CMMI$^{sm}$ (CMM Integration$^{sm}$) Appraisal Tool (CAT Tool)

Mohamad Nazim Jambli (1898309)

School of Computing and Information Technology
Griffith University

Submitted in partial fulfillment of the requirements of the degree of;
Master of Information and Communication Technology

June 2003
Acknowledgements

This research project report has been, frankly speaking, hard to write. As a result, I have a long list of people to thank. Lucky for me, I have wonderful people that always standing behind me to support and give me inspiration to complete this research project.

Nyco Agung, my hardworking research project development team partner. Thanks a lot for your help, support, and understanding in development and completion of our research project.

Dr. Jennifer Gasston, my helpful and friendly supervisor. Thank you for reviews, revises, and discusses on the structure and content of my research project report and presentation.

Mrs. Angela Tuffley, my supervisor and client. Thanks for discusses, reviews, revises, and agrees on the requirements and technical parts of CMMI\textsuperscript{sm} (Capability Maturity Model Integration\textsuperscript{sm}) Appraisal Tool (CAT Tool) prototype.

Finally, I would never finish this report without the support of my loving wife, her parents, and my parents. Thanks everyone!
# Table of Contents

1. INTRODUCTION ...................................................................................................... 1
   1.1 Purpose of the Report .................................................................................. 3
   1.2 Research Problems ................................................................................... 3
   1.3 Research Project Significance ................................................................... 4
   1.4 Research Project Potential Outcomes ....................................................... 5
   1.5 Research Project Approaches ................................................................. 5
   1.6 Overview of the Report ............................................................................ 6

2. LITERATURE REVIEW ........................................................................................... 7
   2.1 Software Quality ........................................................................................ 7
   2.2 Software Quality Management System ................................................... 9
   2.3 Software Process Improvement .................................................................. 12
   2.4 Software Process Assessment .................................................................... 13
   2.5 Software Process Assessment Models ..................................................... 15
      2.5.1 CMM .................................................................................................. 16
      2.5.2 BOOTSTRAP .................................................................................. 18
      2.5.3 ISO/IEC TR 15504 ........................................................................ 20
      2.5.4 CMMI .............................................................................................. 24

3. EVALUATION OF EXISTING TOOLS ................................................................ 30
   3.1 CMM-Quest .............................................................................................. 30
   3.2 SPICE 1-2-1 ........................................................................................... 31
   3.3 BootCheck ................................................................................................. 32

4. RESEARCH METHODOLOGY .............................................................................. 36

5. SYSTEM METHODOLOGY ................................................................................... 38

6. PROTOTYPE REQUIREMENTS AND SPECIFICATIONS ................................. 41

7. PROTOTYPE IMPLEMENTATION ....................................................................... 43

8. CONCLUSIONS AND FUTURE WORK ............................................................ 46

REFERENCES .................................................................................................................... 48

Appendix A: Project Plan ............................................................................................. 51
Appendix B: Statement of User Requirements and Acceptance Criteria ............... 52
Appendix C: Software Requirements Specification .................................................. 53
Appendix D: System Design and Description ............................................................ 54
Appendix E: Test Plan ................................................................................................. 55
List of Tables

TABLE 1: FIVE DIFFERENT APPROACHES TO THE DEFINITION OF QUALITY (GARVIN, 1984) ......................... 8
TABLE 2: COMPARISON OF PRINCIPAL IDEAS (OAKLAND, 1989) ................................................................. 10
TABLE 3: THE POTENTIAL BENEFITS OF SOFTWARE PROCESS IMPROVEMENT (SPI) (SANDERS, 2001) ...... 12
TABLE 4: CHRONOLOGICAL ORDER OF PROCESS ASSESSMENT MODELS (WANG AND KING, 2000) ......... 16
TABLE 5: CAPABILITY LEVELS AND PROCESS ATTRIBUTES (ROUT, 2001) .................................................. 21
TABLE 6: CAPABILITY LEVEL RATINGS (ROUT, 2001) ....................................................................................... 22
TABLE 7: SPICE RATING SCALE (SQI, 2003) ........................................................................................................ 23
TABLE 8: CMMI RULES FOR CHARACTERIZING (SEI, 2001B) ....................................................................... 27
TABLE 9: CMMI RULES FOR AGGREGATING INSTANTIATION-LEVEL CHARACTERIZATIONS (SEI, 2001B)... 27
TABLE 10: CMMI CRITERIA FOR RATING PROCESS AREA (SEI, 2001B) ....................................................... 28
TABLE 11: KEY FEATURES OF EXISTING SOFTWARE PROCESS ASSESSMENT SUPPORTING TOOLS .......... 33
TABLE 12: EXPECTED KEY FEATURES OF CAT TOOL ..................................................................................... 35
TABLE 13: WORK BREAKDOWN STRUCTURE FOR CAT TOOL ......................................................................... 43

