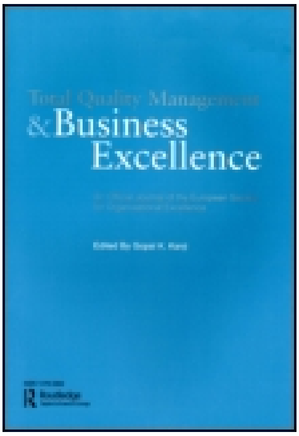

Total Quality Management & Business Excellence

Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/ctqm20

Understanding the effectiveness of Capability Maturity Model Integration by examining the knowledge management of software development processes

Shari S.C. Shang [a] & Shu-Fang Lin [b]

[a] Department of Management Information Systems, National Chengchi University, Taipei, Taiwan, ROC

[b] Department of Business Administration, Yuanpei University, Hsinchu, Taiwan

Published online: 04 Jun 2009.

To cite this article: Shari S.C. Shang & Shu-Fang Lin (2009) Understanding the effectiveness of Capability Maturity Model Integration by examining the knowledge management of software development processes, Total Quality Management & Business Excellence, 20:5, 509-521, DOI: 10.1080/14783360902863671

To link to this article: http://dx.doi.org/10.1080/14783360902863671

Total Quality Management
Vol. 20, No. 5, May 2009, 509–521

Routledge
Taylor & Francis Group

Understanding the effectiveness of Capability Maturity Model Integration by examining the knowledge management of software development processes

Shari S.C. Shang[a] and Shu-Fang Lin[b]*

[a]*Department of Management Information Systems, National Chengchi University, Taipei, Taiwan, ROC;* [b]*Department of Business Administration, Yuanpei University, Hsinchu, Taiwan*

Capability Maturity Model Integration (CMMI) is one of the well-known models that provide best practices for software quality improvement. Many articles praise the benefits of CMMI adoption, such as enhanced knowledge management of software development, improved software quality and increased efficiency of software development. However, these intangibles, especially those relating to knowledge management, have not been investigated yet. To build a deeper understanding of CMMI this paper intends to investigate the effectiveness of CMMI by examining the impact of quality management on the knowledge management of software development processes. A case study approach was employed to collect complete information of the impact of CMMI on software knowledge management from different aspects. The results reveal that the incongruence between managing the knowledge of software development and managing the process of software knowledge management has created blind points in the adoption of the CMMI programme. In addition to complying with the CMMI prescriptive guidelines CMMI-adoption organisations would need to integrate knowledge from both internal and external customers.

Keywords: CMMI; knowledge management; learning; quality improvement

Introduction

Capability Maturity Model Integration (CMMI) was designed as a comprehensive system development model for helping organisations have visible ongoing processes and have some measurements to inspect the process of software development quality (Chrissis et al., 2003; SEI, 2002). Despite the fact that CMMI has gradually gained its recognition as a symbol of quality assurance and has attracted more and more organisations to adopt CMMI, relatively few studies have provided empirical insights into the effectiveness of CMMI. Indeed, much of the existing literature is concerned with the relationship between CMMI and firm performance and therefore tends to ignore the effectiveness of the model itself (SBN, 2007; Staples et al., 2007). More specifically, with the development of the field of best practices there has been a massive outpouring of articles and books dealing with how to implement CMMI. These books take for granted the effect of CMMI on software quality improvement; we know comparatively little about the challenges an organisation may face in the CMMI implementation process. At the same time, the absence of instructions for adopting CMMI becomes an increasingly critical problem for managers (Glazer, 2003). This is particularly because implementing a

*Corresponding author. Email: lucylin@nccu.edu.tw

ISSN 1478-3363 print/ISSN 1478-3371 online

DOI: 10.1080/14783360902863671
http://www.informaworld.com

large-scale project such as CMMI provokes a change in the organisation. A deeper understanding of this issue provides key information for those developing, consulting and other enabling infrastructure and management teams to have more practical resources allocations and plan ahead to overcome barriers.

In an attempt to shed some light on the above-mentioned issues, this paper has two specific objectives: first, it examines the role of knowledge in CMMI. Second, the study uses a case study of an organisation that implemented CMMI: to examine the effectiveness of CMMI from the standpoint of knowledge management, and to consider the extent to which such processes can be enhanced. Therefore, the overall effect of this approach is to provide empirical evidence on the challenges of adopting CMMI and the reasons that hinder the effectiveness of CMMI.

