Design and Evaluation of a CMMI conformant Light-Weight Project Management Approach

[1]Chayakorn Piyabunditkul, [2]Horst Lichter,
[3]Toni Anwar, [4]Apinporn Methawachananont,
[5]Chumphol Krootkaew, [6]Tisanai Krisanathamakul
[1,4,5,6]TGGS-NECTEC/NSTDA, chayakorn@nectec.or.th
[2]RWTH Aachen University, lichter@swc.rwth-aachen.de
[3]Faculty of Computer Science and Information Systems,
Universiti Teknologi Malaysia (UTM), Malaysia, tonianwar@utm.my

Abstract

CMMI is one of the well-known and accepted maturity models that many software organizations have implemented for its quality processes which are expected to bring a good quality for their software products. However, traditional software process models become too heavy-weight to be deployed. The aim of this research is to design the Light-Weight Project Management (LWPM) approach to implement CMMI by mapping between CMMI goals and Agile-Scrum based on defined artifacts and to indicate the differences in applying LWPM and the traditional Waterfall model. Our approach focuses on the Project Management category which composes Project Planning (PP), Project Monitoring and Control (PMC) and Integrated Project Management (IPM). In order to compare both models we collected relevant data by using questionnaire and also the dedicated tool SPIALS (Software Process Improvement Adaptive Learning System).

Keywords*: CMMI (Capability Maturity Model Integration), Standard CMMI Appraisal Method for Process Improvement (SCAMPI), Light-Weight Project Management (LWPM), Very Small Enterprises/Small Medium Enterprises (VSEs/SMEs), CMMIbyScrum Framework (CMMISF)*

1. Introduction

Many software organizations invest a lot of resources and budget to reach the target of high-quality by implementing heavy-weight organization plans and processes like CMMI (Capability Maturity Model Integration). On the other hand, CMMIbyScrum Framework (CMMISF) is an alternative approach to accelerate the transition process by using the Light-Weight Project Management (LWPM) organization framework. This intends to achieve better performance with less effort. However, the LWPM implementation should be done with enough quality of processes and needed products to be qualified by the Standard CMMI Appraisal Method for Process Improvement (SCAMPI).

According to the Software Engineering Institute (SEI)'s definition, SCAMPI is designed to provide benchmark-quality ratings relative to Capability Maturity Model Integration (CMMI) models. SCAMPI, as a benchmarking appraisal method, relies on an aggregation of information collected via defined objective evidence. This collection of data forms the basis of ratings and other appraisal results by appraisal teams which are obligated to seek and consider objective evidence of multiple types in determining practice implementation and goal satisfaction [1].

The SCAMPI method is data-oriented because decisions on practice implementation and goal rating are made based on the aggregate of objective evidences (artifacts and affirmations) available to the appraisal team. These statements are typically collected using interviews, demonstrations, questionnaires, or other means. SCAMPI defines processes and activities for each of the three appraisal phases. Those activities are planned and prepared for the appraisal phase, the conduct appraisal phase, and the report results phase. In this research, the SCAMPI method is presented via a LWPM software process improvement self-assessment tool called “SPIALS (Software Process Improvement Adaptive Learning System)”.

This paper presents a design and evaluation mechanism of CMMI conformance as LWPM tool focusing on project management areas for VSEs/SMEs to increase project performance. The remaining is organized as following; Section 2 presents the background overview of Project Management

category in CMMI, SCAMPI and Agile-Scrum. Section 3 focuses on CMMISF conceptual design, and section 4 presents the user interface of SPIALS tool. Section 5 shows the investigations from experiment experience, and section 6 discusses some conclusions and future work.

2. Related Work and Background

In this research, the CMMI-SCAMPI scope is the Project Management category which contains PP, PMC and IPM in capability level 3 (CL3). The main reason for selecting the project management category is the importance of its activities to the success of the project in overall perspective. The project management activities that should be considered but not limited to, are establishing and maintaining the project plan and commitments, stakeholders involvement to an integrated and defined process, monitor the progress against the plan, measure performance deviation from the plan and issue the corrective action, develop process tailoring from the organization set of standard processes according to project work environment, coordinate and collaborate among relevant stakeholders to form and sustain the integrating team within the project processes [4].

