الجامعة الاسلامية العالمية ماليزيا
INTERNATIONAL ISLAMIC UNIVERSITY MALAYSIA
يونيۏرسيتي اسلام انتارابڠسا مليسيا

MBA PROJECT PAPER

Software Process Improvement Framework - From Concepts To Capabilities

Submitted to the

Management Center

International Islamic University Malaysia

By Tan Chuan Cherng

In partial Fulfillment of the Requirements

For the degree of Master of Business Administration

APRIL 2001

853211 main

16/7/03 aqr

t
[illegible]
16.16
[illegible]
71615
2001

ACKNOWLEDGEMENT

My first thanks go to Ms. Ong Geok Lwee of SEMA for initiating corporate sponsorship and continuous moral support throughout this MBA program. Apropos of that, I am grateful to many current and former associates, particularly, Mr. Tee Kee Ming, Mr. Steve Robinson, Ms. Tay Piak Tiam and Mr. Keith Tanner, who generously share their insights by extending candor guidance and valuable assistance towards the fruition of this paper.

My warmest gratitude goes to my supervisor, Dr. Farooq Ahmad, for his constructive comments and conscientious advice in making this paper feasible.

My tremendous appreciation goes to IIUM Management Center teaching faculty and staff, especially Prof. Mohd. Saeed, Dr. Azaddin, Dr. Kameel, Dr. Rizwan Prof. Mahfooz, Prof. Mokdad, Dr. Azmi, Dr. Ubai and the class of MBA 2001 which make my learning experience intellectual stimulating and rewarding, sparked with thoughtful interactive exchanges and spiced with impromptu good sense of humor.

For their toleration and inspiration, I thank my beloved family members for their relentless patience and understanding during the course of this MBA program.

To all I am indebted. Without their support and encouragement, completion of this program and project paper would not have been possible.

APPROVAL PAGE

Title of Project Paper:

Software Process Improvement Framework -

From Concepts To Capabilities

Submitted by Tan Chuan Cherng (G9816555)

The undersigned certify that the above candidate has fulfilled the conditions of the project paper requirement in partial fulfillment for the degree of Master of Business Administration.

[illegible] 28/07/01

....