This paper consists of four parts. After the introduction, the second section illustrates CMMI and its relationship with knowledge management. Next, we present the background information about the case study firm. In the fourth section, study results and findings are discussed. The paper concludes by presenting the managerial implications of our analysis, and seeks to draw out some of the key issues which confront the firm seeking to utilise the effectiveness of CMMI.

Capability Maturity Model Integrated (CMMI)

CMMI is a US government funded set of best practices mostly used for software development. CMMI was sponsored by the under-secretary of Defense for Acquisition, Technology, and Logistics, and it was designed to build on the ‘best practices’ of capability maturity models to develop an integrated framework for supporting different capability maturity models and other related products. CMMI can be considered as the system that enables an organisation or individuals to have benchmarks for continuous process improvement throughout the software development processes. In other words, CMMI focuses on software development processes, and software development capabilities, so as to enhance software development projects and management capabilities of software development with reduced development time, reduced development costs, and assured quality.

The purpose of CMMI is to develop an integrated framework for continuous process improvement. Indeed, CMMI provides a set of practices that help an organisation to establish its organisational maturity on the depth and width of software development processes, so that the organisation can prioritise the area for improvement. The organisation can also set up objectives and procedures based on the CMMI guidelines for software development process improvement.

There are two important concepts embedded in the CMMI framework: the continuity and the development stage. First, based on the continuity concept, CMMI classifies capabilities into six levels; each level contains objectives, methods for improvement and the capability of process improvement (SEI, 2002). Based on the development stage, on the other hand, CMMI divides software development processes into five maturity stages, in order to support and guide the software development processes. Within each maturity stage, there are certain activities that have to be done to reach the next higher maturity stage. The maturity stage also represents the evolutionary path of the quality improvement process.

Capabilities at a lower stage provide foundations for a higher stage. Each development stage or maturity level identifies an organisation’s software process capability (Paulk et al., 1993). To sum up, through continuous learning from best practices and establishing consistency in software process development, CMMI not only helps the firm to assess its capabilities but also ensures process efficiency, software process knowledge exploitation, and sustained competitive advantage.

The relationship between CMMI and knowledge management

In examining the effectiveness of CMMI from the knowledge management perspective, we need to be mindful of the relationship between knowledge management and CMMI. In the field of knowledge management, there are various definitions for knowledge and knowledge management. In this study, we adopt Pan and Scarbrough's (1999, p. 360) definition that 'knowledge management is the capacity or process within an organisation to maintain or improve organisational performance based on knowledge and experience.' Knowledge can also be divided into different types from different perspectives, but it is more often divided into tacit and explicit knowledge (Nonaka & Takeuchi, 1995). Tacit knowledge is unarticulated, intuitive and non-verbalised, while explicit knowledge can be expressed in some written or spoken form. Knowledge-based resources are embedded in multiple entities of originations such as routines, policies, systems, documents and employees. The objective of knowledge management is to explore and exploit knowledge-based resources to sustain organisations' competitive advantage. Similarly, the underlying foundation of CMMI is the quality management concept which shares a close philosophy with knowledge management in terms of creating and learning knowledge by fostering a stable and efficient working environment so as to advance organisational performance and competitive advantage (Levett & Guenov, 2000).

A number of scholars have, indeed, developed some conceptual or theoretical works that relate quality improvement to knowledge management. Mukherjee et al. (1998), for instance, have identified two types of learning: conceptual and operational learning in quality improvement activities. Conceptual learning in the quality improvement programme consists of 'assessing cause and effect relationships that govern experienced events, and designing an abstract concept to explain this experience, i.e. the acquisition of know-why' (Mukherjee et al., 1998, p. S38). Operational learning in the quality programme consists of implementing changes and observing the results of these changes. In short, it can be referred to as developing a skill of how to deal with experienced events, i.e. know-how (Mukherjee et al., 1998).

While conceptual learning provides a framework for organisations to integrate knowledge created by their members and enables employees to explore root causes of problems, operational learning facilitates the achievement of short-term quality goals. Mukherjee et al. (1998) note that conceptual learning and operational learning are closely related to quality improvement in two forms. On the one hand, the quality improvement concept requires plant managers to seek quick implementable solutions to problems identified. On the other hand, quality improvement also requires plant managers to use more sophisticated tools and frameworks to identify root causes of problems and to see new possibilities for shaping the future.

