant performer
SP 1.1 Establish Organizational Process Needs	CEO	P.M, DD
SP 1.2 Appraise the Organization's Processes	P.M	DD, PM
SP 1.3 Identify the Organization's Process Improvements	P.M	DD, PM
SP 2.1 Establish Process Action Plans	P.M	
SP 2.2 Implement Process Action Plans	P.M	
SP 3.1 Deploy Organizational Process Assets	P.M	
SP 3.2 Deploy Standard Processes	P.M	
SP 3.3 Monitor the Implementation	P.M	DD
SP 3.4 Incorporate Experiences into Organizational Process Assets	P.M	

Organizational Process Definition

Table 6. Specific practices and performers of Organizational Process Definition

Specific Practice	Main performer	Assistant performer
SP 1.1 Establish Standard Processes	P.M	
SP 1.2 Establish Lifecycle Model Descriptions	P.M	PM
SP 1.3 Establish Tailoring Criteria and Guidelines	P.M	PM
SP 1.4 Establish the Organization's Measurement Repository	P.M	PMS
SP 1.5 Establish the Organization's Process Asset Library	P.M	PMS
SP 1.6 Establish Work Environment Standards	P.M	
SP 1.7 Establish Rules and Guidelines for Teams	P.M	PMS

Decision Analysis and Resolution

Table 7. Specific practices and performers of Decision Analysis and Resolution

Specific Practice	Main performer	Assistant performer
SP 1.1 Establish Guidelines for Decision Analysis	P.M	
SP 1.2 Establish Evaluation Criteria	according to object	
SP 1.3 Identify Alternative Solutions	according to object	
SP 1.4 Select Evaluation Methods	according to object	

SP 1.5 Evaluate Alternative Solutions	according to object
SP 1.6 Select Solutions	CEO

Organizational Process Performance

Table 8. Specific practices and performers of Organizational Process Performance

Specific Practice	Main performer	Assistant performer
SP 1.1 Establish Quality and Process Performance Objectives	CEO	P.M
SP 1.2 Select Processes	P.M	
SP 1.3 Establish Process Performance Measures	P.M	PMS
SP 1.4 Analyse Process Performance and Establish Process Performance Baselines	P.M	
SP 1.5 Establish Process Performance Models	P.M	

Causal Analysis and Resolution

Table 9. Specific practices and performers of Causal Analysis and Resolution

Specific Practice	Main performer	Assistant performer
SP 1.1 Select Outcomes for Analysis	P.M	
SP 1.2 Analyze Causes	P.M	PMS
SP 1.3 Implement Action Proposals	P.M	PMS
SP 2.1 Evaluate the Effect of Implemented Actions	P.M	PMS
SP 2.2 Record Causal Analysis Data	P.M	PMS

Quantitative Project Management

Table 10. Specific practices and performers of Quantitative Project Management

Specific Practice	Main performer	Assistant performer
SP 1.1 Establish the Project's Objectives	PM	DD, P.M
SP 1.2 Compose the Defined Process	PMS	P.M
SP 1.3 Select Subprocesses and Attributes	PMS	P.M
SP 1.4 Select Measures and Analytic Techniques	PMS	P.M
SP 2.1 Monitor the Performance of Selected Subprocesses	PMS	P.M
SP 2.2 Manage Project Performance	PM	PMS, P.M
SP 2.3 Perform Root Cause Analysis	P.M	PM, PMS

Organizational Performance Management

Table 11. Specific practices and performers of Organizational Performance Management

Specific Practice	Main performer	Assistant performer
SP 1.1 Maintain Business Objectives	CEO	DD, P.M
SP 1.2 Analyse Process Performance Data	P.M	
SP 1.3 Identify Potential Areas for Improvement	P.M	
SP 2.1 Elicit Suggested Improvements	P.M	
SP 2.2 Analyse Suggested Improvements	P.M	
SP 2.3 Validate Improvements	P.M	PMS
SP 2.4 Select and Implement Improvements for Deployment	P.M	PMS
SP 3.1 Plan the Deployment	P.M	
SP 3.2 Manage the Deployment	P.M	
SP 3.3 Evaluate Improvement Effects	P.M	PMS

Measurement and Analysis

Table 12. Specific practices and performers of Measurement and Analysis

Specific Practice	Main performer	Assistant performer
SP 1.1 Establish Measurement Objectives	P.M	PMS
SP 1.2 Specify Measures	P.M	PMS
SP 1.3 Specify Data Collection and Storage Procedures	P.M	PMS
SP 1.4 Specify Analysis Procedures	P.M	PMS
SP 2.1 Obtain Measurement Data	P.M	PMS
SP 2.2 Analyse Measurement Data	P.M	PMS
SP 2.3 Store Data and Results	P.M	
SP 2.4 Communicate Results	P.M	PM, PMS

3.5 Contribution Rates to implementing the Organizational-level Process Areas

Table 13 shows how much members in the organization contribute to implementing the Organizational-level Process Areas.