Dr. Farooq Ahmad

Assistant Professor

Department of Business Administration

Kuliyyah of Economics & Management Sciences

International Islamic University Malaysia

Date: 15 Apr 2001

TABLE OF CONTENT

ABSTRACT ---------- **1**

1. INTRODUCTION ---------- **2**

2. CONCEPTS: THE 4PS OF SOFTWARE ENGINEERING PRINCIPLES ---------- **3**

3. PRACTICES ---------- **4**

3.1 CODE READABILITY 4

3.2 REVIEWS AND INSPECTIONS 4

3.3 REUSABILITY 5

3.4 TESTING – TRACING FOOTPRINT OF DESIGN ARCHITECTURE 5

3.5 AUTOMATED SOURCE CODE CONTROL 6

3.6 USER INVOLVEMENT 7

3.7 INCREMENTAL DEVELOPMENT 7

3.8 CHECKLISTS 7

4. PROCESSES ---------- **8**

4.1 SEI'S CAPABILITY MATURITY MODEL FOR SOFTWARE (SW-CMM) 8

4.2 ISO9001 10

5. PERFORMANCE ---------- **12**

5.1 METRICS AND MEASUREMENTS 12

6. PEOPLE ---------- **13**

6.1 LEARNING ORGANIZATION 13

6.2 KNOWLEDGE MANAGEMENT (KM) 14

7. CAPABILITIES – REALIZATION OF SOFTWARE ENGINEERING PRINCIPLES ---------- **15**

7.1 PROJECT MANAGEMENT PACK 15

7.1.1 Project Planning *15*

7.1.2 Project Start-Up and Closure Checklists *15*

7.1.3 Programming Guideline *16*

7.1.4 Management by Risk Checklist *16*

7.1.5 Project Control Guidelines *17*

7.1.6 Project Review Procedures *22*

7.1.7 Project Debrief *25*

7.2 PERFORMANCE AND METRICS 26

7.2.1 Performance Index *26*

7.2.2 Management Dashboard *26*

8. CONCLUSION ---------- 28

REFERENCES ---------- 29

APPENDIX 1 – PROJECT QUALITY PLAN TEMPLATE ---------- 33

APPENDIX 2 – PROJECT START-UP CHECKLIST ---------- 39

APPENDIX 3 – PROJECT CLOSURE CHECKLIST ---------- 47

APPENDIX 4 – PROGRAMMING GUIDELINE ---------- 51

APPENDIX 5 – MANAGEMENT BY RISK CHECKLIST ---------- 52

APPENDIX 6 – REVIEW FORM ---------- 60

APPENDIX 7 – PERFORMANCE INDEX ---------- 61

APPENDIX 8 – MANAGEMENT DASHBOARD ---------- 62

Abstract

Long haunted issues in software development are discussed in tandem with tried and true recommendation from software practitioners. With the aim of improving existing software process performance and product capabilities, SEMA is engaging pertinent core values, embracing by establishing robust guidelines, and with further extension via continual improvement in sight.

1. Introduction

With software becoming the new physical infrastructure of the information age, underpinning global economic activities as change catalysts & opportunity drivers [1], problems long associated with software creation, i.e. unpredictable quality, schedule slippage and uncontrollable costs are looming challenges and stirring concerns [2]. These difficulties arise mainly because of the intertwining of factors such as software intangibility crafted by manual labor, subtle & multi-dimension requirements that often not fully understood in sufficient detail and improper estimation & change in requirements which will lead to scope creeping [3].

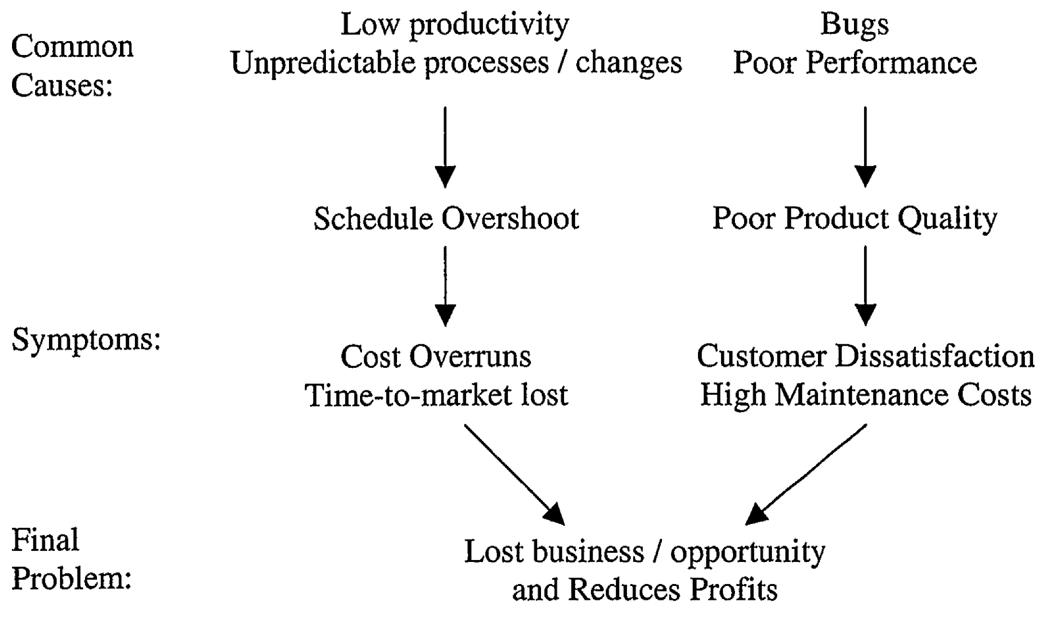

Figure 1: Long distressing problem with associated symptoms and causes

2. Concepts: The 4Ps of Software Engineering Principles

For many software practitioners, excellence in software engineering have always been distantly elusive, however, various effort have been initiated by voluntary group and governance to outline body of knowledge and promote good software engineering principles [4,5]. A profound similarity can be found among those guidelines, i.e. the distinguishing elements of **best practices**, **flexible processes**, **performance driver** and **people involvement**. These have been embraced by SEMA in responding to the journey of quality fulfillment.

3. Practices

Practices are accumulated from crucible of the real world and product of years of hard-won experience. These essential ingredients constitute a set of highly leverage discipline which is the starting point for structuring and deploying an effective software engineering processes [6].