Mike Cohn, Mountain Goat software, who has experiences with Fortune 500 companies and also small startups for over fifteen years in Scrum and agile projects since 1995, gives a definition of Agile-Scrum as an agile process for software development, projects progress via a series of iterations called sprints. Each sprint is typically 2-4 weeks long. While numbers of agile approaches can be used for managing any project, Scrum is ideally suited for projects with rapidly changing or highly emerged requirements [4]. With this approach, VSEs/SMEs will be able to satisfy the customer through continuous delivery of working products in software development process.

The context of the synergy between CMMI and Agile methods shows the optimistic of the deployment of light-weight Software Process Improvement (SPI) procedure, XP and Scrum for example which possess the strength of engineering area, to achieve the solid-less defect output product. To achieve the standard according to SCAMPI ML3 framework, the additional process area adoptions are needed especially in process management and support categories [2]. However, those gaps can be fulfilled with less significant efforts. So it is recommended to be a good starting point for the small unit team [3].

In this paper, we reference CMMI based on version 1.2 (CMMI-DEV version 1.2-August 2006), SCAMPI on version 1.2-August 2006 and Agile-Scrum, that Sutherland and Schwaber jointly presented in a paper describing the Scrum method in 1995 [10].

2.1. Project Management Category in CMMI

CMMI, a process improvement framework, is a collection of performance targets and activity recommendation as best practices [12]. CMMI for Development (CMMI-DEV) version 1.2 consists of 22 process areas of which each area has its own capability or maturity level that provide organizations the essential elements of effective processes. This paper focuses on project management category of CMMI-DEV which is a model designed for software development processes and it is a reference model that covers the development and maintenance activities applied to both products and services. The chosen project management category includes Project Planning (PP), Project Monitoring and Control (PMC) and Integrated Project Management (IPM) [4].

2.2. The Standard CMMI Appraisal Method for Process Improvement (SCAMPI)

SCAMPI is designed to provide benchmark quality ratings based on the CMMI model. It is applicable to a wide range of appraisal usage modes, including both internal process improvement and external capability determinations.

As a benchmarking appraisal method, SCAMPI relies on an aggregation of information collected via defined types of objective evidence. The objective evidence feeds an “information-processing engine”, whose parts are made up with series of data transformations and the characterizations of practice implementation as gap or compliance, to be the preliminary findings. These findings will be validated by the organizational unit before they become the final findings. The critical concept is that

these transformations are applied to data reflecting the enacted processes in the organizational unit and the CMMI model, and this collection of data forms the basis for ratings and other appraisal results [1].

2.3. Software Development Life Cycle (SDLC)

The software development process, also known as a software development life cycle (SDLC), is a structure imposed on the development of a software product. Similar terms include software life cycle and software process. It is often considered as a subset of systems development life cycle. There are several software process models. They describe approaches to perform variety of tasks or activities that take place during the development process. This research focuses on two types of such models; the first one represents heavy-weight traditional process models referred as “Waterfall model” and second one represents light-weight models called “CMMISF”.

2.3.1 Waterfall Based Process Model

On the one hand, the traditional waterfall process model has been the mainstay for software developers for many years. For software products that do not change very much once they are specified, the waterfall model is still viable. However, the traditional waterfall model is no longer appropriate [5]. On the other hand, this classic process model is often represented as a simple prescriptive waterfall software phase model, which has been perhaps most useful in helping to structure, staff, and manage large software development projects in complex organizational settings, which was one of the primary purposes [8]. Generally, the waterfall model describes a linear and sequential development method with distinct goals for each phase of development which is requirement gathering, analysis, design, coding, testing, implementation, post implementation [9]. The artifacts of the Waterfall model are manifold, such as project plan, work breakdown structure, progress report. The roles are grouped by its functionalities; for instance management, operation, quality and customer/user group.