Table 13. Contribution Rates to implementing the Organizational-level Process Areas

	P.M	PMS	PM	CEO	DD	PMD
M	44	4	4	2	0	3
A	7	20	5	1	6	3
S	47.5	14	6.5	2.5	3	4.5

Where M and R are the number of Specific Practices a member in the organization takes on respectively main performer and assistant performer,

and S = M + 0.5 × A.

Fig. 3 is a chart presentation of Table 13.

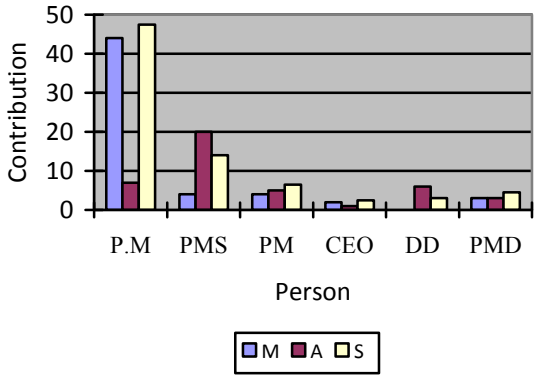

Figure 3 Contribution Rates.

As shown in Table 13 or Figure 3, Process Manager takes charge of most activities in the Organizational-level Process Areas.

4- CASE STUDY RESULTS

We selected 4 small size software organizations in DPRK and applied our program to them for process improvement.

These organizations usually had 28~43 developers, 4~5 management personnel, 5~10 agile projects. So these organizations could be viewed as typical small size software organizations in the domestic software industry.

Organization A had used informal software development methods and had initiated process improvement based on ISO 9001 since 2005, and eventually acquired the domestic Quality Management System certificate complying with ISO. But likely as most small size software organizations, it suffered from overload of the ISO Quality Management System, and individuals in project teams also followed development practices similar to agile methodologies. Finally, the organization gave up the certificate and continued to base its software developments on informal development and management methods fitting with its own reality. Organization B and C had aimed at Quality Management System certificates complying with ISO and had been establishing their organizational-level processes, but they had given up their desired certificates halfway because they had fears of failing from organization A's experience. Organization D, founded in 2012, had been making their effort to establish their own software development system.

These organizations were managing all their projects successfully using agile methods, and were orienting to CMMI compliance to establish their disciplined

processes and to push their reputations up. On their ways to these orientations there were many obstacles in defining, implementing the organizational-level processes. In advocacy of the advantage of agile methodology, most project teams were seeking only for efficient progress of their own project, and even some teams regarded organizational-level processes as interfering and overlording under their viewpoint of “The end justifies the means”. And senior managements had not any organization-wide systems where projects in organizations could be managed, monitored and controlled effectively.

Table 14. Summary of the case organizations.

Organization	Field	Projects	Developers per project (average)	Project durations (months)
Org. A	Security	3	5	10
	Web application	5	4	7
	artificial intelligence	2	4	12
Org. B	Multimedia communication	2	6	8
	Android application	4	4	6
Org. C	Embedding software	2	7	9
	network communication	3	6	8
Org. D	network security	2	4	10
	Web application	5	6	8

Table 15. Common process areas across the 4 organizations, identified as to de improved up.

Category	Process Areas	Contexts
Newly established	Supplier Agreement Management	Hardly happening
	Organizational Process Focus	Informal
	Organizational Process Definition	Informal
	Decision Analysis and Resolution	Informal
	Organizational Performance Management	Never
	Causal Analysis and Resolution	Never
Complemented	Quantitative Project Management	ill-defined objectives and criterion, undocumented measurement and statistical management methods
	Organizational Process Performance	Only collection of some data
	Measurement and Analysis	Not pre-defined objectives, non-consistent measurements

We recommended these organizations to apply our proposal, which had been helped, observed, and analysed by us for about 2 years.

First of all, a Process Manager and Process Management Supporters were appointed in each organization. Next, the Process Manager did analyse current processes of his/her own organization and did categorize them into the ones newly established and the ones complemented with our guides.

In the 4 organizations, their process improvement efforts were initiated in 2013 and lasted for about 3 years. Formal CMMI appraisals were conducted in 2016, results of which are shown in Table 16.

Table 16. Appraisal results of the Organizational-level Process Areas in the 4 organizations
(1: performed, 2: managed, 3: defined)

Category	Process Areas	Org. A	Org. B	Org. C	Org. D
Newly established	Supplier Agreement Management	3	3	3	3
	Organizational Process Focus	3	3	3	3
	Organizational Process Definition	3	3	3	3
	Decision Analysis and Resolution	3	3	3	3
	Organizational Performance Management	3	3	3	3
	Causal Analysis and Resolution	2	3	2	1
Complemented	Quantitative Project Management	3	3	3	3
	Organizational Process Performance	3	2	2	1
	Measurement and Analysis	3	3	3	3

As seen in the above Table, all the 4 organizations have been appraised at CMMI level 4.