3.1 Code Readability

With project members spending a lot of time writing, reviewing and revising the projects source code. Maintenance programmers spend more than half of their time figuring out what source code does. Their job can be much easier with emphasis on good layout, careful variable, function & class name and meaningful comment [7].

3.2 Reviews and Inspections

Started by Gerald Weinberg's concept of ego-less programming, i.e. the idea that no matter how smart a programmer is, reviews will be beneficial [8]. With right instrumentation and appropriate training, they are powerful techniques for defect detection, especially for upstream activities that always cost lesser to correct as compared when these defects travel downstream. For instance,. one sentence of requirement will be represented with five pages of design diagrams and coded into 500 lines, which lead to few dozens of test cases. Reviews and inspections help foster broader understanding and learning, leading to better code [9]. However, they can also be abused, e.g. where people become indifferent to the skill set of the review team, or when they don't bother with testing.

3.3 Reusability

Code reusability, or any piece of job for this matter can avoid unnecessary "reinvent of wheel", limit effects of abrupt changes and improve performance by reducing complexity and improving portability in writing routines [10].

3.4 Testing – Tracing Footprint of Design Architecture

Modern software development produces the architecture first, followed by usable increments of partial capability, and then completeness. With the noble intention of ensuring check and balance, back and forth the system characteristics, a.k.a. (also known as) expected functionality [11]. Instead, testing has been considered as the black sheep of quality assurance (QA) practices as far as development speed is concerned, testing discovers that product quality might be too low for it to be released, hence schedule overshoot and cost overrun. Testing thus becomes the messenger that delivers bad news [12]. Moreover, the fact that testing is the last item on critical path making it so vulnerable to shortchanging.

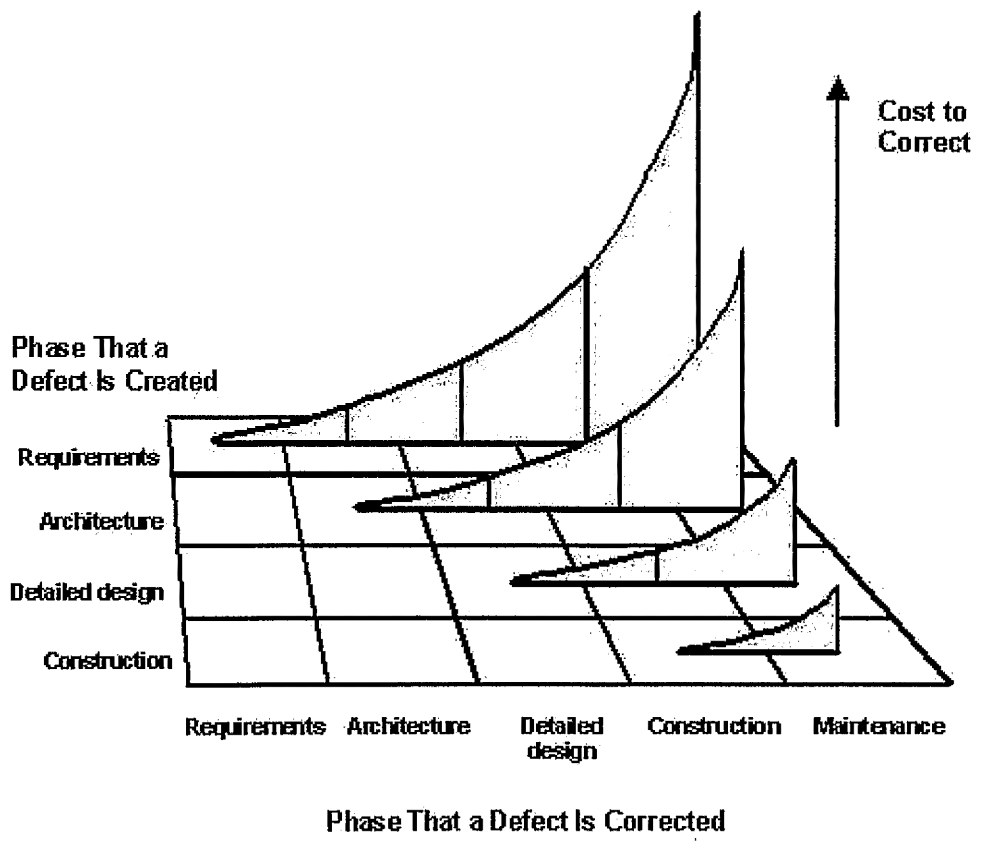

Figure 2: The longer a defects remains undetected, the more expensive it becomes to correct

3.5 Automated Source Code Control

Automated source code control, e.g. software configuration management, SCM) takes care of mountainous housekeeping details and voluminous manual paper work, pushing programming efficiency to a greater height. Problems such as inadvertent overwriting other work and off-site backup from master source, making SCM indispensable enabler in today's software development work [13]. SCM creates virtually no overhead and pays for itself the first time anyone need to retrieve the version written yesterday.