List of Figures

FIGURE 1 - DEMING'S PLAN-DO-CHECK-ACT CYCLE OF IMPROVEMENT (GILLIES, 1992) ....................... 11
FIGURE 2 - FOCUSED PROCESS IMPROVEMENT CYCLE (SANDERS, 2001) .................................................. 14
FIGURE 3 - A TREE STRUCTURE OF BOOTSTRAP PROCESS MODEL (BICEGO, ET AL., 1998A) .................. 18
FIGURE 4 - CMMI MODEL COMPONENTS (SEI, 2001A) ................................................................................ 25
FIGURE 5 - CLASSIFICATION OF IS RESEARCH METHODOLOGY (IIVARI, 1991) .................................... 36
FIGURE 6 - PROTOTYPING MODEL FOR CAT TOOL ..................................................................................... 39
FIGURE 7 - EIGHT MAJOR FUNCTIONS OF CAT TOOL .................................................................................. 41
Abstract

In the modern business world nowadays, most software organizations must continually improve all aspects of their business especially the improvement in their software processes in order to increase productivity and gain competitive advantage. One way of doing this is to conduct process improvement by implementing a disciplined and systematic approach to software process and quality improvement, such as the Software Engineering Institute's (SEI) Capability Maturity Model® (CMM®) (1993) or CMM Integration™ (CMMI™) (2001a), the ISO/IEC 15504 standard for software process assessment (1998) and the BOOTSTRAP process assessment model developed by BOOTSTRAP Institute (Bicego, et al., 1998a). These various popular process assessment approaches are normally used for conducting process assessment and improvement in any software organization. However, the success of process improvement largely depends on the effectiveness of the process assessment in a particular software organization. It is very important to conduct process assessment correctly and effectively in order to gain full benefits from the assessment. This is the main objective of the project reported here; that is to develop a tool to support software process assessment for any software organization that wants to improve their software and product quality. The development of this software process assessment tool, which is also known as CMMI Appraisal Tool (CAT Tool) is based on the continuous representation of CMMI™ process assessment model. This model is chosen for the CAT Tool because it is one of the emerging international standards for process assessment and there is a need to develop the tool for this model due to its rigorous rule of evidence as opposed to the requirements of other process assessment models. CAT Tool is a window-based process assessment supporting tool to record data collection from CMMI™ appraisals. CAT Tool is focused on collecting the objective evidence needed for CMMI appraisals in advance for any software organization that wants to prepare and conduct process assessment.
1. INTRODUCTION

Nowadays, most software organizations must continually improve all aspects of their business especially improvement of their software processes in order to increase productivity and gain competitive advantage in the modern business world. Dion (1993) reported that Raytheon has gained $7.70 avoidance of rework cost for every $1.00 invested in process improvement and has doubled their productivity within four years of investment. This is because the main aim of conducting process improvement is to improve productivity, increase product quality and gain competitive advantage (Gasston, 1996). To conduct process improvement, the software organizations need to implement a disciplined and systematic approach to software process and quality improvement, such as Software Engineering Institute’s (SEI) Capability Maturity Model® (CMM®) (1993) or CMM Integration™ (CMMI™) (2001a), the ISO/IEC 15504 standard for software process assessment (1998), and the BOOTSTRAP process assessment model (Bicego, et al., 1998a).

These various popular process assessment approaches are normally used for conducting process assessment and improvement in any software organization. According to Thayer and Thayer, (2001, p. 597-598 cited CMU/SEI-93-TR-25 1993), the software process assessment (SPA) is defined as “an appraisal by a trained team of software professionals to assess the state of an organization’s current software process, determine the high-priority software process-related issues facing an organization, and obtain the organizational support for software process improvement.”, and the software process improvement (SPI) is defined as “an action taken to change an organization’s software processes so that they meet the organization’s business needs and help it to achieve its business goals more effectively.” (2001, p. 598 cited ISO/IEC 1998). Based on these definitions, SPA is concerned with assessing the software processes in one organization against a process standard or framework such as the Capability Maturity Model for Software (SW-CMM), and SPI used by software organizations to improve their software processes in line with their business aims. However, the success of most process
improvement depends on the effectiveness of process assessment in particular software organization.