In addition to the examination of knowledge creation from the standpoint of learning literature, Linderman et al. (2004) provide another angle to examine knowledge creation in the quality improvement process by extending Nonaka and Takeuchi's (1995) theory. Linderman et al. (2004) reviewed and synthesised literature on the role of knowledge in quality management practices and proposed an integrated model to illustrate how socialisation, externalisation, combination and internalisation can be used as activities for knowledge creation in quality practices implementation. Socialisation in the quality programme occurs when employees interact with customers and share tacit knowledge. This also provides a basis for learning and to understand customer requirements, consequently helping establish a foundation for shared experience and mental models between customers and the organisation (Dean & Bowen, 1994). Externalisation in the quality programme refers to the process of capturing tacit knowledge of customers and explicating articulating it into product and process concepts (Dean & Bowen, 1994; Nonaka & Takeuchi, 1995). Combination in the quality improvement programme refers to the process by which the organisation combines the understanding of various customer data (i.e. gap analysis of perception and specification). Internalisation in the quality improvement programme refers to reinforcing the knowledge gained from the improvement efforts in the organisation (Flynn et al., 1994; Nonaka & Takeuchi, 1995).

It is widely accepted that organisational knowledge is embedded in products or processes, as firms create, store and analyse data and knowledge about market opportunities and technological development and use these to develop appropriate goods and procedures. Thus, we argue that it is appropriate to examine the effectiveness of CMMI from the perspective of knowledge management, as there is a strong link between knowledge management and CMMI. However, the effectiveness of CMMI has not adequately been investigated from such an aspect. Literature pertaining to CMMI has noted the effectiveness of CMMI adoption (PR News, 2007), but provides little understanding of the role of knowledge in CMMI adoption and of the barriers to successful implementation of CMMI (Glazer, 2003). Though Linderman et al. (2004) have identified the link between quality management practices and knowledge, we argue that as learning is the underlying philosophy of a quality improvement programme (Dada & Marcellus, 1994), it is necessary to examine the effectiveness of a quality improvement programme (such as CMMI) with both the knowledge creation process and learning. Thus, based on the literature review, we propose a conceptual model of knowledge management in a quality improvement programme (see Figure 1).

In this paper, we will use the proposed model (Figure 1) as a framework to examine the effectiveness of the CMMI model implemented for the quality improvement in software development processes. From Figure 1, it can be seen that socialisation, externalisation, combination and internalisation are the essential activities for knowledge creation as

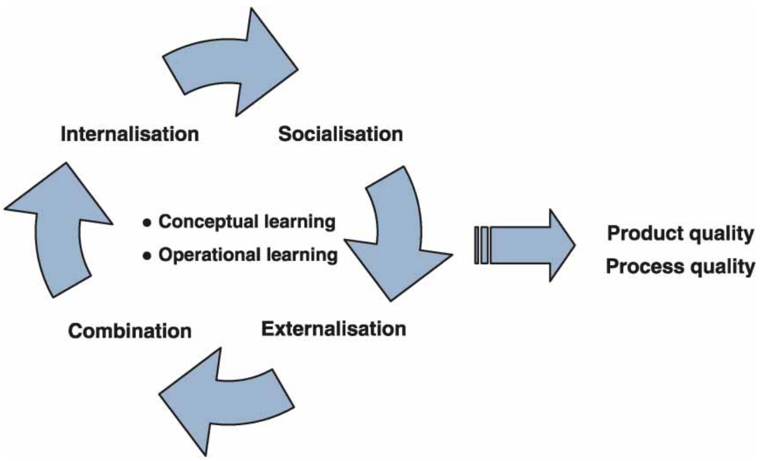

Figure 1. The conceptual model of knowledge management in a quality management programme.

extended from Nonaka and Takeuchi's (1995) theory, and combined with two types of learning (i.e. conceptual learning and operational learning) can lead to knowledge creation, and consequently lead to product and process quality improvement. The imperativeness of learning to knowledge creation and quality improvement has been well recognised in the literature (Garvin, 1993; Leonard-Barton, 1992; Rowley, 1999), and the link between knowledge creation and quality improvement has also been investigated in a few studies (Linderman et al., 2004; Mukherjee et al., 1998). However, it is more sensible to divide quality improvement in the software development process quality into process quality and product quality. Deutsch and Willis (1988) define software process quality as the capability required for the software development processes such as techniques, tools, employees, organisation and training; the software process quality now is commonly defined by the process capability maturity. Software product quality refers to the product features presented in the software development processes such as functional accuracy, efficiency, maintenance and the completeness of the documentation.

