Applied Mathematics & Information Sciences
An International Journal

Comments on Software Process Improvement Methodologies Using QFD

Wei Xiong[1], *Yonghui Cao*[1,2]

[1]School of Management, Zhejiang University,Hangzhou 310058 P. R. China
[2]School of Economics and Management, Henan Institute of Science and Technology, Xinxiang 453003 P. R. China

Received: 7 Oct. 2012, Revised: 21 Dec. 2012, Accepted: 23 Jan. 2013
Published online: 1 May 2013

Abstract: With rapid technological innovation and changes, the key to the survival company is the continuous improvement of its process. In this paper, we evaluate software process improvement (SPI) methodologies using QFD. There are Richardson's approach, Zultner's approach and SAP'S QFD for SPI. Then, we put on proposed model: CMMI staged model using QFD.

Keywords: Software Process Improvement, CMM, QFD

1 Introduction

Software Process Improvement (SPI) has become the key to the survival of many software development organizations. Many international SPI models/standards are developed for SPI. These models and standards share some common concerns in terms of quality and process improvement. However, their emphases are different. For instance, the ISO standard addresses the minimum criteria for a quality system. It is unfair to make a judgment on which one is better [1]. However, considering the more detailed guidance and greater breadth provided by CMM or CMMI, it may be a better choice for some software development organizations [2].

The CMMI model provides a structured view of process improvement across an organization. It can help set process improvement goals and priorities, provide guidance for quality processes and provide a yardstick for appraising current practices. Table 1 shows the different CMMI maturity levels and their characteristics at that stage. The purpose of the CMMI effort is to support process and product improvement and to reduce redundancy and eliminate inconsistency experienced by those using multiple standalone models [3].

The CMMI is designed to cover processes across the full development cycle and is designed to be broad enough to cover a wide range of systems and products [3]. It is recognized that not all process areas or practices within a process area may apply to a specific organization. Therefore, tailoring criteria are provided that allows the CMMI to be used with only the processes or activities that apply [3].

Table 1. CMMI Maturity Levels

Maturity Level	Staged Representation Maturity Level	Characteristics
1	Initial	Process unpredictable poorly controlled and reactive
2	Managed	Process characterized for projects and is often reactive
3	Defined	Process characterized for organization and is proactive
4	Quantitatively Managed	Process measured and controlled
5	Optimizing	Focus on process improvement

In this study, frameworks were developed to help map business and other process requirements of an organization to CMM elements, and help develop action

* Corresponding author e-mail: wxiong@zju.edu.cn, caoyonghui2000@126.com

plans to satisfy those requirements using Quality Function Deployment (QFD).

2 QFD Application in Software

In the manufacturing field, QFD is used to focus on the quality aspects of projects. It could also be used in a software engineering g environment. The success of any software organization stems from customer satisfaction, and customer satisfaction comes from receiving a quality software product. Thus, concentrating on quality pulls the organization ahead of the intense competition, and ultimately brings success.

Customers want value from their software. They want the product to help them solve problems and seize opportunities. It is also important to understand that customer actually have three types of requirements. These are normal, expected, and exciting requirements (Zultner, 1993). Normal requirements are those that can be gathered by simply asking the customer. Expected requirements are requirements are requirements that are not mentioned but are expected. An on-line-help system is an example of this. Exciting requirements are requirements that are unexpected, but highly satisfying when they are delivered. These are the product features that really impress customers, or are made possible by new technology that the customers are not aware of.

To put quality into a software product, a software engineer has to understand what is meant by "software quality". There are two views of software quality-the traditional view and the more modern view (Zultner,1993) The traditional view focuses on the minimization of defects. This is accomplished through existing software engineering approaches such as code inspections, reviews, walk thoughts, and testing. With this view, the software engineer understands the causes of defects and strives to detect and correct them. The modern view of software quality aims at maximizing the value of the software. The software engineer understands the needs of the customer and designs value into the system. The difference between the two views is very important. With the traditional view, the best one can do is to have no defects in the system. However, even if a product has no defects, it is not necessarily of value to a customer. Therefore, the traditional view of software quality is insufficient. Furthermore, over 50% of software development errors occur in the requirements analysis phase (Eriksson 1998). These "defects" cannot be caught by the traditional view.