3.6 User Involvement

Techniques such as Joint Application Development (JAD) and user interface prototyping that beyond the limitation set by pure paper documentation are the effort to overcome the number one cause of software project failure, i.e. requirements / expectation problems [14].

3.7 Incremental Development

Horror stories of software meltdowns during integration phase has prompted software professionals to bring in the ideas of component-based development, daily build, spiral life-cycle model and RAD (Rapid Application Development). Requirement and design flaws are detected, isolated and resolved earlier in the life-cycle, i.e. less costly [15], avoiding the big-bang integration risk, provide steady evidence of progress, keep quality levels high and help boost team morale as everyone can "see" that the software works [16].

3.8 Checklists

Checklists are an often-overlooked, low-tech development tools, but they are useful in many ways as they are created from experience and hence inherently practical. Use them at design stage to ensure it takes account of all relevant consideration, use them at code review helps reviewers catch the most common problems and prevent from careless (or sometime needless) mistake during last minute rush for software release [7].

4. Processes

Process can be defined as set of interrelated or interacting activities which transforms inputs into desired outputs [17]. Process is context-sensitive with ultimate goal shaping the chain of process in term of scope and the size of end-product [18]. Hence, professional judgement is necessary in sensibly deploying any model embraced [19,20], i.e. the process model (simplified vies of real world) must show where flexibility is permitted and where deviation is allowed [21], in order not to stifle creativity of developers but cushioned them with reliable processes [22].

The concept of process improvement, which developed within the circle of quality movement, requires first that the existing process be stabilized statistically (repeatable) by eliminating the special causes of variation. It then become predictable (i.e. false alarms and missed signals can be differentiated), and its capabilities become accessible to analysis and improvement [23]. Continuous process improvement occurs when the cycle of stabilizing, assessing, and improvement a given process becomes internalized or institutionalized [24]. Two citations of renowned software process models are discussed below [25]:

4.1 SEI's Capability Maturity Model for Software (SW-CMM)

SEI's (Software Engineering Institute) SW-CMM is a common sense application of process management and application of Total Quality Management (TQM) within software development and maintenance environment [26]. Various kudos as well as brick-bats received since the version 1.1 release in 1993. Among the critics are the omissions of soft people issues such as motivation, productivity and possible dysfuntional behaviors [27].

The maturity of SW-CMM is the extend to which a specific process is explicitly defined, managed, measured, controlled and effective. It can be organized into five maturity levels:

1) **Initial** – The software process is characterized as ad hoc, and occasionally even chaotic. Few processes are defined, and success depends on individual effort and heroics.

2) **Repeatable** – Basic project management processes are established to track cost, schedule, and functionality. The necessary process discipline is in place to repeat earlier success on project with similar applications.

3) **Defined** – The software process for both management and engineering activities is documented, standardized, and integrated into a standard software process for the organization. All projects use an approved, tailored version of the organization's standard software process for developing and maintaining software.

4) **Managed** – Detailed measures of the software process and product quality are collected. Both the software process and products are quantitatively understood and controlled.

5) **Optimizing** – Continuous process improvement is enabled by quantitative feedback from the process and from piloting innovative ideas and technologies.