5- CONCLUSIONS

We presented an efficient organizational structure and roles for reaching higher level of CMMI in small size software organizations consisting of Scrum teams.

Roles of Process Manager and Process Management Supporters responsible for implementing the organizational-level processes were specified. These roles might be taken charge of by QA director and testers, without hiring new persons.

We applied our proposal to 4 small size software organizations which had various experiences and histories of process management. The case studies

show that small software organizations can reach CMMI level 4 without hiring new process improvement specialists and damaging their agility.

REFERENCES

[1] K. Łukasiewicz, and J. Miler, “Improving agility and discipline of software development with the Scrum and CMMI”, IET software, vol. 6, no. 5, pp. 416-422, 2012.

[2] J. A. H. Alegrıa, and M. C. Bastarrica, “Implementing CMMI using a combination of agile methods”, CLEI Electronic Journal, vol. 9, no. 1, pp. 1-15, 2006.

[3] M. Fritzsche, and P. Keil, “Agile methods and CMMI: compatibility or conflict?” e-Informatica Software Engineering Journal, vol. 1, no. 1, pp. 10-23, 2007.

[4] F. G. Wilkie, D. McFall, and F. McCaffery, “An evaluation of CMMI process areas for small-to medium-sized software development organisations”, Software Process: Improvement and Practice, vol. 10, no. 2, pp. 189-201, 2005.

[5] Paul E. McMahon, “Integrating CMMI and Agile Development Case Studies and Proven Techniques for Faster Performance Improvement”, Addison-Wesley, pp. 57-134, 2010.

[6] A. Cockburn, ‘‘Crystal clear: a human-powered methodology for small teams”, Pearson Education, 2004.

[7] M. J. Boyer and H. Mili, “Agile business rule development”, In Agile Business Rule Development, Springer Berlin Heidelberg, pp. 49-71, 2011.

[8] L. Rising, and N. S. Janoff, “The Scrum software development process for small teams”, IEEE software, vol. 17, no. 4, pp.26-32, 2000.

[9] M. Kajko-Mattsson, G. A. Lewis, D. Siracusa, T. Nelson, N. Chapin, M. Heydt, and H. Snee, “Long-term life cycle impact of agile methodologies”, 22nd IEEE International Conference on Software Maintenance, ICSM'06, pp. 422-425, Sep, 2006.

[10] M. Poppendieck, and T. Poppendieck, “Lean Software Development – An Agile Toolkit for Software Development Managers”, Addison-Wesley, Boston, 2003.

[11] K. Beck, ‘‘Extreme programming explained: embrace change”, addison-wesley professional, 2000.

[12] K. N. Rao, G. K. Naidu, and P. Chakka, “A study of the Agile software development methods, applicability and implications in industry”, International Journal of Software Engineering and its applications, vol. 5, no. 2, pp. 35-45, 2011.

[13] G. Miller, “Want a better software development process'? Complement it”, IT professional, vol. 5, no. 5, pp.49-51, 2003.

[14] CMMI Product Team, “CMMI for Development, version 1.2”, Available at http://www.sei.cmu.edu/library/abstracts/reports/06tr008.cfm, Software Engineering Institute. 2006

[15] CMMI Product Team, “CMMI for Development, Version 1.3”, Available at http://www.sei.cmu.edu/library/abstracts/reports/10tr033.cfm, Software Engineering Institute. 2010.

[16] M. C. Paulk. “Using the Software CMM with Good Judgment”, ASQ Software Quality Professional, vol. 1, no. 3, pp. 1-13, 1999.

[17] R. Van Solingen, “Measuring the ROI of software process improvement”, IEEE software, vol. 21, no. 3, pp. 32-38, 2004.

[18] M. Doernhoefer, “Surfing the net for software engineering notes”, SIGSOFT Softw. Eng. Notes, vol. 31, no. 2, pp. 17-25, 2006.

[19] J. Herbsleb, D. Zubrow, D. Goldenson, W. Hayes, and M. Paulk, “Software quality and the capability maturity model”, Communications of the ACM, vol. 40, no. 6, pp. 30-40, 1997.

[20] J. J. Jiang, G. Klein, H. G. Hwang, J. Huang, and S.Y. Hung, “An exploration of the relationship between software development process maturity and project performance”, Information & Management, vol. 41, no. 3, pp. 279-288, 2004.

[21] CMMI Product Team, “Capability Maturity Model Integration (CMMI), Version 1.1, CMMI for Systems Engineering, Software Engineering, Integrated Product and Process Development, and Supplier Sourcing (CMMI-SE/SW/IPPD/SS, v1.1)”, Software Engineering Institute, 2002.

[22] C. Keith, “Agile game development with Scrum”, Addison-Wesley Professional, pp. 13-124, June, 2010.