2.3.2 Agile-Scrum

Scrum is an iterative, incremental framework for project management often applied in agile software development. Scrum has not only reinforced the interest in software project management, but also challenged the conventional ideas about such management. Scrum focuses on projects where it is difficult to plan ahead with mechanisms for empirical process control, applying feedback loops as the core element of product development compared to traditional command-and-control oriented management. It represents a radically new approach for planning and managing software projects, bringing decision-making authority to the level of operation properties and certainties. Scrum reduces defects and makes the development process more efficient, as well as reducing long-term maintenance costs [6] [11]. Generally, Scrum defines three main phases; Pregame, Game and Postgame. The Pregame phase has a planning process to develop a comprehensive backlog list which contains all requirements, estimations and high-level system architecture design of the project plan. The Game is a development phase where development id done in iterative cycles. An iterative cycle is called “Sprint”, which contains a set of development activities conducted over a pre-defined period such as team meeting, distribute, review, and adjustment product to conform a standard. Finally, in the Postgame phase the software is prepared for general release, integration, and system test. Furthermore, user documents and training materials are created. It also includes Sprint review and Sprint Retrospective for continuous process improvement. Scrum defines three main artifacts; Product backlog, Sprint backlog, and Burn down chart. The roles in Scrum are Scrum master, project owner and team. [7]

3. CMMISF Conceptual Design

The CMMISF proposed in this paper is a conceptual framework for an effective practice. The CMMISF is based on CMMI and composes 3 components which are part 1: Base model, part 2: Methodology and part 3: Evaluation. The deploying method is Scrum and the applied evaluation criteria are based on SCAMPI. Scrum as deployed here, includes 4 practices; Sprint Planning Meeting (SPM), Dairy Scrum Meeting (DSM), Sprint Review Meeting (SRM), and Sprint Retrospective (SR).

While the Scrum artifacts are Product Backlog (PB), Sprint Backlog (SB), Product Burndown Chart (PBC) and Sprint Burndown Chart (SBC). SCAMPI includes either Maturity level or Capability level.

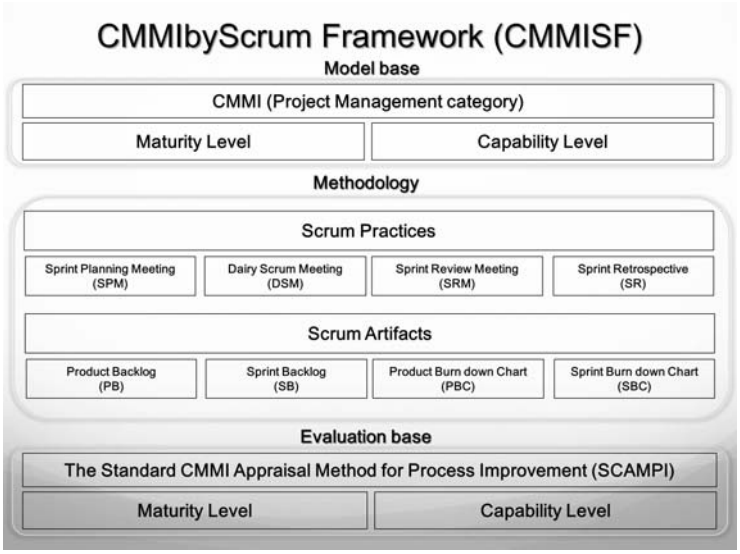

Figure 1. CMMISF framework based on SCAMPI

Our framework was designed to show how a light-weight approach can reduce the resources needed to complete the same software process via traditional approach. With a self-assessment tool, practitioners can analyze their processes for suggestion to maintain the quality within the boundary of an accepted standard.

To illustrate our idea, the Project Management category of the based model CMMI was selected. Project Planning (PP), Project Monitoring and Control (PMC) and Integrated Project Management (IPM) were processes we included in our experiment while the Requirement Management (REQM), Supplier Agreement Management (SAM) and Quantitative Project Management (QPM) were left off for simplicity reasons.