Method

The research design we followed was a case study analysis, which is the preferable method for analysis in the early stage of research on a topic when a fresh perspective is needed (Yin, 1994). Though the qualitative approach has been criticised for being less generalisable, it can yield broad and rich descriptions of actual events in real-life contexts which uncover and preserve the meanings that those involved ascribe to them (Gephart, 2004). In order to achieve the rigour of the qualitative approach, taking in-depth interviews with people who have directly experienced the phenomenon of interest process is essential (Patton, 1990). We also combined data collection methods such as interviews and use of archives. Our main questions were:

- What are the knowledge management processes embedded in CMMI?
- What are the barriers to effectiveness of CMMI?

We collected data about the case that implemented CMMI in 2004 and progressed to level 2 in 2005 through interviews and documentations. In total, we conducted in-depth interviews with seven interviewees (see Table 1). The people we interviewed work in various processes of software development such as project managers, software engineers and quality assurance managers. The reason for choosing only seven

Table 1. Interviewees' background information.

Interviewees	Years of service	Time spent	Expertise
Project manager	16	4 hrs	Large-scale system integration project management
Project manager	15	2 hrs	Open infrastructure system integration management
Project manager	11	2 hrs	Closed and open system integration project management
Software engineer	9	3 hrs	Product quality management
System designer	10	3 hrs	Large-scale system maintenance management
System engineer	16	3 hrs	The implementation and maintenance of customised systems and product systemisation
Quality manager	17	3 hrs	Quality management and quality project implementation

interviewees was that these people all had much experience of managing a development project that followed every step of the newly designed CMMI processes. We used a structured, open-ended interview guide, took copious notes, and collected and read documents that the interviewees gave us. Typically, the interview lasted from two to four hours. The analysis of data relies on interpretations of the people we interviewed; we look for common patterns that they shared and searched for consensus views among them. Most of the people we interviewed had been at the firm for more than 10 years.

The interview guide was developed based on a Balanced Scorecard (BSC) framework to understand the barriers to implementing CMMI. The BSC is an approach to performance measurement that combines traditional measures with non-financial measures to provide managers with richer and more relevant information about the activities managed (Kaplan & Norton, 1992). The framework organises impact clusters into four perspectives: financial, customer, internal process and learning vs. growth. As these measures are used in alignment with organisational strategy, the result not only provides an understanding about a firm's performance but also helps the firm to identify improvement areas.

Thus, the use of a BSC framework helps us to understand the impact of CMMI adoption not only on financial aspects of a firm, but also on customers, internal processes and organisational learning capabilities. Evaluating the impact of the disciplined knowledge management project on multi-dimensions not only helps in understanding the effectiveness of CMMI implementation but also enables the organisations to identify barriers against such adoption.

Although the case study approach has obvious merit, it is nonetheless limited by the fact that the findings are based on the participants' subjective understandings and perceptions. We recognised this limitation and have tried to overcome it in a number of ways. First, we tried to overcome the limitation by interpreting our findings through the lenses of literature. Second, we triangulated evidence from diverse sources. To the extent possible, we corroborated what the people we interviewed said by relying on the documents they gave us. These documents included organisational manuals and work instructions, and minutes of management meetings as well as reports by internal and external auditors. This helps us to ensure the validity of the interviews. Third, after completing drafts of the case study, we contacted some knowledgeable persons in the firm to review the draft and incorporate their comments into revised versions. We believe that future researchers could expand on our analysis with additional studies of companies of different sizes.

The case

The case study firm was established in 2000 and hired more than 300 employees working in software system development. The firm not only operates its business in Taiwan, but also has some subsidiaries in China. Customers of the firm range from a credit card centre to insurance firms and asset management firms. The services provided by the case company include:

- Application system development and maintenance service: this service is developed based on the customers' needs;
- Creating a packaged software program: this service requires the understanding of a specific industry and provision of solutions related to the industry.

In order to improve internal efficiency of the software development process, the firm adopted CMMI in 2004 and progressed to CMMI level 2 in 2005.

Findings and discussion

This study examined the activities of knowledge management and contents of knowledge managed in CMMI implementation and the effectiveness of CMMI on both software process and product quality through understanding the impact of implementing CMMI on financial, customer, process and learning and growth aspects. The knowledge managed and the activities of knowledge management in each stage are described. The last section discusses the consequences of the quality management programme.

Introduction stage activities

CMMI was introduced in April 2004; the impetus to adopt the system was customer dissatisfaction with existing software quality. In search for the solution, the case company identified that there was a growing tendency to adopt CMMI in the global market, but not many local Taiwanese firms adopted CMMI. Most of these firms were at their beginning of CMMI implementation, which encouraged the case company to adopt such a programme to increase software development quality and competitiveness in the software industry in Taiwan. The underlying philosophy of CMMI adoption the company held was to develop standard processes for software development procedures with the aim of improving quality and overall efficiency. The case company achieved CMMI level 2 in July 2005, and now it has progressed to level 3.