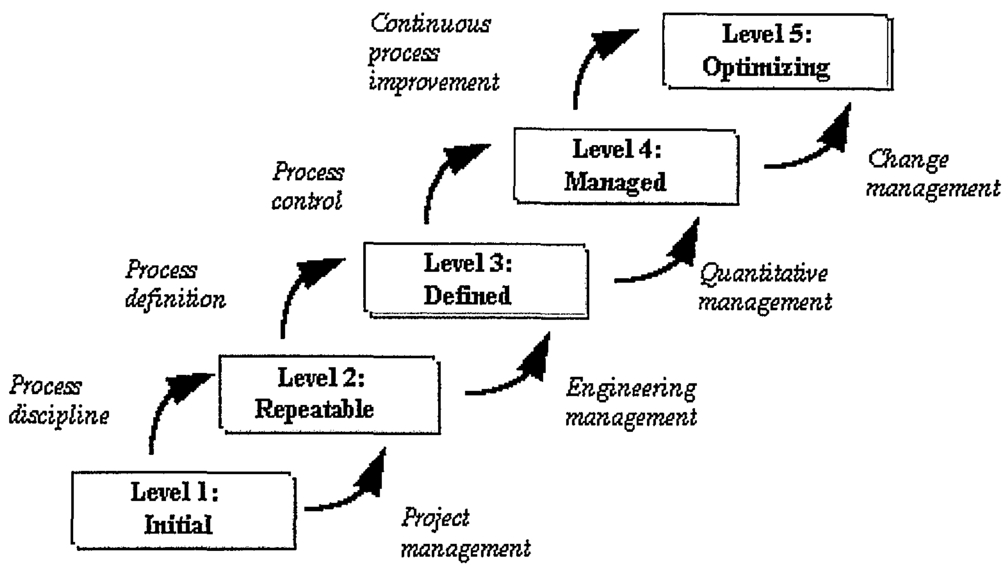

Figure 3: SW- CMM version 1.1 (1993)

4.2 ISO9001

The ISO9001:2000 Quality Management System is the much polished process model compared with previous version, the focus on driving measurable performance towards achieving and sustaining customer satisfaction [28]. Triggered by "customer requirement" and end with ultimate aim of "customer satisfaction", underpinning by continual improvement [29], intended to help an organization respond to the changing needs of its customers, while stimulating the efficiency and improving its competitive position [30].

The process approach emphasizes the importance of

a) understanding and fulfilling requirements,

b) the need to consider processes in terms of added value,

c) obtaining results or process performance and effectiveness, and

d) continual improvement of processes based on objective measurement.

In addition, the methodology known as "Plan-Do-Check-Act" (PDCA) can be applied to all processes. PDCA can be briefly described as follows:

Plan:	establish the objectives and processes necessary to deliver results in accordance with customer requirements and the organization's policies.
Do:	implement the processes.
Check:	monitor and measure processes and product against policies, objectives and requirements for the product and report the results.
Act:	take actions to continually improve process performance