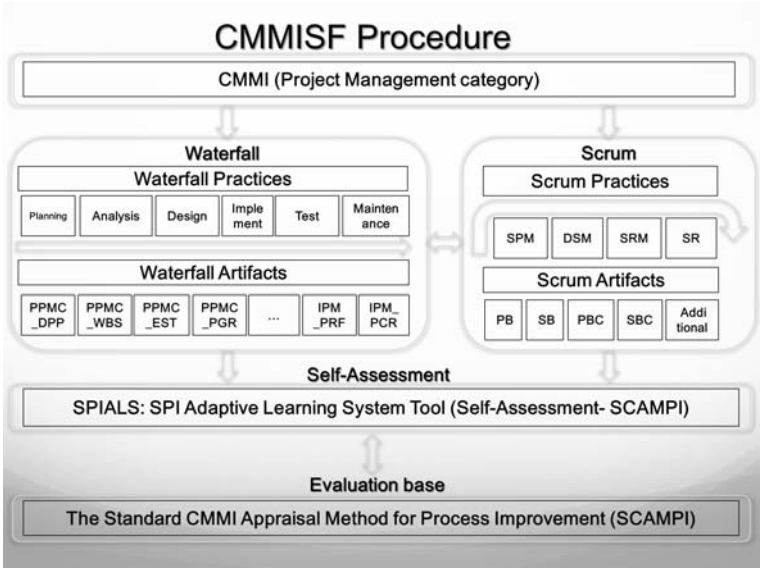

Figure 2. The CMMISF Procedure

Figure 2 presents the experiment procedures: the traditional waterfall model and Scrum were applied to perform software processes in the CMMI Project Management category as stated above. The

measurement results were recorded for analyses and comparisons. The underlying analysis concept is depicted in figure 3. We used artifacts from both process models in our comparison processes. The initial results that we obtained are published in [3]. The structure of CMMI (left side) can be mapped to the proposed Scrum approach (right side). The details of the experiment results are explained in section 5.

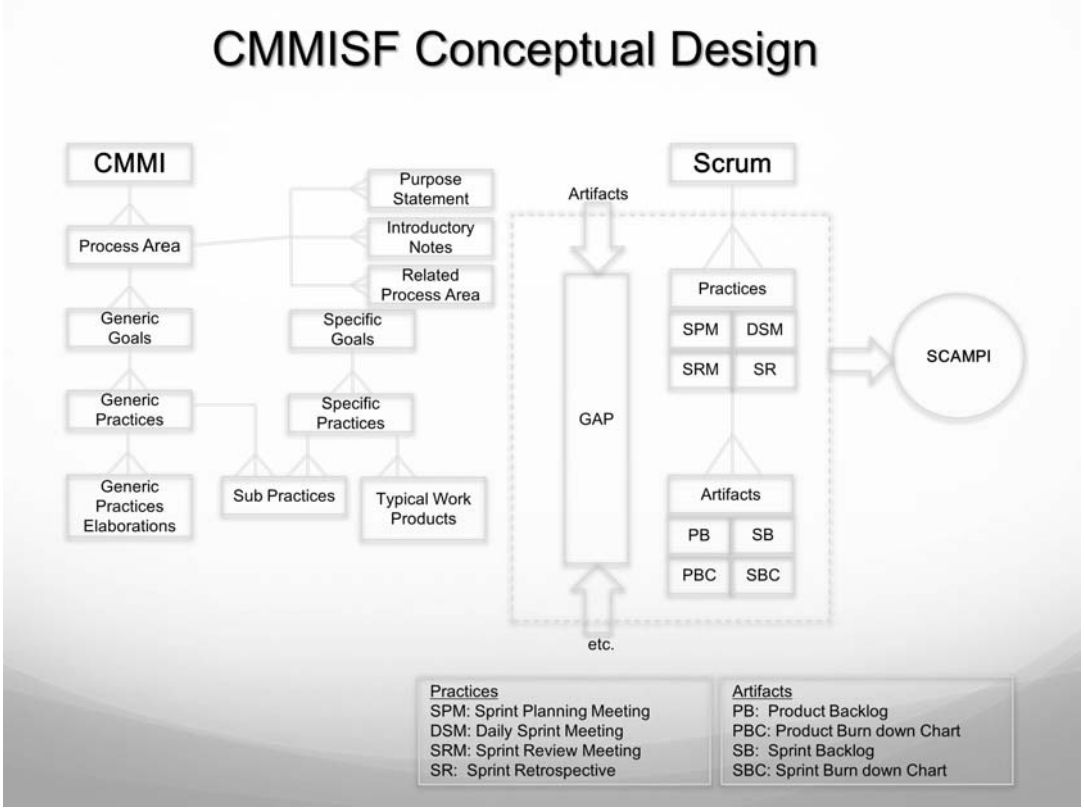

Figure 3. The CMMISF Conceptual Design

Our self-assessment tool SPIALS can be used to collect the process evidences from practitioners from both process models to compare them based on SCAMPI. The functionality of SPIALS is described in detail in the following section.