The software engineer needs to maximize the value in software products. This is accomplished by determining what is of value to the customers. These areas become the priorities of the project, and the team's best efforts are concentrated there. The task of determining what is of value to customers is not easy, and should be done with an approach that is systematic and quantifiable. This is where QFD plays an important role. QFD can be used to accomplish several things. It can be used to evaluate the impact of product features on customer value, and be used for considering trade-offs of product features in the design. It can also be used to set a development strategy or direction. For instance, one can use QFD to determine whether a software package should aim for technical excellence or have improved ease of use. Finally, the House of Quality in QFD can be used to analyze competitive products as well.

3 Comments on Software Process Improvement Methodologies Using QFD

3.1 Ita Richardson's Approach

Ita Richardson discussed the importance of improving software process to small software development companies, as are the difficulties faced by these companies when implementing such improvements [4]. He thinks that the generic Quality Function Deployment Software Process model consists of two matrices, the Business Process matrix and the Software Process Improvement House of Quality. This generic model can be used by software development companies to provide an action plan for use by the organization. The development of the action plan is represented in Figure 1.

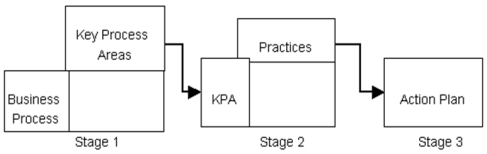

Figure 1. Using the generic QFD model

The first stage of this model is Measurement of Business Processes. This stage is optional. Businesses will be asked to focus on their business requirements, and to provide three measurements: current performance, planned performance, and the importance of this particular requirement on their business. The outcome from this section of the model would then be used in implementation stage 2.

The second stage is Measurement of Key Process Areas. Companies can indicate how they currently perform in each Key Process Areas (KPA). They are also required to give measurements for their planned performance and the importance of that key process area

to their business. Another feature of this stage is that a cost/time effective scale can be included. As an initial step an organization may be interested in pursuing those items which would have an immediate effect or items which are not too costly. The Quality Function Deployment Software Process Improvement model allows these to be taken into account.

The third stage is developing an Action Plan. Once self-assessment has been input to SPI/HoQ in Stage 2, the priority practices to be pursued are now identified. These are then incorporated within a software process improvement action plan for implementation within the organization.

The generic QFD model was later improved into a four-stage model, as is represented in Figure 2.

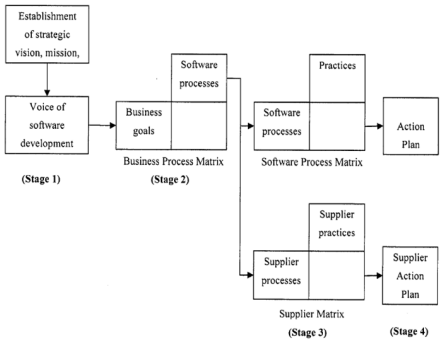

Figure2. A Generic SPI/QFD Model

In the first stage, mission statement of the company which needs the establishment of strategic vision, mission is identified with business goals. The identified goals represent the voice of this software development company. In the second stage, business goals are correlated with software processes using business process matrix. In the third stage, Software processes and practices are correlated by using software process matrix. The relationships used in the matrix are generic, indicating the effect that a practice will have on the process. The importance of each practice is calculated from process importance and relationship values. Similarly, importance of supplier practices is calculated using supplier processes. In stage 4, the action plan is derived. The action plan indicates the prioritized practices. Action plan helps the company to decide the order of important practices to improve upon, in order to influence software processes, and consequently business goals. Similarly, the supplier action plan is derived from supplier matrix.

Richardson's generic model is useful for small companies because their self-assessments are easy. Of course, the small companies can use the matrices readily without extra efforts and investments. However, in larger companies, the organizational structures become more complex, which makes self-assessment in this model more difficult. Also it does not deal with inter relationships between the practices in action plan, which correspond to roof of house of quality in QFD.

3.2 Zultner's approach

Business Process Reengineering (BPR) is considered to dramatically improve the core business process-software development by many software organizations. Zultner's BPR model with QFD divided the Business Process Reengineering into four major phases: analysis of the current development process, generation of new process concepts (alternatives), selection of the best new development process, and implementation of the selected new process [5], the complete process in the diagram is represented in Figure 3.