In order to stimulate knowledge creation at the introduction stage, the company adopted new practices that were aimed at increasing interdepartmental communications, facilitating organisational coordination, and changing the existing culture. Initially, a representative was selected from each department to participate in the meetings to discuss various issues such as how to improve existing processes in compliance with CMMI guidelines; what would be the most effective way of writing document templates; what review processes each department should follow and so on. An engineering-process-group (EPG) team, consisting of a project manager, a system analyst, a programming engineer and a manager from the testing department, was assembled to take responsibility for the CMMI implementation. The team members were selected for their expertise in the aspect concerned, as they were expected to coordinate between the EPG team and their own department.

It took almost nine months for the EPG team to finalise the first version of key process areas (KPAs) of developing and sustaining a measurement capability that is used to support management information needs. The KPAs were continuously adjusted after the first version. Within this period of time, a consultant was hired to audit the quality and reliability of the proposed KPAs in order to meet the CMMI standards. In addition, training was provided to enhance employees' skills and knowledge about new processes and new quality standards. All employees were tested after the training course to ensure that they understood the CMMI philosophy and were capable of performing under the new working procedures. At the same time, meetings and discussions took place in every department to facilitate knowledge sharing and to seek new working procedures that meet CMMI requirements. These indicated that CMMI adoption had stimulated a strong need for managing knowledge embedded in functional processes. In addition, knowledge sharing and knowledge creation were induced through socialisation, externalisation, combination and internalisation, which consequently ensured the consistency of software development processes. However, it was also identified that at the introduction stage, the case company neither shared tacit knowledge with customers, nor articulated customers' tacit knowledge into designing new software development processes.

From the above-mentioned activities during the introduction process, it can be seen that a continuous learning concept was embedded into CMMI. All employees were encouraged to learn about CMMI and how to do things in accordance with CMMI principles. Also, the creation of document templates provides a basis for future benchmarking. Moreover, the discussions at EPG level, interdepartmental level or departmental level allowed knowledge sharing and knowledge exploration. These findings reflect Linderman et al.'s (2004) view that standard quality management practices, in this case CMMI, help synthesise information from heterogeneous sources, allowing knowledge to be created through combination.

Implementation stage of knowledge management

The employees were not equipped with appropriate knowledge for conducting quality management processes. To begin with process efficiency, the primary purpose of adopting CMMI was to create efficient and standardised software development processes in compliance with CMMI guidelines. However, after setting up new processes which were audited by experts, the results did not turn out to be positive as expected due to the lack of integrating certain knowledge into new processes. For example, the interviewees pointed out that 'business domain and modelling knowledge was not equipped to those who were responsible for the processes of customer requirement management and requirement development'. This can lead to deficiencies in meeting customer requirements. The results of the case study revealed that 'the system engineers often had difficulties in validating the effectiveness of the built system architecture on the ground that they did not have the sufficient knowledge about customers' business processes', though following CMMI guidelines in iteratively verifying the requirements. For instance, system engineers did not have sufficient knowledge about credit card processing and relied on business users to provide the requirements of the system. Though the user requirements have been reviewed several times, it is very difficult for inexperienced engineers to decide on the completeness of the design of the business model. This finding is similar to some previous research in IT projects (Parker, 2000; Ranger, 2001): the failure to integrate business domain knowledge into the requirement management process hinders an organisation in defining, analysing, specifying and validating what is necessary for developing a system that meets customers' needs (Alshawi & Al-Karaghoul, 2003).

Moreover, though using a peer review process for the requirement management process is strongly recommended by the CMMI guidelines, such a peer review process did not help the organisation in terms of knowledge sharing and knowledge creation but rather created conflicts and dissatisfied customers. Most of the interviewees pointed out that 'a peer review process was time consuming and, in the reality, reduced the responsiveness to solve urgent issues.' Employees complained bitterly about the increased workload in doing a peer review in addition to the regular work schedule and most often there were disagreements among peers. This only led to conflicts among reviewers and prolonging the review process before a software project could actually be executed. In some cases, the peers were junior who did not have sufficient knowledge of business domain and the skills to run the customer requirement review processes adequately, which consequently resulted in poor product quality and customer dissatisfaction. This negative outcome of CMMI adoption was seen previously in other types of improvement project; as Jambekar (2000) notes, the inefficiency of improvement project adoption often resulted from adding to the employees' workload.