4. The SPI Adaptive Learning Tool (SPIALS)

The CMMISF proposed here is a framework for process activities that conforms to the best practices adopted by Agile-Scrum methods instead of using a traditional process model such as the Waterfall model. However, it intends to keep organization process quality which means the implementation will conform to the accepted standard SCAMPI.

In the following we present how to apply the tool SPIALS to perform a self-assessment guided by the SPIALS self-assessment model SPIALS also produces a gap report analysis and SPI proposal report which can be used to start a process improvement program. Figure 4 shows an excerpt of the user interface offered to enter organization information such as general information, organization participant, project and type of SDLC. In addition, the Software Engineering Process Group (SEPG) has to manage organization artifacts for project participants as shown below.

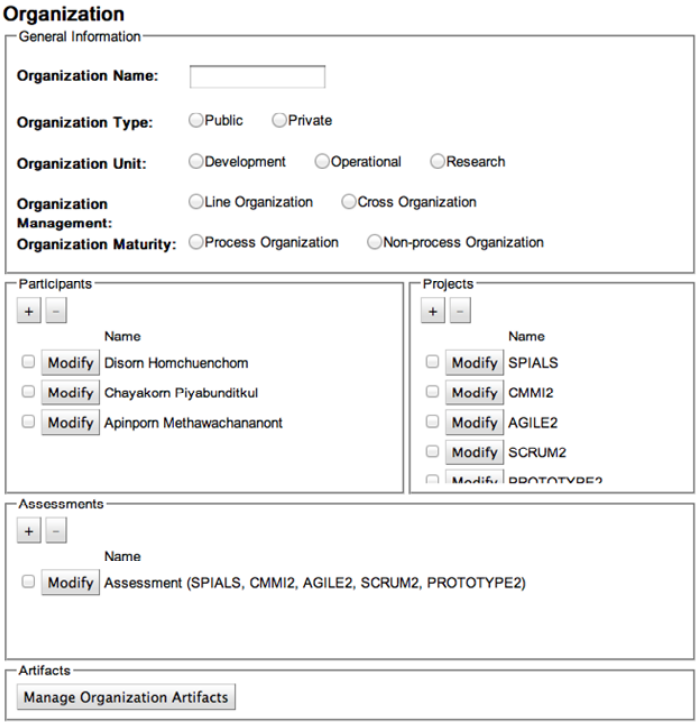

Figure 4. The User Interface of "SPIALS" on Organization Information

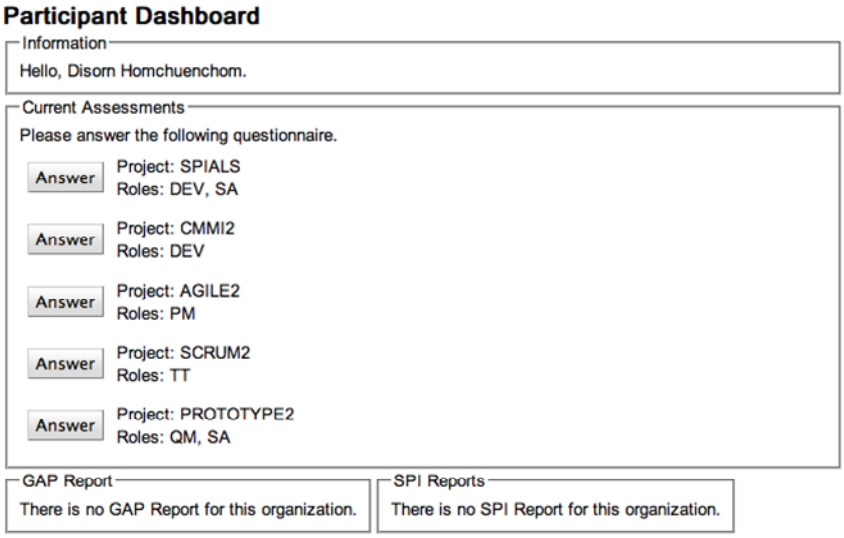

Figure 5. The User Interface of "SPIALS" on Participant Dashboard

Figure 5 shows another example of the user interface for each participant if someone has multi-roles in the same project or different projects.

Finally, according to figure 6, SPIALS produces two reports. First, the gap analysis report contains an overall organization summary, gap analysis result, strength and weakness of organization and its project. Second, the SPI proposal report describes the details for a continuous process improvement. It explains how to fulfill organization weaknesses and shows the values of measurement comparison in terms of effort and User Acceptance Test (UAT) defects.