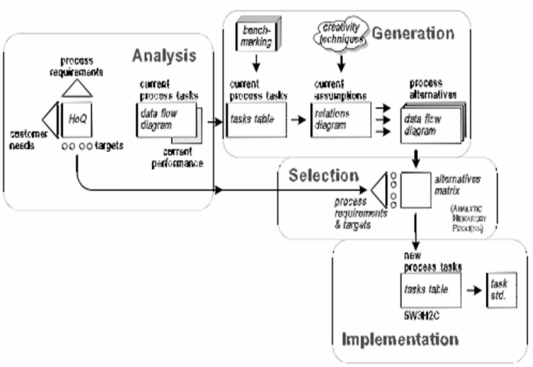

Figure3. QFD Process for Business Process Reengineering

In order to analyze the current development process and define the requirements for the new reengineered process, it needs to use the House of Quality matrix and a data flow diagram, the organization must understand what is wrong (and right) with the current process and what the current process is. The Customer Needs/Process Requirements the House of Quality matrix which shows what the new development process must be and do in order satisfy the customer requirements.

After analysis, the starting point in developing a new software process, is inventing the new process. There are two basic sources for new idea: we can examine the current process, and we can work from completely fresh assumptions. Benchmarking and creativity techniques, used in QFD for product development, are used here to generate ideas for new processes.

Among the generated new process alternatives, selecting the best one is a crucial step. The criteria in the Process Requirements/Alternatives matrix are the process requirements from the Customer Needs/Process Requirements HoQ in Analysis phase. Use the process requirements from the House of Quality, and Analytic Hierarchy Process (AHP) to select the best of your process alternatives for the new development process.

Once the best new development process has been selected, define the new process tasks in detail with a QFD tasks table in implementation phase. The table lists the new process tasks by descriptive name (What), the performer(s) of the task (Who), the location (Where), the schedule (When), the rationale or purpose (Why), a cross-reference to the details of doing the task (How), the target value for the task performance level (How much), the estimated effort required (How expensive), and how the task steps can be checked (Check points), and how the task results can be inspected (Control point) (Akao, 1964) [6]. This 5W3H2C table provides a complete operational definition of the new process's tasks for use as a detailed implementation guide.

Zultner's approach uses either the major competitor's performance or creative thoughts of employees, but not existing standards which are widely used in a particular industry, as the source of process improvement. Although this approach may help address specific issues in an organization, it is difficult to apply this approach in different situations or environments to produce consistently efficient process improvement results when compared with a method using a popular reference model. This is because elements in this approach such as creative thoughts, are not always dependable.

3.3 SAP'S QFD for SPI

The QFD project described in this paper was based on this client-supplier relationship between QaP and the Quality Managers. Its primary purpose was to give the Quality Managers effective aid in improving the development process within their groups. To do so, it was necessary to identify and prioritize the general requirements SAP's software developers and management had regarding their software development process and contrast them with the appropriate process improvements. Because the Quality Managers are fully integrated into development, they possess a great deal of inside knowledge about the process in their groups. Therefore, their involvement in the project was of great importance to its success.

The project was initiated as a pilot project in the spring of 1997, with the goal of assessing QFD's suitability for software process improvement. If it were successful, it would be considered to institutionalize the use of QFD in SAP's software process improvement efforts. Because of previous successful cooperation, the project was carried out in cooperation with the German QFD Institute and the Chair of Business Computing of the University of Cologne.

The project structure was oriented on Deming's Plan-Do-Check-Act cycle (see Deming, 1986) [7].

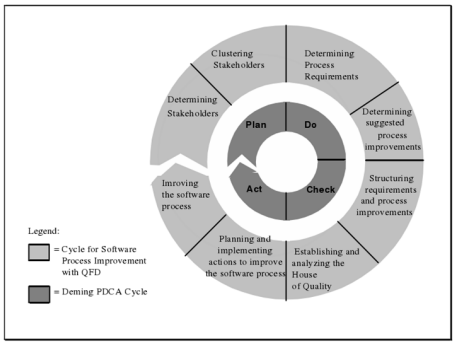

Figure4. the cycle of software process improvement with QFD

4 Proposed Model: CMMI staged model using QFD

4.1 Capability Maturity Model Integration (CMMI)

CMMI is an approach developed by Software Engineering Institute (SEI) to provide organizations elements for effective processes, which include development, acquisition and maintenance of products or services. It is used to integrate systems engineering, software engineering and process development developments in a single framework. CMMI can be used for: 1)Product and service development; 2) Service establishment, management, and delivery; 3) Product and service acquisition (CMMI Product Team, 2006) [8]. Literature survey in this study focuses on CMMI for product and service development. There are several important concepts in CMMI, including representations, maturity and capability levels and process categories and process areas.

