Lulu Zhang

The Software Test Improvement Model in Practice

Master's thesis in
Information Systems Science
1.5.2005

University of Jyväskylä
Department of Computer Science and Information Systems
Jyväskylä

ABSTRACT

The Software Test Improvement Model In Practice
Lulu Zhang
Jyväskylä, University of Jyväskylä, 2005. Pp. 110.

The research in this thesis analyses how to improve the software testing work by using the test models. The Test Improvement Model is picked as the investigation object. It presents the theoretical background for the study, through the in-depth literature review in the areas of software testing and software testing models. The literature review reveals that the testing is a critical process during the software development to achieve better software quality.

To improve the product quality, the testing process has to be improved first. However, without the reasonable software testing improvement model, it is hard to improve the software testing process. Therefore, good software testing model has to be chosen to guide the software testing process improvement work.

The following topics are discussed in the literature review:

- Which testing models currently exist
- Which model would be the most practical to improve the testing process
- What are the limitations of the other testing models

The several research questions raised: 1) How to improve the testing work? 2) Is the Test Improvement Model practical in Finnish company? 3) Is there anything can be adjusted in the Test Improvement Model in the Finnish company context?

The study addresses these research questions by 1) taking literature review, conceptualizing the model and proposing the possible adjustments; 2) conducting the empirical investigation by means of case study.

After the case study, this thesis concludes that the Test Improvement Model is practical to implement in Finnish companies. However, the necessary supplements might be needed to add into the current Test Improvement Model.

KEYWORDS

software testing, test improvement, test improvement model, Test Improvement Model, model, practice

ACKNOWLEDGEMENT

This thesis could not have been written without Dr. Markku Sakkinen, who not only served as my supervisor, but also encouraged and challenged me throughout my whole study phase. He gave me proactive supervision of my research work, decisive guidance and great attention. He shared his knowledge with me without any selfishness.

I would like to thank the Faculty of Information Technology for giving me the opportunity to study at University of Jyväskylä and conduct the research. I am grateful to all the people of the Faculty of Information Technology who helped me during my study.

I am also thankful to Prof. Timo Käkölä, who was the head of my study program. He inspired and helped me with my studies in Finland. Dr. Zheying Zhang, who lectured two excellent courses I attended and shared her knowledge when I wrote on my thesis. Dr. Nazmun Nahar, who gave me two interesting courses in software business and plenty of tips on how to survive in a foreign country.

My special thanks go to Botnia Hightech (BH) Oy. Without BH's cooperation and support, the case study could not been done so smoothly. I would like to repeat my sincere thanks to Mr. Pekka Hietalahti and Mr. Jani Rutanen, for their assistance in arranging the case study interviews, sharing their experience with me and spending time on my thesis. I am also grateful to Dinh Tran for his careful and patient proofreading of my thesis.

Last but not least, I would like to deeply thank my parents, Chunfeng Zhang and Zhili Peng; my sister, Dr. Zheying Zhang and her family; my uncle, Dr. Zhiqi Peng and his family for their constantly encouragement, suggestions and the endless family love and support. Finally, I would like to show my appreciation to my boyfriend, without his company, this path would be boring to walk alone.

TABLE OF CONTENTS

1. INTRODUCTION 7
1.1 Background 7
1.2 Software Testing Concept 10
1.3 Software Testing-Related Product Quality 12
1.4 Research Motivation 14
1.5 Research Objective 17
1.6 Structure of the Thesis 18
2. PROBLEM AND QUESTION 19
2.1 The Main Reasons for Company to Have the Software Testing Model 20
2.2 Research Problems and Questions 21
3. LITERATURE REVIEW 23
3.1 Foreword 24
3.2 Some Classical Software Testing Models 25
3.2.1 Beizer's Testing Mental Model 26
3.2.2 Gelperin and Hetzel's Model 29
3.2.3 The Software Testability Maturity Model (TMM) 31
3.2.4 The Test Improvement Model 35
3.3 Software Quality and the Improvement 37
3.4 Summary 37
4. THE TEST IMPROVEMENT MODEL 38
4.1 Levels and Goals of the Test Improvement Model 39
4.2 Key Areas of the Test Improvement Model 42
4.2.1 Organization 43
4.2.2 Planning and Tracking 48
4.2.3 Test Case 52
4.2.4 Testware 55
4.2.5 Reviews 57
4.3 The General Assessment of the Test Improvement Model 58
4.4 Summary 60
5. RESEARCH APPROACH 60
5.1 Background of the Chosen Research Approach 61
5.2 Research Design 63
5.3 Data Analysis 65
5.4 Quality of the Study Design 66
6. CASE STUDY— THE TEST IMPROVEMENT MODEL USABILITY AND IMPROVEMENT 68
6.1 Background 68
6.2 Case Study Phases 68
6.3 Case Data Analysis 70
6.3.1 Discussion of the Research Questions 71
6.3.2 Organization 74
6.3.3 Planning and Tracking 83
6.3.4 Test Case 87
6.3.5 Testware 90
6.3.6 Reviews 91
6.3.7 Two Additional Key Areas 92