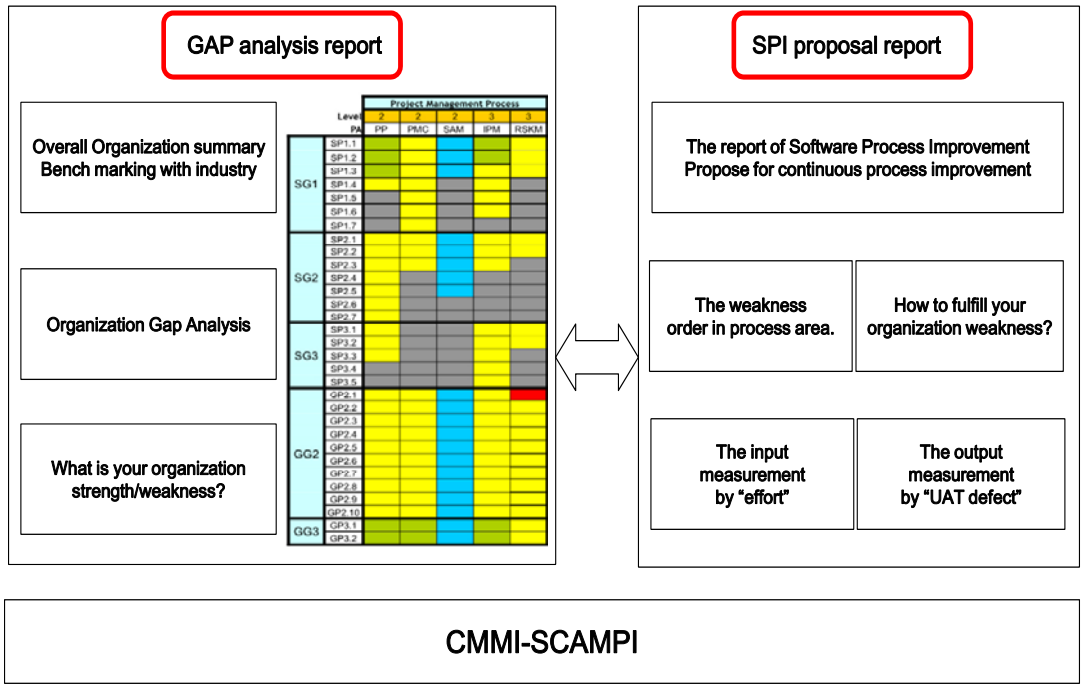

Figure 6. GUI of SPIALS for Gap Analysis and SPI Proposal Report

The relevant artifacts from those topics are mapped according to the mapping mechanism presented in figures 2 and 3. SPIALS has an automated process to create a result such as the Gap Analysis Report that describes the gap between organizational SCAMPI targets and the classified current practices. The gap analysis also shows the comparison results with the industrial bench-marking systems. Beside the gap analysis reports, SPIALS also displays the Software Process Improvement report (SPI proposal) explaining how to fulfill those gaps.

5. Results of Investigation from Experimental Experiences

We have performed an experiment in an organization which implements the waterfall model and CMMISF to fulfill SCAMPI, based on Project Planning (PP), Project Monitoring and Control (PMC) and Integrated Project Management (IPM) process area. Active participants in this study used data from the NECTEC-CMMI project. The data are indicators for the project effort and the defect rate obtained in the UAT is used to compare the differences in the use of resources in similar projects. After that, we proceed to evaluate the implementation of the standards by the output of the SCAMPI project, both in comparison with the standards of the SCAMPI. The operation was carried out under the SCAMPI ML3.

The experiment selects artifact templates related to process areas mentioned above. Altogether, the effort to create artifacts based on 9 templates has been analyzed (see Table 1).