(1)Representation Types

There are two representations of CMMI, continuous and staged. Either of the representations type needs to be chosen for a project. With the continuous representation which is more flexible, it is possible to select certain process areas of CMMI. It is also possible to improve different process areas at different intensity. The staged representation is more systematic and structured alternative. With this approach, every process needs to be addressed at the same rate. Instead of specifying the

process areas and their particular capability levels to be developed as in continuous representation, a maturity level is defined for the entire project.

(2)Capability and Maturity Levels

The other important concept is levels, that is, capability and maturity levels. The term "capability level" is used in the context of continuous representation of improvement, which allows the organization to choose specific improvement areas and improve them incrementally. Levels are used to show how ideal a certain process is, or the organization as a whole. There are six capability levels starting with number 0: 0) Incomplete: This level indicates that a process is not performed, or only partially performed; 1) Performed: This level indicates that a process is performed, meaning that it satisfies necessary goals of the particular process area; 2) Managed: This level indicates that a process is managed, meaning that it was planned and implemented according to organizational policies that involve resource allocation, stakeholder involvement, monitoring, controlling, testing, and evaluating; 3) Defined: For a managed process the standards, process descriptions and procedures can be very different for each specific instance of a process across different projects. For a defined process, standards, process descriptions and procedures must conform to the organizations standards, and be more consistent; 4) Quantitatively Managed: A quantitatively managed process is managed and controlled using quantitative techniques, such as statistical ones; (5) Optimizing: An optimizing process conforms to all previous maturity requirements, and focuses on continually improving process performance (CMMI Product Team, 2006) [8].

The term "maturity level" is used in the context of staged representation of improvement, which is concerned with the overall maturity of the organization and it allows organizations to improve processes in a set of processes areas. Maturity levels are very similar to capability levels, in that they reflect levels of planning and understanding of the processes. There are five maturity levels and are denoted by numbers ranging from 1 to 5:1) Initial: At the initial level, processes are usually not planned and chaotic. Success in these processes depends on the individual skills or people working in the organization; 2) Managed, which is as the capability level 2; 3) Defined, which is same as the capability level 3; 4) Quantitatively managed, which is same as the capability level 4; 5) Optimizing, which is same as the capability level 5 (CMMI Product Team, 2006) [8].

(3)Process Categories and Process Areas

There are four process area categories, and 22 process areas at CMMI for product and service development (CMMI Product Team, 2006) [8]. If a continuous representation is selected, an organization has the freedom to select a desired number of process areas, and develop each at different capability levels. If a staged representation is selected, first a maturity level is chosen. Some process areas are only addressed at certain maturity levels.

The process categories are as follows:

Process Management: This category involves five process areas that are oriented towards "defining, planning, deploying, implementing, monitoring, controlling, appraising, measuring, and improving processes" (CMMI Product Team, 2006, p.52) [8]. This process category involves process areas of organizational process focus, organizational process definition, organizational training, organizational process performance and organization innovation and deployment.

Project Management: This category involves process areas activities related to "planning monitoring and controlling the project" (CMMI Product Team, 2006, p.55) [8]. This process category involves process areas of project planning, project monitoring and control, supplier agreement management, integrated project management, risk management and quantitative project management.

Engineering: This category involves process areas that are related to development and maintenance activities across engineering disciplines. This process category involves process areas of requirements development, requirements management, technical solution, product integration, verification and validation.

Support: This category involves process areas that are used to support product development and maintenance. This process category involves process areas of configuration management, process and product quality assurance, measurement and analysis, decision analysis and resolution and causal analysis and resolution.