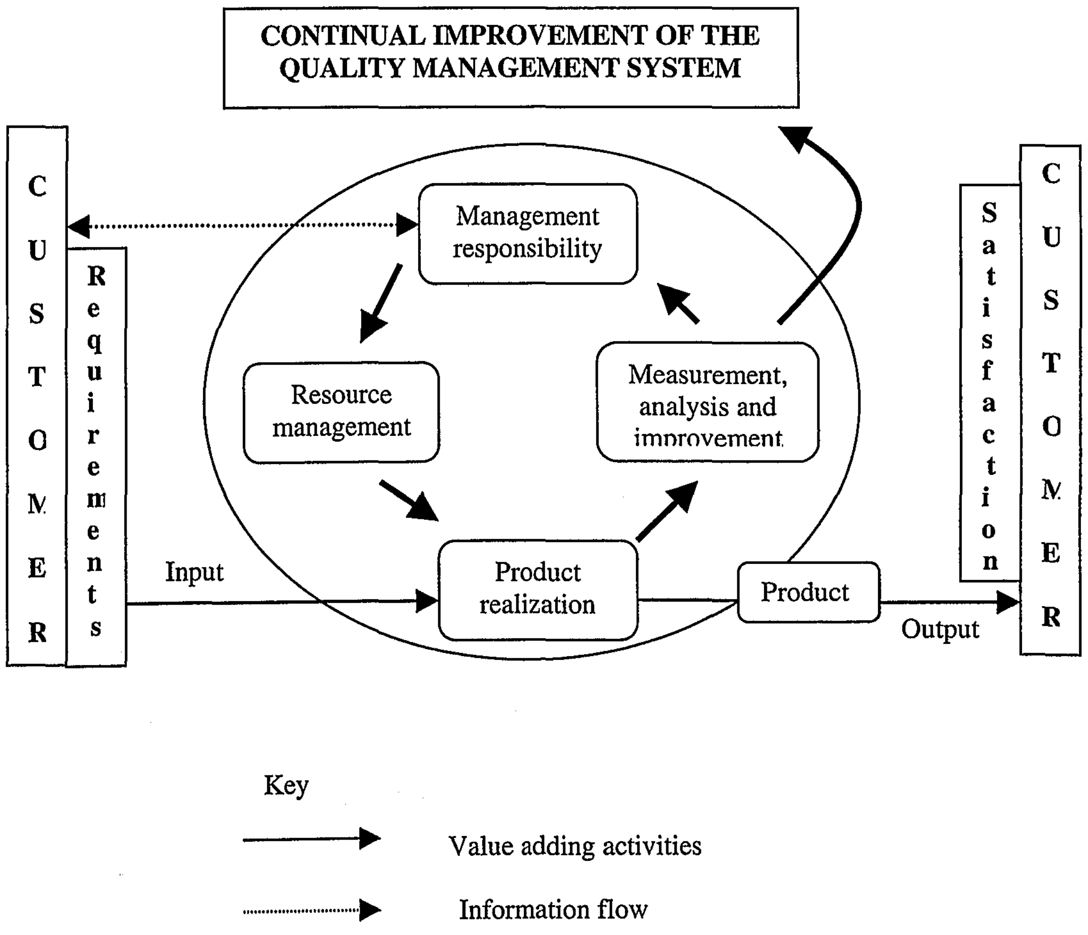

Figure 4: Model of a process-based quality management system ISO9001:2000

5. Performance

The widely trumpeted financial measures such as profitability reflect only a partial, and frequently a short-term performance of a firm [31]. Other non financial indicators, e.g. customer satisfaction and retention, response time, market share, product development and quality and employee morale proved to be too important to ignore in a more reflective broad base performance indicators for investors' inference [32]. The continuos monitoring of these indicators will provide greater granularity (i.e. more detail, better insights) on the overall health of company [33].

5.1 Metrics and Measurements

Though seemingly synonymous, measurement (pure degree / extent of any dimension) is the basis to derive metrics, i.e. meaningful indicators. Metrics characterize the processes quantitatively by representing what ought to be (baseline/threshold), has been and can be achieved in term of ratios or percentage [34].

Some pointers on practical measurements collection and metrics establishment are [35]:

(1) Measure what matters, i.e. one that energize people to focus effort in improving things that contribute to the success of organization

(2) Keep it simple, i.e. simple to understand, operate and act upon when there is deviation

(3) Don't seek perfection on first go – just do it!

6. People

People is the pivotal role of any productivity exercise, as evidence from the mushroom of studies related to soft skill such as leadership, participation, motivation and behavioral [36, 37, 38, 39, 40]. Among others are learning organization and knowledge management.

6.1 Learning Organization

The ability to learn faster than competitors has been touted as the only sustainable competitive advantage, human as the only resource that can assimilate knowledge and improve upon application of it is central of learning organization. As advocated by Peter Senge [41], learning organization is the realization of that concept by practicing this five disciplines [42]:

(1) Systems Thinking - the ability to analyze in terms of interdependencies and patterns of change in the larger whole;

(2) Personal Mastery - the commitment to one's vision as well as truth, energized by creative tensions (gaps between vision and reality);

(3) Mental Models - continuous testing the validity of deep-seated internal image of how the world works and improve upon it;

(4) Shared Vision – forge a commonality that gives a sense of purpose and coherence in turning cohesive action into over-arching desires accomplishment;

(5) Team Learning – collaborative and concerted effort by complementing each other's strengths and compensating for each other's weaknesses.

6.2 Knowledge Management (KM)

The concept of KM emerged as computer users sought to turn the deluge of information /experience into meaningful knowledge to be shared, with the hope to enhance their wisdom. KM is a collection of indexing, classifying, and information-retrieval technologies, e.g. distributed technologies can be used for disseminating information and creating virtual forums that connect experts & XML with inherent indexing capabilities has been seen as prime candidate for deploying KM. Coupled with methodologies to achieve intended results, e.g. in the area of collaboration such as customer relationship management (CRM), business processes and etc [43].

By embedding learning capabilities, i.e. tractable organization memory, companies can reduce the information overload of their employees and improve the consistency and effectiveness f knowledge used throughout an organization [44].