Table 1. Effort Distribution in a Project applying by Waterfall model

	Effort (**man-hour**)	**%**
Waterfall model Summary Effort	**374**	**100.00**
TP_PPMC_DPP (Template for PDP Management Plan)	156	41.71
TP_PPMC_WBS (Template for PDP Work Breakdown Structure)	78	20.86
TP_PPMC_EST (Template for PDP Estimation sheet)	36	9.63

TP_PPMC_PGR (Template for PDP Progress Report)	33	8.82
TP_PPMC_QAP (Template for QA Plan)	15	4.01
TP_PPMC_ISL (Template for Issue Log)	18	4.81
TP_IPM_SKL (Template for Skill Matrix and Competency Evaluation)	18	4.81
TP_IPM_PRF (Template for PDP Request Form)	6	1.60
TP_IPM_PCR (Template for PDP Closure Report)	14	3.74
Additional Template	0	0.00

The CMMISF document comprises 3 templates (PBC and SBC are concluded in TP_Burndown Chart). Table 2 shows the effort distribution. It also indicates the additional template with an effort of 12 man-hours or 3.39% for fulfilling the SCAMPI requirement.

Table 2. Effort Distribution in a Project applying CMMISF

	Initial (man-hour)	**Additional (man-hour)**	**Total (man-hour)**	**%**
CMMISF Summary Effort	**223**	**131**	**354**	**100.00**
TP_Product Backlog	122	80	202	57.06
TP_Sprint Backlog	56	24	80	22.60
TP_Burndown Chart	45	15	60	16.95
Additional Template	0	12	12	3.39

Figure 7 shows the effort comparison between Waterfall model, and CMMISF. As table 2 shows the CMMISF project spends 20 units less than in the Waterfall model project which is including total addition 131 units from traditional Agile-Scrum plus 12 units from new additional templates to fulfill SCAMPI in CMMISF, the performance of CMMISF is better than Waterfall model approximately by 5.35 %.

The experimental application of CMMISF that results in reduced efforts and lower the defect rates at UAT is desirable, although applications with CMMISF must be more reliable documents as defined in SCAMPI ML3. Yet, when combined efforts of the output are increased, the overall result is still at a satisfactory level, with the better performance of the Key Performance Indicators (KPI) as mentioned above. The figure is as follow:

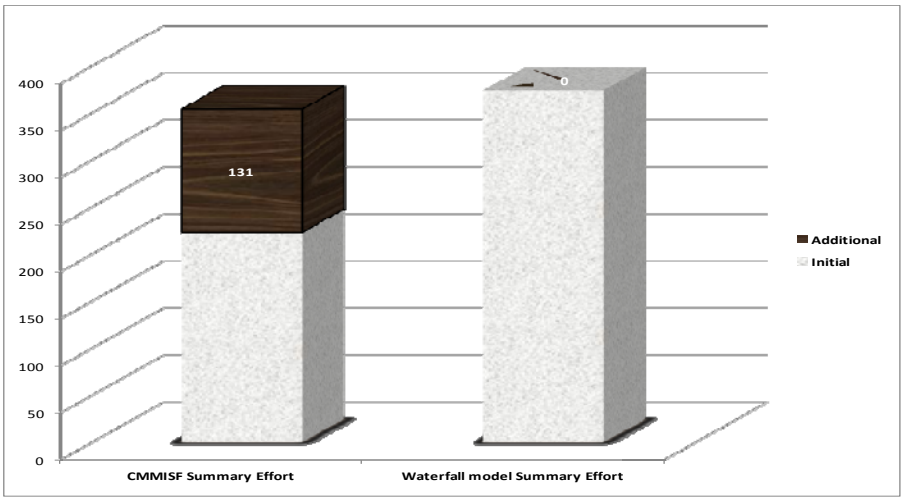

Figure 7. The comparative results of total efforts (unit: man-hour)

Table 3 summarizes the UAT (User Acceptance Test) defects found in the project using Waterfall model. The requirement & analysis phase and the rapid prototype & design phase are the phases that have the highest number of UAT defects. Totally, number of defects is 21 based on 62 performed UAT test cases.

Table 3. Distribution of UAT defects in the Project applying the Waterfall model

NECTEC-Waterfall model Template	Project Planning	Requirement & Analysis	Rapid Prototyping & Design	Implementation	Integration	Deploy	Maintenance	Closure	Total
UAT (Passed)	3	11	10	0	4	12	0	1	41
UAT defect (Not Passed)	0	7	7	0	5	2	0	0	21

Table 4 shows the number of defects found in project implemented with CMMISF (12 defects based on 62 performed UAT test cases).