Next, all employees were provided at least an eight-working-day training course on the concepts, scopes and guidelines of CMMI to enhance employee learning and growth. Though employees had general ideas about CMMI, many interviewees reported dissatisfaction in learning and personal growth. They expected to enhance their knowledge and skills of the new way of developing software projects, but discovered that they had to learn only the process specific to their work rather than in a broad sense. An interviewee noted that 'my subordinates sometimes complained that they were trained to respond to problems rather than thinking of a better solution for solving a problem'. This indicated that the case company appeared to pay more attention to operational learning than conceptual learning. As pointed out by Mukherjee et al. (1998), merely focusing on operational learning could limit an organisation's capacity for quality improvement on the ground that operational learning does not help solve the root cause of problems. Besides, adopting only one mode of learning could put the firm into a weak position to create new knowledge.

From the aspects of customer satisfaction and organisational performance, it was revealed that the adoption of CMMI did not increase customer satisfaction; nor did it reduce software development costs. The case study revealed that customers were not happy with the new way of services. The most frequent complaint of the customers was that 'they used to be able to communicate with engineers or software designers directly about the project, but they now ought to follow the company's new processes and only to get something that might not be what they actually expected'. This indicated that customers' knowledge about the product is not integrated into the software development process while following CMMI.

As identified from the case study, CMMI emphasises software process quality – such as sophisticated skills, tools, employees, organisations and training – as a means of improving process efficiency of software development. Yet, CMMI did not focus on software product quality such as design methods, end user products such as documents, program codes, verifying methods and software framework design methods in software development processes. The incongruence of emphasis on two sides of one coin, in this case product quality and process quality, not only had hindered the knowledge creation and knowledge capturing from the customer side, but also resulted in customer dissatisfaction (Deutsch & Willis, 1988; Douglas & Fredendall, 2004). Customers were kept away from involvement in software development projects not only at the requirements management stage, but also in respect of system analysis and design. Though employees are assumed to have better knowledge and skills on system analysis and design, exempting communication with customers could also lead to some doubts about customers' preferences such as the appropriateness of interfaces, system performance, system architecture and inter-operability (Markus et al., 2002).

Quality improvement outcomes

As the software development processes were prolonged compared to pre-CMMI implementation as identified before, customers were not happy about this change either. This led the organisation to find low return of investment in CMMI, as some defects were identified such as a high rework rate, not being able to deliver project output on time, and the overall cost of software development was not reduced.

As a result, customer satisfaction was not increased to a satisfactory level. For instance, in eight large projects in total run after CMMI had been adopted in the company, the project delivery time was not improved. In particular, one of the software development projects was planned to be developed in nine months, but it actually took 22 months to deliver to the customer. This significant delay not only caused the schedule variance to increase to 144% (overtime) and the main working day (over cost) was increased from 4253 days (estimated) to 7738 days (actually used), but also the defect-SIT (system integration test) and defect-UAT (user assistant test) were reported as high as 426 times and 304 times. Defect-SIT refers to the defects identified by the engineers after the software product has developed and defect-UAT refers to the defects identified by the customer. Thus, it can be seen that though the case company had followed the software development process and complied with CMMI guidelines, the software process quality and software product quality were not improved after implementing CMMI in this project. Following investigation into the reasons for such significant delay in project delivery, it was found that the requirement stability index was only 12.97%, which indicated that the company was unable to manage customer requirements properly. As some of the interviewees pointed out, the main reason for not meeting customer requirements was the lack of business domain knowledge. In addition, system trouble reports and system change reports were increased dramatically from 18 times in August to 57 times in September. During this period of time, the case company was progressing to CMMI level 3. This implied that the CMMI experts audited the consistency of the software development process with the CMMI guidelines, but not the quality result of software developed. Major defects meant that the incidence of customers requiring refunds and compensation also increased from one case in June to eight cases in October, November and December. These defects provided strong evidence to prove that the quality was not improved as much as the case company expected and customer satisfaction did not reach a satisfactory level as hoped. Fortunately, the variance between planed software development time and actual developed time was narrowed down after 11 months of implementing CMMI.