7. Capabilities – Realization of Software Engineering Principles

While abundance of process models, touted best practices, tried performance setting and people management theories available in the market, pragmatic approaches, together with enabler tools have been tailored for SEMA needs. Among others, Project Management Pack addressing both the Processes and Practices whereas Performance Index and Management Dashboard look after Performance and People.

7.1 Project Management Pack

With the organization structure geared towards System Integration (SI) business, project management framework became a pivotal component in ensuring the successful delivery of project to customer. A project management pack has been established based on input from personnel involved in managing project.

7.1.1 Project Planning

Planning forms the most essential part of any project, as project manager will determine and fine the most suitable process flow and practices that don't stifle creativity but in the mean time ensure constancy in purpose. Please refer to Appendix 1 for an outline of a Project Quality Plan:

7.1.2 Project Start-Up and Closure Checklists

These checklists (Appendix 2 & 3) offer guide to **identify priority, effort** and **specific deliverables/completion criteria** associated with activities essential for success. In essence these state what information:

- already exists in some form or other within SEMA (e.g. quality standards)
- needs to be generated by the project manager and/or staff working on the project (e.g. project specific standards)
- to be solicited / supplied by the customer
- to be gathered from subcontractors and suppliers
- to be considered during closure of project, e.g. acceptance criteria, hand-over procedures, warranty consideration, requirements on archiving and etc.

7.1.3 Programming Guideline

This guideline (Appendix 4) helps programmer to write efficient and maintainable code.

7.1.4 Management by Risk Checklist

A risk is defined as anything that threatens the successful completion of the project. SEMA adopts "Management by Risk" approach as an aid to project management. In a nutshell, the Project Manager should consider all areas of the project from the perspective of risk in order to apply risk management on them. This technique has the following aims:

- **Identify, prioritize** (based probability and severity) and where possible quantify the risk.
- Reduce the likelihood of risks maturing by active management.
- Develop and **document contingency** for each risk (i.e. what to do if the risk matures, or why no risk reduction plan is needed).

- Risks are **monitored** and re-assessed (if condition warrant, e.g. a change in priority, new contingency / action, closure of risk etc) within the framework of the project management and progress meeting.
- Risk will be **closed** when it has been (1) eliminated (2) matured (3) replaced by other risks.

Risk may be identified: (1) from checklist, as below; (2) based on project personnel observation and past experience; (3) during progress meetings; and / or (4) while evaluating change request.

Risk can be categorized either as internal (within SEMA control, e.g. assignment of someone with little previous experience) or external (e.g. supply and delivery of customer machine by third party). External risks should always be shared with the customer. From SEMA past experience, this approach to risk management significantly increases the likelihood of project success. By focusing management attention on the significant threats, it encourages well-defined forward planning to counter them, please refer Appendix 5 for the detail.

7.1.5 Project Control Guidelines

Type of documents

- **Controlled document** – updateable and is likely to be re-issued. E.g.: proposals, contracts, project related plans, project-produced standards and procedures requirements and specifications documents, design documents, test documentation and all documents to / from customer.

- **Uncontrolled document** – usually not to be re-issued. E.g.: email, memos, meeting minutes, progress reports, etc

Procedures (internal origin)

1. **Initiate** - project team member or customer who wants to make changes or enhancement to the released Project Document raise a **change request** (e.g. in memo format) or **mark-up** on the released document and submit it to the respective Project Manager/leader for review and approval.
2. **Review and approval** - when the request has been agreed and approved, the relevant document owner (author) can proceed to make the changes.
3. **Revision** - when the changes have been made, the document owner shall update the amended document(s) to the Project Manager who keep the latest master copy of the controlled project documents.
4. **Information on changes** - the version history should be updated in the "Document Control" page. Where appropriate, document author can opt to register the changes via methods such as redlining, left margin bar, highlighted text, strikeout etc. can be used to differentiate between the current and previous version. Whatever the method used, it should be stated in the "Document Control" page.
5. **Distribution** – Project Manager needs to ensure the updating process taken place for these documents for all the distributed control copy, i.e. removal of old version and replacement with latest version.
6. **Obsolete** - any obsolete document (including the master copies) that retained for record purposes should be marked clearly the word "OBSOLETE", with slash " / " or double slash " / / " printed on the front of the document.