Table 4. Distribution of UAT defects found in the Project applying CMMISF

NECTEC-CMMISF Template	The pregame phase	The game phase (development phase) - 24 sprints	The postgame phase (closure phase)	Total
UAT (Passed)	0	40	10	50
UAT defect (Not Passed)	0	10	2	12

In comparison, the number of defects in the CMMISF project is less than half of the ones applying the waterfall model. The effort needed in the CMMISF project is only little less than in the waterfall based project. However, to confirm this first experiment result, we plan to extend the investigation to get a sound basis for our analysis.

6. Conclusions and Future Work

The comparison of the application of a heavy-weight traditional like Waterfall model and the light-weight CMMISF based on SCAMPI, shows that CMMISF is a good starting point for VSEs/SMEs with less effort and less UAT defects. However, there are some constrains from this experiment, for instance the considered process areas are only in project planning, project monitoring and control, and integrated project management area. Also the numbers of sampling are limited only from the internal control environment with the SCAMPI C characteristics defined via SPIALS by our researchers' team. In regards to the SPIALS tool, we use artifacts to represent the success of activities/practices. And finally, the self-assessment by using a questionnaire method is based on trusts. The results presented in this paper depend on the researcher team process improvement's spirit.

In the future, we plan to increase the number of experiments and other defined categories to confirm the result of our first research experiment. On the other hand, the process area scope can be extended further beyond the project management group, maturity level 2 or 3 with a recommended additional artifact. We hope that our results and the presented approach will be beneficial for VSEs/SMEs to implement CMMISF with high performance.

7. References

[1] SCAMPI Upgrade Team, "Standard CMMI® Appraisal Method for Process Improvement (SCAMPISM) A, Version 1.3: Method Definition Document", CMU/SEI-2011-HB-001, March 2011.
[2] Chayakorn Piyabunditkul, Apinporn Methawachananont, Sompol Chaimongkhon, Boonchai Charoendouysil, "Accelerated Adoption CMMI by Agile Methodologies", TGGS-RWTH Aachen-NECTEC, In Proceeding(s) of the ProMAC Symposium 2009 (ProMAC2009), Bangkok, Thailand, October 28 – 30, 2009.
[3] Chayakorn Piyabunditkul, Nithipat Wongchingchai, Apinporn Methawachananont, "Step forward CMMI-Project Management by optimized Scrum", TGGS-RWTH Aachen-NECTEC, In Proceeding(s) of the 5th International Conference on Project Management (ProMAC 2010), Makuhari Messe, Japan, October, 13-15, 2010.
[4] CMMI Product Team, "CMMI for Development, Version 1.2", CMU/SEI-2006-TR-008, August 2011.
[5] Elaine L. May and Barbara A. Zimmer, "The Evolutionary Development Model for Software", Hewlett-Packard Journal, August 1996.
[6] Schwaber, Ken, "Agile Project Management with SCRUM", Microsoft Press. ISBN 978-0-735-61993-7, February 2004.
[7] Schwaber, Ken, "SCRUM Development process", Proceedings of the 10th Annual ACM Conference on Object Oriented Programming Systems, Languages, and Applications (OOPSLA), Austin, Texas, 15-19 October 1995.
[8] Walt Scacchi, "Process Models in Software Engineering", Encyclopedia of Software Engineering, 2nd Edition, John Wiley and Sons, Inc, New York, December 2001.
[9] Government of Ontario IT Standard (GO-ITS), "Application Development Standard: Standards for SDLC", Queen's Printer for Ontario, 2007.
[10] Jeff Sutherland, "The Scrum Papers: Nut, Bolts, and Origins of an Agile Process", CTO and Worldwide Scrum Consulting Practice Manager, 2007.
[11] Md. Junaid Arafeen, Saugata Bose, "Improving Software Development Using Scrum Model by Analyzing Up and Down Movements on The Sprint Burn Down Chart: Proposition for Better Alternatives ", JDCTA: International Journal of Digital Content Technology and its Applications, Vol. 3, No. 3, pp. 109-115, 2009.
[12] Fatemeh Kafili Kasmaee, Ramin Nassiri, Gholamreza Latif Shabgahi, "Achieving CMMI Maturity Level 3 by Implementing FEAF Reference Models", IJACT: International Journal of Advancements in Computing Technology, Vol. 2, No. 4, pp. 115-122, 2010.