Conclusion and implications

CMMI has been seen as best practice for improving the quality of the software development process. Though the fundamental objectives of CMMI and knowledge management are to increase organisational efficiency through effectively managing knowledge, little research has been done to examine the value of knowledge in CMMI and the effectiveness of CMMI from the standpoint of knowledge management. We examined the value of knowledge in the CMMI implementation stage, and the barriers to the effectiveness of CMMI in four aspects: process, learning/growth, customer and organisational performance based on the perspective of knowledge management. As Marcus and Naveh (2005) suggested, understanding the value of knowledge played in a systemised programme, such as CMMI in this case, enables us to understand how organisational knowledge is created and enhanced through a continuous improvement programme. At the introduction stage, through following the guidelines of CMMI, knowledge creation and knowledge sharing were identified. These knowledge management activities indeed helped organisations to establish software development process standards that meet the CMMI philosophy. However, after implementing CMMI, some barriers to effective

CMMI implementation were identified, as knowledge creation was limited to internal rather than incorporating customers' knowledge about products or services into the new process. This finding is partially consistent with that of Linderman et al. (2004), who identified that quality improvement practices could create more knowledge through socialisation, externalisation, combination and internalisation to achieve better firm performance.

We also identified some knowledge creation activities, in our case study, such as socialisation, externalisation, combination and internalisation, but the activities of socialisation and externalisation were limited to inter-organisation rather than involving customers, not to mention the articulation of customers' tacit knowledge into software development processes. Though CMMI has placed some focus on the requirement management, customers' knowledge about products is not considered during the process of software development. As identified, the failure to incorporate customers' knowledge into the software development process has resulted in dissatisfied customers. In addition, though CMMI fostered learning in the organisation, the studied company appeared to focus merely on operational learning rather than conceptual learning. As the conceptual learning was limited to only a few employees such as those in the EPG team, most employees were only trained to respond to problems within his/her own working boundary. This also brought a lot of anxiety to employees, as they were afraid of stagnating in terms of their growth and learning. As suggested by Mukherjee et al. (1998), a combination of conceptual learning and operational learning can facilitate knowledge codification and enables better problem solving and effective dissemination of new knowledge. Moreover, we also identified another cause underlying the barrier to effective CMMI implementation, which was the incongruence of focus – on process quality rather than product quality. CMMI puts considerable emphasis on ensuring the quality of the software development process, but this, as identified in this study, was not enough to satisfy customers. With the lack of attention to one of two sides of the coin (the product and process quality), the CMMI adoption has brought deficiencies to organisations in terms of financial performance, meeting customer expectations, increasing process efficiency and facilitating internal growth.

Based on our findings on the case study, there are some implications for those who intend to adopt CMMI or those who are interested in implementing quality improvement programmes to advance organisational performance. First, from the standpoint of knowledge management, it is important to integrate knowledge from both internal (i.e. employees) and external (i.e. customers) sources so as to achieve better customer satisfaction through enhanced quality improvement. Second, implementing a programme such as CMMI induces learning, but to focus solely on operational learning is not effective for employees' learning and growth. Third, customer satisfaction is affected not only by process quality but also by product quality. However, CMMI has focused mostly on process quality.

Additionally, top management support is one of the critical success factors in a large-scale project adoption. The adoption of CMMI requires consistent investments in improving various processes and enhancing employees' knowledge and skills. However, the return on investment in quality improvement projects or knowledge management projects cannot be seen in a short period of time, thus top managers' commitment to the continuous improvement is crucial. In addition, a sound reward system can be another mechanism to shorten the return on investment and to increase process efficiency, as motivated employees are more committed to organisations. Providing reasonable expectations for customers is also vital. It is important to communicate with customers about the changes and limitations of developing a system, rather than providing empty promises.

The finding about the gap between knowledge management and knowledge management processes in the software quality management programme can also be applied to other quality management programmes such as Information Technology Infrastructure Library (ITIL) and International Organization for Standardization (ISO) programmes which emphasise the process of knowledge management but do not address the content of the knowledge managed. As a result, though the quality of processes may have complied with the global standards the quality of the actual products or services delivered may not be satisfactory.

Last, some limitations of this study need to be highlighted regarding the sample and method. First, we used a case study as a sample; while the case study company had progressed to CMMI level 2, it is important to have more organisations with different CMMI levels to enhance our understanding on whether or not CMMI causes similar impact on other organisations. Second, the case study company operated in the system integration industry; this again restricts our understanding of organisations engaging in different process development businesses. Further studies can expand the sample size to organisations achieving different CMMI levels and operating in different software process development fields.

References

Alshawi, S., & Al-Karaghoul, W. (2003). Managing knowledge in business requirements identification. *Logistic Information Management*, *16*(5), 341–350.