6.4 Summary 96
7. CONCLUSION 96
7.1 Major Contributions 97
7.2 Limitation 98
7.3 Further Study 100
REFERENCES 102
APPENDIX—THE QUESTIONNAIRE GUIDE 106

LIST OF FIGURES

FIGURE 3-1. Basic Test Process Data Flow (1988) 30
FIGURE 3-2. TMM Levels (Hines, 2001) 33
FIGURE 4-1. Test Improvement Model Structures 43
FIGURE 4-2. Team Model (Ahonen et al, 2003) 45
FIGURE 4-3. Interdepartmental Model (Ahonen et al, 2003) 46
FIGURE 4-4. Resource Pool Model (Ahonen et al, 2003) 46
FIGURE 4-5. Traceability Example Model 51
FIGURE 4-6. Test Case Reuse Based on Requirement Relationship 54
FIGURE 4-7. Assessment Work Flow 59
FIGURE 6-1.Testing Planning and Tracking Working Procedure 86

LIST OF TABLES

TABLE 3-1. Tester's Mental Phases (Beizer, 1990) 27
TABLE 3-2. Test Improvement Model Maturity Level (Ericson et al, 1997) 36
TABLE 3-3. Test Improvement Model Strategies (Ericson et al, 1997) 36
TABLE 4-1. Activities of Organization KA (Ericson et al, 1997) 44
TABLE 4-2. Activities of Planning and Tracking KA (Ericson et al, 1997) 49
TABLE 4-3. Activities of Test Case KA (Ericson et al, 1997) 53
TABLE 4-4. Activities of Testware KA (Ericson et al, 1997) 56
TABLE 4-5. Activities of Reviews KA (Ericson et al, 1997) 57
TABLE 6-1. Investigated Organization 74
TABLE 6-2. Investigated Planning & Tracking 84
TABLE 6-3. Investigated Test case 88
TABLE 6-4. Investigated Testware 90
TABLE 6-5. Investigated Reviews 91

1. INTRODUCTION

In the recent decades, software testing technologies have become very popular. The usage of matured technology model can effectively facilitate the technology development. This thesis studies how to improve the software testing process by using the testing improvement model, based on the current working situation; i.e. which areas should the professionals focus more on.

The first chapter introduces the software testing technologies and the current situation of software testing models. It will explain why software testing is so hard to accomplish and what aspects are directly related to software testing. The research motivations and objectives are also declared in this chapter.

1.1 Background

All software products need to be tested before they can be delivered to the customers. The testing methodology got awareness by professionals at the same time when the software industry emerged. However, while the software development technology quickly advanced, the software testing technology did not develop as well as expected (Coulter, 1999).

What was the reason hindering the development of software testing? The early investigation (Beizer, 1990) shows that in software companies, the software engineers disliked software testing because they did not want the testers to create extra test code to test the original code which they had worked on so hardly. So, from the mental side, the testing work is not a likable job. If professionals did not like the job, apparently, they did not make effort to develop it. Experienced professionals announced that if the software companies do not have any software testers, the software engineers have to carry out the responsibility to test their own code. The task to implement the extra

testing code and to prove that the software works correctly is certainly a hard job for the programmers. It is therefore worth to doubt the quality of the end software, since the application can be either partly tested or not tested at all. As a result, the software has flaws, which causes the software product quality to be below the expectations. The lack of motivation to carry out software testing heavily hinders the development of software testing technology.

On the other hand, all of the software venders have to face the fact, that when the software industry develops, the software complexity and size expands quickly at the same time. This will cause the market and the customers to require higher software quality to satisfy their needs (Gelperin and Hetzel, 1988). To achieve higher software quality, requires better-covered software testing, in all the aspects of the software product's usability. In such high-competence marketing environment, the knowledge of how to produce high quality software in a relatively short time becomes the main strategy to keep up in the business world. In order to make profit, no matter if the company likes it or not, the software testing has to be built in seriously. Companies without software testing, have to implement the software testing or outsource the testing to external partners; whereas companies, which have already implemented software testing, have to try to efficiently and visually improve their testing to better meet the market needs.

The quality of the final product, and the development costs are both involved in the testing process. According to Burnstein et al (1996), software testing is the most important cost factor in the software development process. In typical cases, software testing can constitute up to 50-60% of the total effort spent to create a software system. Therefore, to some extent, software testing is not only the process of software development; it weightily influences the software product quality and likewise the profit, and is therefore related to the business performance as well. As a result, a successful software testing process ensures higher quality of the software product, retains the software development costs, lowers the business risks and enhances the value of the software product. For example, if the company upgrades the manual testing to

automatic testing, the work efficiency can be significantly increased, and at the same time the salary expenses decreased.

Software testing affects the product quality and the product cost. The involved two issues are rather important for the software companies. But, what is the reason that it is so much harder to improve the software testing, compared with the other stages of the software development process?