It is very important to conduct process assessment correctly and effectively in order to gain full benefits from the assessment. This is the main objective of the research project reported here; that is to develop a tool to support the software process assessment for any software organization that wants to improve their software and product quality. But, due to time constraints and the client’s requirements, this research project has been re-scope. The development of the process assessment tool prototype will purposely be developed based on the continuous representation of CMMI® process assessment model and will mainly focus on the support of the process assessment’s data collection function.

The CMMI model was chosen as a process assessment model for this prototype because it is one of the emerging international standards for process assessment that is specific to software assessment and there is a need to develop the tool for this model due to its rigorous rule of evidence as opposed to the requirements of other process assessment models. The prototype will be developed as a data collection tool because it is essential for any software organization that wants to conduct process assessment to prepare and collect objective evidences in advance of the actual process assessment. This will minimize the effort necessary for preparation by an external appraisal team, increase their efficiency and understanding, and improve accuracy of appraisal results. The objective evidence is the qualitative or quantitative information, records, or statements of fact pertaining to the characteristics of an item or service or to the existence and implementation of a process element, which is based on observation, measurement, or test and which can be verified (SEI, 2001b). The analysis of the objective evidence at an early stage is critical in setting the stage for the process assessment (SEI, 2001b).

This prototype will be developed as a window-based tool to record data collection from CMMI® appraisals, and the process definitions and guidelines for this prototype will be based on CMMI SE/SW Version 1.1 (SEI, 2001a) and SCAMPI (Standard CMMI® Appraisal Method for Process Improvement) Version 1.1 (SEI, 2001b).
The following sections of this part will describe the purpose of the report, the research problems, the research project significant, the research project potential outcomes, the research project approaches, and the overview of the report.

1.1 Purpose of the Report

The purpose of this report is to document the research project that has taken place. This research project is divided into two parts. The first part of this research project is to explore the different topics related to the research questions and the second part is to develop a CMMI<sup>sm</sup> Appraisal Tool (CAT Tool) prototype that is based on the research findings and client’s requirements.

1.2 Research Problems

In recent years, a number of software tools have been developed for supporting software process assessment such as CMM-Quest that is based on the staged representation of CMMI<sup>sm</sup> model, SPICE 1-2-1, which is based on ISO/IEC 15504 and BOOTCHECK developed for the BOOTSTRAP process assessment model (Bicego, et al., 1998a). Most of the process assessment supporting tools has been developed only for specific process assessment models. There is a need to develop a better tool that can support multiple process assessment models to ensure that the outcomes from software process assessment are more accurate and reliable for improving software process and product quality.

In addition, currently there is also no process assessment supporting tool that has been developed specifically for continuous representation of CMMI<sup>sm</sup> model because the rules of evidence for this model is very rigorous as opposed to the requirements of other process assessment models. Therefore, there is a need to develop a tool that can support process assessment for continuous representation of CMMI<sup>sm</sup>.
The major concerns for developing such a tool are as follows:

- What are the basic requirements to develop any process assessment supporting tool?
- What are the tool’s requirements that can be integrated with all process assessment models?
- What are the extra features needed in a tool that can support process assessment?

1.3 Research Project Significance

This research project is essential for any software organization that wants to self-assess their software process and product quality to ensure their organization can compete with others and gain competitive advantage from the improvement of their process and product quality. This research project is set out to show the importance of developing an effective tool that can integrate process assessment supporting tool’s functions and support multiple process assessment models for improving software process and product quality.

This research project is also important because the main objective of this project is to develop a tool to support continuous representation of CMMI\textsuperscript{sm} process assessment model and currently there is no process assessment supporting tool that has been developed specifically for continuous representation of CMMI model due to its rigorous rules of evidence. By developing such tool, the effort necessary for preparation by CMMI\textsuperscript{sm} external appraisal team will be minimized, their efficiency and understanding will be increased, and the accuracy of their appraisal results will be improved. At the same time, these will minimized the time and cost of conducting the CMMI\textsuperscript{sm} appraisal in any software organization.
1.4 Research Project Potential Outcomes

The potential outcomes from this research project can be categorized into two parts: theoretical and practical outcomes. The potential theoretical outcome from this research project is adding the body of knowledge about software process improvement and assessment supporting tools.