Chrissis, M.B., Konrad, M., & Shrum, S. (2003). *CMMI guidelines for process integration and product development*. New York: Addison-Wesley.

Dada, M., & Marcellus, R. (1994). Process control with learning. *Operations Research*, *42*(2), 323–336.

Dean, J.W., & Bowen, D.E. (1994). Management theory and total quality: Improving research and practices through theory development. *Academy of Management Review*, *19*(3), 392–418.

Deutsch, M.S., & Willis, R.R. (1988). *Software quality engineering: A total technical and management approach*. London: Prentice Hall.

Douglas, T.J., & Fredendall, L.D. (2004). Evaluating the Deming management model of total quality in services. *Decision Science*, *35*(3), 393–422.

Flynn, B.B., Schroeder, R.G., & Sakakibara, S. (1994). A framework for quality management research and an associated measurement instrument. *Journal of Operations Management*, *11*(4), 339–366.

Garvin, D.A. (1993). Building a learning organization. *Harvard Business Review*, *71*(4), 78–91.

Gephart, R. (2004). Qualitative research and the Academy of Management Journal. *Academy of Management Journal*, *47*(4), 454–462.

Glazer, H. (2003, September 12). Commentary: Two key challenges to implementing the CMMI. *The Daily Record,* p. 1.

Jambekar, A.B. (2000). A systems thinking perspective of maintenance, operations, and process quality. *Journal of Quality in Maintenance Engineering*, *6*(2), 123–132.

Kaplan, R., & Norton, D.P. (1992). The Balanced Scorecard: Measures that drive performance. *Harvard Business Review*, *70*(1), 71–79.

Leonard-Barton, D. (1992). The factory: A learning laboratory. *Sloan Management Review*, *34*(1), 23–38.

Levett, G.P., & Guenov, M.D. (2000). A methodology for knowledge management implementation. *Journal of Knowledge Management*, *4*(3), 258–276.

Linderman, K., Schroeder, R.G., & Zaheer, S. (2004). Integrating quality management practices with knowledge creation processes. *Journal of Operation Management*, *22*(6), 589–607.

Marcus, A., & Naveh, E. (2005). How a new rule is adjusted to context: Knowledge creation following the implementation of the ISO 9000 quality standard. *International Journal of Organizational Analysis*, *13*(2), 106–126.

Markus, M.L., Majchrzak, A., & Gasser, L. (2002). A design theory for systems that support emergent knowledge process. *MIS Quarterly*, *26*(3), 179–213.

Mukherjee, A.S., Lapre, M.A., & Van Wassenhove, L.N. (1998). Knowledge driven quality improvement. *Management Science*, *44*(11), S35–S49.

Nonaka, I., & Takeuchi, H. (1995). *The knowledge creation company: The Japanese companies create the dynamics of innovation*. New York: Oxford University Press.

Pan, S.L., & Scarbrough, H. (1999). Knowledge management in practice: An exploratory case study. *Technology Analysis and Strategic Management*, *11*(3), 359–374.

Parker, A. (2000, July 5). Common committee calls for action on IT fiascos. *Financial Times*, p. 1.

Patton, M.Q. (1990). *Qualitative evaluation and research methods* (2nd ed.). Newbury Park, CA: Sage.

Paulk, M., Curtis, B., & Wener, C. (1993). *The capability maturity mode for software, version 1.1* (Technical report CMU/SEI-93-TR-024). Pittsburgh, PA: Software Engineering Institute.

PR News. (2007, July 16). Objectiva's software development process achieves CMMI Level 5 certification. *PR News.* Retrieved July 16, 2007, from http://www.freshnews.com/news/computers-internet/article_38864.html?Document+Sciences

Ranger, S. (2001, July 5). State IT failure squanders £1 bn: Our survey counts the cost of pathway – NIRS and the reset. *Computing*, p. 1.

Rowley, J.E. (1999). Knowledge management: Philosophy or discipline? *Journal of Knowledge Management*, *3*(3), 172–179.

SBN. (2007). Project management success with CMMI: Seven CMMI process areas. *Scitech Book News*, *31,* 3.

SEI. (2002). *CMMI–SE/SW/IPP/SS, continuous representation, V1.1*. Pittsburgh, PA: Software Engineering Institute.

Staples, M., Niazi, M., Jeffery, R., & Abrahams, A. (2007). An exploratory study of why organizations do not adopt CMMI. *Journal of Systems and Software*, *80*(6), 883–895.

Yin, R. (1994). *Case study research: Design and methods* (2nd ed.). Beverly Hills, CA: Sage.