(4)Goals and Practices

In CMMI terminology, a goal may involve several practices that need to be implemented. There are generic goals and practices, and specific goals and practices. Same generic goals and generic practices apply to all process areas. Application of generic goals and specific goals into process areas is mandatory in CMMI implementation. Generic goals and practices exist at corresponding capability or maturity levels. For example, "Institutionalize a Managed Process" is a generic goal at the maturity or capability level 2. "Plan the Process" is a generic practice among ten generic practices within that generic goal. If a capability or maturity level of 2 is targeted for example, all the generic goals and generic practices at level 1 and level two need to be implemented.

In addition, there are specific goals and specific practices that are particular to each process area. Specific goals and practices exist at different levels corresponding to capability and maturity levels. Specific goals and specific practices are required to be implemented. For example, "Develop the Design" is a specific goal for the process area "Technical Solution" at the capability or maturity level 2. "Design the Product or Product Component" is a specific practice among four specific practices within that specific goal. If a capability or maturity level of 2 is targeted, all the specific goals and

specific practices at level 1 and level two that are particular to selected process areas need to be implemented.

4.2 CMMI staged model using QFD

The SPI framework for CMMI staged model, as shown in Figure 5

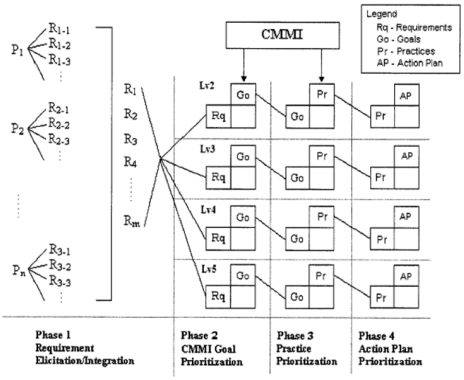

Figure 5.Software Process Improvement through CMMI Staged Model Using QFD

For each of the four maturity levels, the set of requirements with adjusted priorities are related to the goals. The goals are prioritized based on those process requirements. Thus, the goals that achieve higher overall satisfaction of process requirements get higher importance. CMMI staged model has generic practices categorized into four common features as well as the specific practices which correspond to the "Activities Performed" common feature. The priorities of Practices are determined by their correlations with goals. Thus, the generic practices in each common feature and the specific practices are prioritized separately based on the priorities of the goals. Practices that aim to achieve higher overall satisfaction of goals receive higher importance values. Separate sets of action plans are derived from the generic practices in each of the common features as well as from the specific practices. The actions that help to support more important Practices receive higher priorities.

As a result, the process requirements are reflected in PA goals, Practices, and the actions. The actions both follow the process maturity standards in CMMI staged model and satisfy the process requirements. Those actions with higher importance values help to achieve higher process requirements satisfaction.

Because of the close resemblance between CMMI staged model and CMM, the four phases for the SPI framework based on CMMI staged model as shown in Figure 2.

In Figure 2, phase 1 is exactly the same with the SPI framework based on CMM. Various perspectives are represented as P1 through Pn. Each perspective contains multiple requirements. The software process requirements in perspective 1 are represented as R1-1, R1-2, etc. These perspectives of software process requirements can then be prioritized based on their relative importance within the organization and integrated into one single set of requirements. In Figure 2, these integrated requirements are represented as R1 through Rm, where m is the total number of software process requirements from all perspectives. The prioritization ensures that requirements from different perspectives are comparable with each other, and the integration reflects the correlations among requirements from different perspectives. The deliverable of this phase is a set of prioritized and integrated software process requirements, which serves as the input to the next phase.

The second through fourth phases of this framework are applied to Level 2 to Level 5 of the CMMI staged model. The prioritized and integrated requirements from Phase 1 are linked to all goals in each of the four levels in CMMI staged model using relationship matrices. These prioritized goals are used as the basis for the prioritization of Practices. Finally, the prioritized Practices are transformed into prioritized action plans using House of Quality (HoQ).

In the second phase, which is "CMMI goal prioritization", the goals of all PAs in a particular maturity level are selected and prioritized based on the requirements from the previous phase. This phase helps to achieve two important objectives. First, the organization needs to comply with the CMMI standard. At the same time, the organization needs to ensure that by reaching a particular maturity level, the process is also satisfying the business and other requirements within the organization. In Phase 2, a relationship matrix is used to establish connections between the requirements from the organization and the goals in CMMI. This matrix demonstrates that complying with the CMMI standard also helps satisfy the business and other requirements in the organization. Second, the final set of action plans needs to be prioritized based on the priorities of requirements so that more important actions receive more resources. The goals serve as the bridge between requirements and the action plan. By prioritizing the goals, requirements from the organization can be transformed to the Practices in the third phase, and finally to the action plans in the final phase. In this way, a set of actions can be executed not only to achieve a specific maturity level in CMMI, but also to satisfy organizational process requirements.