On the other hand, the potential practical outcomes are to be able to develop a better process assessment supporting tool that can be used by any software organization to prepare for any software process assessment and improvement in order to improve their software process and product quality. This research project can also be used as a guideline to develop a new generation of process assessment supporting tools in the future.

1.5 Research Project Approaches

There are different approaches have been adopted to further investigate the research problems. The first approach adopted is using the library research method to explore topics related to the project. The results of this method are provided in the Literature Review part.

Another method to find the solution for the research problems is by evaluating existing various process assessment supporting tools. By doing this evaluation, the key features of tool to support process assessment can be identified. The results from this evaluation are described in part 3 of this report.

The constructive technical development research methodology is adopted in addition to other two methods described above. This methodology can be used in the development of CAT Tool prototype. This methodology is outlined in the Research Methodology at the following part.
1.6 Overview of the Report

The remaining parts of this report will be organized as follows; section 2 discusses the literature review as a result from the library research method, section 3 provides the results from the evaluation of existing process assessment tools, section 4 describes the research methodology adopted to investigate further the research problems, section 5 outlines the system development methodology used to develop the CAT Tool prototype, section 6 covers the system requirements and specifications for the CAT Tool prototype, section 7 contains the prototype implementation and last section focuses on conclusions and future work for the future development of process assessment supporting tools.
2. LITERATURE REVIEW

This part of the report will review various topics related to the prototype development project, which includes software quality, software quality management systems, software process improvement, software process assessment and various software process assessment models.

2.1 Software Quality

It is difficult to define quality since it means different things to different people and within different contexts. Different software processes also require different quality attributes to be emphasized on for improvement. The Oxford Advanced Learner’s Dictionary (OALD, 1989), states that quality is ‘the degree of goodness or worth’. This definition is insufficient because it is too general and inappropriate for defining quality in software development.

A more appropriate definition of quality comes from the quality ‘gurus’; Deming, Juran and Crosby. According to Gillies (1992), Deming defined quality as “A predictable degree of uniformity and dependability at low cost and suited to the market.” and Juran defined quality as “fitness for purpose”. Juran’s definition is useful for combining the quality management ideas in software development. Crosby (1979) defined quality based upon “conformance to specification” and stated that “Quality is free”. His approach to quality is considered as a prevention approach that seems most appropriate for improving quality in the manufacturing sector.

Another definition of quality is from International Standards Organization (ISO). In ISO 8042 (1986, in Gillies, 1992), ISO defined quality as “the totality of features and characteristics of a product or service that based on its ability to satisfy specified or implied needs”. ISO associates quality with the ability of the product or service to fulfill
its function through the features and characteristics of the product. These features of quality might be useful when applied to software.

The definition by the Department of Defense (Gillies, 1992 cited DoD 1985) specifically refers to software quality. DoD defined software quality as "the degree to which the attributes of the software enable it to perform its intended end use", which combines the need to provide a good solution with the requirement to meet the intended purpose or user's satisfaction.

The definitions provided so far might all be useful to define quality. But, most of these definitions are conflicting with each other because of different perspectives on quality. Therefore, Garvin (1984) suggested five different approaches to the definition of quality as indicated in Table 1.

<table>
<thead>
<tr>
<th>No.</th>
<th>Definition of Quality</th>
<th>Description</th>
</tr>
</thead>
</table>
| 1.  | Transcendent          | - relates quality to innate excellence  
- classical definition of quality  
- impossible to quantify and difficult to apply in a meaningful sense to a large software project |
| 2.  | Product-based         | - economist’s view - the higher the quality, the higher the cost  
- views quality as a precise and measurable variable  
- apply to a large software project |
| 3.  | User-based            | - personal view of quantity and highly subjective  
- very hard to quantify  
- determine by the quality of match between the user's requirements and the designer's design |
| 4.  | Manufacturing-based   | - manufacturer's view - identify quality as "conformance to requirements"  
- easy to quantify  
- cost reduction approach by preventing defects earlier |
| 5.  | Value-based           | - ability to provide what the customer requires at a price that they can afford  
- quality product provides performance at an acceptable price or conformance at an acceptable cost  
- difficult to apply in practice |

Table 1: Five Different Approaches to the Definition of Quality (Garvin, 1984)

Most of the existing quality definitions fall into one of the approaches listed above. This is because different types of people will have different definitions of quality. For
example, the business-minded people views quality as product-based or user-based approaches because they believe that higher quality means better performance, enhanced features, and other improvements that increase cost. On the other hand, the manufacturing people used manufacturing-based approach because they view quality as conformance to specifications and they usually expect quality improvements to result in cost reductions. But, according to Garvin (1984), it is essential to shift one’s approach to quality as products move from design to market because the characteristics of quality must first be identified through market research (user-based approach); then these characteristics must be translated into identifiable product attributes (product-based approach); and lastly manufacturing process must then be organized to ensure that products are made precisely to the user’s specifications (manufacturing-based approach). The understanding of the various definition of quality described above is essential to understand other concepts related to software quality.

2.2 Software Quality Management System

One of important concepts related to the software quality is a Quality Management System (QMS). QMS can be defined as ‘the organizational structure, responsibilities, procedures, processes and resources for implementing quality management’ (ISO 8042, 1986). This definition shows that a QMS provides a systematic structure to ensure that the process is carried out in a formal and systematic way which aligned with the required standard.

Most of the QMS definitions and concepts emerged from the development of ideas of the three quality experts; Deming, Juran and Crosby. Deming had suggested fourteen points for management. Juran offered ten points for quality improvement compared to Crosby’s 14 steps to quality improvement. These three approaches are compared in Table 2 by Oakland (1989, in Gillies, 1992).
<table>
<thead>
<tr>
<th>Crosby</th>
<th>Deming</th>
<th>Juran</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>Conformance to requirements</td>
<td>Predictable degree of uniformity and dependability at low cost</td>
</tr>
<tr>
<td><strong>Senior management responsibility</strong></td>
<td>Responsible for quality</td>
<td>Responsible for 94% of problems</td>
</tr>
<tr>
<td><strong>Performance standard</strong></td>
<td>Zero defects</td>
<td>Many scales: use SPC, NOT zero defects</td>
</tr>
<tr>
<td><strong>General approach</strong></td>
<td>Prevention</td>
<td>Reduce variability: continuous improvement</td>
</tr>
<tr>
<td><strong>Structure</strong></td>
<td>14 steps</td>
<td>14 points</td>
</tr>
<tr>
<td><strong>SPC</strong></td>
<td>Rejects statistically acceptable level of quality</td>
<td>SPC must be used</td>
</tr>
<tr>
<td><strong>Basis for improvement</strong></td>
<td>A process, not a programme</td>
<td>Continuous: eliminate goals</td>
</tr>
<tr>
<td><strong>Teamwork</strong></td>
<td>Quality improvement teams: quality councils</td>
<td>Employee participation in decisions</td>
</tr>
<tr>
<td><strong>Cost of quality</strong></td>
<td>Quality is free!</td>
<td>No optimum, continuous improvement</td>
</tr>
<tr>
<td><strong>Purchasing</strong></td>
<td>Supplier is extension of business</td>
<td>Use SPC through strong co-operation</td>
</tr>
<tr>
<td><strong>Vendor rating</strong></td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td><strong>Single sourcing of supply</strong></td>
<td></td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 2: Comparison of Principal Ideas (Oakland, 1989)

This QMS basic concept can be implemented within software development through the adoption of a structured methodology. According to Gillies (1992), the structured methodology specified which procedures should be carried out first in the software development processes and a QMS should ensure that these procedures are carried out to the required standard. He also stated that the essential part of any QMS is the requirement for continual improvement to correct errors documented that is also known as a feedback loop, possibly suggested by Shewart (1981), but made famous by Deming as the ‘plan-do-check-act’ cycle of improvement as shown in Figure 1. This cycle of improvement can be adopted in software development for improving quality of software processes.
because it shows how to correct errors documented continuously through four stages of continuous improvement; plan, check, do and act.

Figure 1 - Deming’s Plan-Do-Check-Act Cycle of Improvement (Gillies, 1992)

Most of the approaches for systematically improving the software processes for software organizations have emerged from QMS concepts. Examples of such approaches are the Capability Maturity Model® (CMM®) (1993) or CMM Integration™ (CMMI™) (2001a) by SEI, the ISO/IEC 15504 standard for software process assessment (1998) and the BOOTSTRAP process assessment model (Bicego, et al., 1998a). These approaches apply the same principles of Total Quality Management (TQM) to the software process.