The third phase of the framework is "practice prioritization". It involves the prioritization of Practices within all PAs of a specific level. The prioritization is carried out on the basis of the deliverables from Phase 2.

According to CMMI specifications, all these Practices have to be performed in order to reach that particular maturity level. These Practices serve as a bridge between the requirements and the final actions, and it is necessary to know how these Practices reflect the software process requirements. In order to show the connections between the requirements and the final action plans, these practices have to be prioritized based on the goals, which are now reflecting requirements priorities. The mapping between the goals and Practices has been has been clearly shown in CMMI documentation.

The fourth phase of the framework is "action plan development and prioritization". A set of actions is derived from the prioritized Practices. These actions should reflect the requirements integrated in the first phase. Meanwhile, they also state what needs to be executed in order to reach a particular CMMI maturity level. These actions guide the process improvement. Thus, more resources should be assigned to those actions with high priorities.

As shown in the above framework, by incorporating requirements from the organization into action plans through goals and Practices, the connection between the objectives of the organization and CMMI maturity levels becomes clear.

5 Conclusions

In this paper, SPI frameworks are developed to derive action plans based on software process requirements with the help of QFD and in accordance with CMM. The proposed framework integrate the best features of the existing methodologies, such as using QFD to translate process requirements into the action plan and integrating the process requirements from multiple groups of stakeholders, and addresses the limitation of the previous studies, such as omitting the differences among different groups of stakeholders and lack of conformance to reference models.

CMM is chosen in this framework because of their popularity in the industry and proven effectiveness, CMM, for many years, has shown positive results in terms of both tangible benefits such as cost, schedule, product quality, productivity, and amount of rework and intangible benefits such as improvements in the quality of work life, organization communications; organization learning and efficiencies; the ability to attract, retain, and develop software professionals; and the coherency of its organization culture.

Acknowledgements

This work is financially supported by the National Natural Science Foundation of China (Project No. 90718038). Thanks for the help.

References

[1] Paulk, Mark C. A Comparison of ISO 9001 and Capability Maturity Model for Software. Technical Report. CMU/SEI-94-TR-12, ESC-TR-94-12, July, 1994.
[2] Francois Coallier, How ISO 9001 Fits Into the Software World, IEEE Software, Vol. **11** No. 1, January 1994, pp. 98-100.
[3] Dennis M. Ahern, Aaron Clouse, Richard Turner, CMMI. Distilled: A Practical Introduction to Integrated Process Improvement, Second Edition.
[4] Ita Richardson. Quality Function deployment - A Software Process Tool? Third Annual International QFD Symposium. Linkoping, Sweden, Oct. 1997.
[5] Zultner, Richard E. Business Process Reengineering with Quality Function Deployment: Process Innovation for Software Development. 7th Symposium on QFD (ISBN1-889477-07-9), 1995.
[6] Akao, Yoji. 1964. Check Points, Control Points, and Evaluation Points (in Japanese). Quality Control 15 (Spring Special Issue): 42-48. Available from JUSE. Seminal article on control points and check points.
[7] Deming W.E.(1986). Out of the crisis. Cambridge MA:MIT Center for Advanced Engineering.
[8] CMMI Product Team. CMMI0 for development, version 1.2. Retrieved July 13, 2007, from Carnegie Melton University, Software Engineering Institute. August. 2006.
[9] Song, Ki-won; Kim, Jin-soo. Measurement and Management of the Level of Quality Control Process in SoC (System on Chip) Embedded Software Development, International Journal of Advanced Robotic Systems, APR 5 2012.

Dr. Wei Xiong is professor of the school of management Zhejiang University, doctoral tutor. His research interests are in the areas of QFD, routing systems, and information systems.

Yonghui Cao received the MS degree in business management from Zhejiang University in 2006. He is currently a doctorate candidate in Zhejiang University. His research interest is in the areas of management information systems.