TQM described by Oakland (Gillies, 1992 cited Oakland 1989) as “A method of ridding people’s lives of wasted effort by involving everybody in the process of improving the effectiveness of work, so that results are achieved in less time”. Gunasekaran (1999 pp. 987-988 cited Oakland 1994) defined TQM as “an approach to improving effectiveness and flexibility of business as a whole. It is an essential way of organizing and involving the whole organization, every department, every activity, every person at every level”. In other words, TQM is the mutual cooperation of everyone in an organization to produce products and services, which meet the needs and expectations of customers in order to gain competitive advantage. But Kanji (Gillies, 1992 cited Kanji 1990) states “Quality is to satisfy customer’s requirements continually. Total Quality is to achieve quality at low
cost. Total Quality Management is to obtain total quality by involving everyone’s daily commitment”.

2.3 Software Process Improvement

According to Thayer and Thayer (2001, p. 598 cited ISO/IEC 1998), the software process improvement (SPI) can be defined as “an action taken to change an organization’s software processes so that they meet the organization’s business needs and help it to achieve its business goals more effectively.”. SPI is used by software organizations to improve their software processes in line with their business aims because most of the business is driven by market requirements for higher quality products, on-time delivery, and lower cost (Sanders, 2001). In order to ensure the highest return on investment, every software organization needs to improve their software processes.

By implementing SPI, any software organization can gain benefits from it. The benefits of SPI are shown in Table 3 (Sanders, 2001).

<table>
<thead>
<tr>
<th>Software Process Improvement (SPI) Benefits</th>
<th>Reduced</th>
<th>Improved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development costs because of less repetition and greater reuse.</td>
<td>Teamwork and more effective communication.</td>
<td></td>
</tr>
<tr>
<td>Rework because you identify and eliminate problems early rather than late.</td>
<td>Project staff start-up time because training is on defined processes.</td>
<td></td>
</tr>
<tr>
<td>Reliance on testing to ensure quality.</td>
<td>Predictability of budgets and schedules.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Tool usage to complement the processes used</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Project start-up time because of documented history of past projects.</td>
<td></td>
</tr>
</tbody>
</table>

Table 3: The potential benefits of software process improvement (SPI) (Sanders, 2001)

Paulk, et al., (1995) predicted the implementation of software process improvement in one software organization will improve control, predictability and effectiveness of the process in that particular organization. The survey by Herbsleb and Goldenson (1996) found that the other benefits of process improvement are an increase in product quality, productivity, staff morale, customer satisfaction, and ability to meet budget. These show
that it is worthwhile to invest in process improvement to improve the quality of the software process and product. However, the success of most process improvement depends on the effectiveness of the process assessment itself conducted in a particular software organization.

2.4 Software Process Assessment

Software process assessment (SPA) is essential for most software organization to continually improve all aspects of their business effectively in order to gain competitive advantage to compete with others. SPA can be defined as "an appraisal by a trained team of software professionals to assess the state of an organization’s current software process, determine the high-priority software process-related issues facing an organization, and obtain the organizational support for software process improvement." Based on these definitions, SPA is concerned with assessing the software processes in one organization against a process standard or framework such as the CMM® (SEI, 1993) (Thayer and Thayer, 2001, p. 597-598 cited CMU/SEI-93-TR-25 1993).

According to Hunter and Thayer (2001), SPA can be used for Capability Determination (CD) or Software Process Improvement (SPI). CD is used by a software organization to determine the capability of potential contractors and SPI is used by software organizations to improve their software processes in line with their business aims. Although, the original purpose of software process assessment was capability determination, but it is now used regularly for software process improvement.

The general SPI steps have been described by Sanders (2001) as the focused process improvement cycle as shown in Figure 3. It is based on the cycle of improvement as a more detailed version of general SPICE software process assessment steps. It includes some features of SPIRE improvement process, such as the evaluation of staff attitudes, and some specific details and examples for the improvement plan (Bicego, et al., 1998b in Sanders, 2001).
The focused process improvement cycle steps are as follows:

- **Step 1: Refining Business Goals as They Relate to Software**
  In order to improve business in software organization, there is a need to understand the organization’s business goals first. Then, identify what is required from software development processes for achieving those goals and the areas that need to be assessed and improved.

- **Step 2: Assessing Current Software Practices**
  There is important to assess the current software practices implemented in the organization before develop a plan for improvement. The people in the organization also need to understand that they are not the one that will be assessed but the processes in the organization.

- **Step 3: Evaluating Staff Attitudes**
  This is not part of every process assessment, but it helps to understand how people in the organization view the SPI process.
• Step 4: Developing a Focused Plan for Improvement
This is where the organization plans on how to conduct the improvement process. But firstly, the specific area and specific subject for improvement that will give the most payback need to be selected.

• Step 5: Implementing the Plan with States Goals for Improvement
The organization need to maintain its visibility of the improvement project in order to ensure they will gain benefits from process improvement. The organization also needs to ensure that everyone in the organization knows what they need and suppose to do to improve the processes in their organization.

• Step 6: Evaluating the Improvement Process and Results
After implementing the process improvement, the outcomes the improvement will be presented in terms of:
  o The summary of the outcome
  o The collection of data that enable the organization to show goals met or other useful information
  o Lessons learned in the improvement project, thus making sure that they are able and in the process of making the improvements
  o Recommendations for further improvement
  o Final cost statement

2.5 Software Process Assessment Models

A number of process improvement approaches to systematically improving the software processes have been developed in the last few years. According to Paulk (1998), there are three popular approaches to software process improvement or also known as software process assessment models. They are the SEI’s Capability Maturity Model® (CMM®)(1993), the ISO/IEC 15504 standard for software process assessment (1998), BOOTSTRAP by BOOTSTRAP Institute (Bicego, et al., 1998a) and CMMI® by SEI (2001). These approaches apply the same principles of Total Quality Management (TQM) to the software process. Table 4 below shows the chronological order and the
philosophies implied in each of these popular process assessment models (Wang and King, 2000). These models are described in the following section in a chronological order.

<table>
<thead>
<tr>
<th>Chronology</th>
<th>PAMs</th>
<th>Philosophy or Background Orientation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1987</td>
<td>CMM</td>
<td>To present a software project contractor's perception on the organizational and managerial capacity of a software development organization.</td>
</tr>
<tr>
<td>1993</td>
<td>BOOTSTRAP</td>
<td>To present a combined view of software lifecycle processes and quality system principles.</td>
</tr>
<tr>
<td>1998</td>
<td>ISO/IEC TR 15504 (SPICE)</td>
<td>To present a set of structured capability measurements for all software lifecycle processes, and for all parties such as software developers, acquirers, contractors and customers.</td>
</tr>
<tr>
<td>2000</td>
<td>CMMI</td>
<td>To present a software project contractor's perception on the organizational and managerial capacity of a software development organization through hardware and software integration.</td>
</tr>
</tbody>
</table>

Table 4: Chronological Order of Process Assessment Models (Wang and King, 2000)

2.5.1 CMM

The Capability Maturity Model (CMM) was developed in the mid 1980s by the Software Engineering Institute’s (SEI’s) Software Process Program led by Watts Humphrey (1989). The CMM was launched at the SEI’s Affiliates Symposium held in Pittsburgh in June 1987. It is based on two questionnaires, a maturity questionnaire (concerned with the maturity of the software process) and a technology questionnaire (concerned with the extent to which advanced technology was used in the process) (Hunter and Thayer, 2001).

According to Paulk et al. (1995), CMM concentrates on the process. In the CMM, all processes are documented as procedures, standards, methods and organizational policies. CMM-based appraisals probe the implementation of the process through the process’s audit trail in the document reviews and interviews. This approach describes what a process should address rather than how the process should be implemented.
The framework of CMM is based on the principles of product quality. It has five maturity levels; Initial, Repeatable, Defined, Managed, and Optimizing levels (SEI, 1993). These maturity levels are characterized through the activities performed by the organization to improve the software process, by the activities performed on each project and by the resulting process capability across projects. The predictability, effectiveness and control of an organization’s software processes are improving as the organization moving up these five maturity levels. According to Paulk et al. (1995), there are different criteria that needed for organization to reach different levels of maturity. For organization to reach Level 1, they should focus on to improve its software process. At Level 2, the organization should have managed to produce:

- Requirements management
- Software project planning
- Software project tracking and oversight
- Software subcontract management
- Software quality assurance
- Software configuration management

For organization to be at Level 3, they should have managed to produce:

- Organization process focus
- Organization process definition
- Training program
- Integrated software management
- Software product engineering
- Intergroup coordination
- Peer reviews

At Level 4, the organization should have managed to produce:

- Software quality management
- Quantitative process management

For the organizations that want to be at Level 5, they should have managed to produce:

- Defect prevention
- Technology change management