As it was mentioned at the beginning of the thesis, the testing job was not a likeable job, and the software tester's attitude towards the testing needed a long time to be considered as an accepted task, which impeded the testing technology development. Beizer's mental model (1990) gives very detailed description on how software tester's mental awareness changed and how it affected the software testing process improvement. Except this reason, the other reason is that the rapidly changed software technology development made it more difficult for the professionals to analyze and choose the right testing process. The related factors for selecting the testing process includes, setting up the test environment, choose the test tools, evaluate the test process methodology, etc. The quality of the tester's performance is recognized as the third reason to influence the outcome of the testing. Because wrongly executed test operations might straightly cause the test cases to fail. Even though the main reasons were found, there is no single silver bullet (Beizer, 1990), which could analyze and correct all of the different software testing processes. However, it is possible to find a useful and proper software testing model, which can be valuable to guide the testing work in practice. The realities prove that a proper software testing model can significantly help companies to improve the software testing performance from both the technical and the managerial side (Ericson et al, 1997).

This thesis focuses on the Test Improvement Model, which was selected from a group of independent models by exploring and comparing them. The research questions are proposed based on the literature review and are investigated and answered through the case study.

1.2 Software Testing Concept

“Software testing is the process of executing a software system to determine whether it matches its specification and executes in its intended environment.” (Whittaker, 2000, 77)

Software testing has the different taxonomy according to the different rules. It is apparent that there is no absolutely correct way to divide the software testing methods in a very common manner. Usually, the normal classification is based on the matter of software development life cycle (Whittaker, 2000), which uses the V model (Marick, 1999) to divide the software testing process into unit testing, integration testing, system testing and acceptance testing. This classification is mostly used in the software development companies, where the software is produced from the scratch to the final delivered product. However, if considering the software development process in more detail, the software testing could be executed in all of the different release phases. For example, in the software design release phase, the software functionality-based unit testing should be done. In the software release phase, the software acceptance testing could be carried out. In the telecommunication field, besides the software development phase related testing, the software testing usually means more, which includes the GSM protocol software testing, WCDMA signaling testing, SIM testing, system testing, et cetera.

In fact, no matter what kind of classification it is, the goal of software testing is always the same, which is to discover the symptoms caused by software bugs in the product, figure out the proper methods to prevent the bugs, and eventually improve the software quality, product stability and usability (Beizer, 1990).

After the objective of software testing is clearly claimed, the most concerned question the professionals care is that how to ensure the objective to be met in the real software

testing work. Normally, engineers like to put their attentions on the software testing technique from two aspects (Beizer, 1990). One aspect is the software testing technology aspect, in which the testing technique is mainly correlated with the testware, including test cases design and test tool adoption. In more details, it determines if the test cases are designed by tester or automatically generated from some software; it determines also if the test cases should be executed by the tester or by the computer automatically. The other aspect is mostly focused on how to improve the testing work efficiency from the managerial perspective. In other words, how to build the software testing team, plan the software testing project, arrange and choose the testware, and how to evaluate the test cases and review the test result, et cetera.

In order to build and improve the software testing work process, professionals have put more focuses on how to choose the more specific and even tailored testing model in the company. The model is usually summarized by the professionals from the real work experience, which could directly reflect on the improvement of the testing work efficiency and the product quality, also helpful to control and even deduct the cost. Therefore, the software testing process needs the model. Since the software testing process plays as a fairly important role in software development process from the both perspectives of technology and management, the practical benefits gained from the model-oriented testing work process are also from both of the perspectives. The benefits are, 1) the model is elicited from real work. So, based on the model, the professionals know more on how to do the job and what should be done. 2) The usual working process is standardized in a model, which simplifies the working process in the details. 3) The model could be applied to the similar projects as the reference, which could help company to better and quicker meet the product requirements. The similar projects can borrow the right experiences from the model. 4) Model is handy to operate. From the technical perspective, an improved qualified software testing model improves the accuracy of the software testing work, decreases the problem occurrence and guarantees the software quality. Meanwhile, the economical software testing process straightly dedicates to save the costs for the whole software development function. Furthermore, from the management point of view, efficient and economical software testing process

can help controlling the testing working schedule and budget, as well as ultimately improve the company operational performance in its totality.

On another side, software testing is not an exact science. It is both an art and a science (Murugesan, 1994). In the past decades, software testing has been researched and developed. However, professionals have realized gradually that test and evaluation methods, techniques, and tools in themselves do not guarantee effective testing and do not ensure high quality of software. It has to do with the professionals' recognition to the software testing itself. The key is to improve the attitude of the software professionals towards testing and to broaden the objectives of testing (Murugesan, 1994). The software test improvements could not be fully done just by giving engineers the necessary technology training and improving the software testing technologies. But, the suitable selection of testing model could inevitably give professionals more sense of how they could improve and what extra should be given more attention in order to improve software testing and software quality.

Therefore, it is obviously necessary that the suitable software testing model should be found to assist the software testing work in practice.

1.3 Software Testing-Related Product Quality

Software quality is defined by the International Standard Organization in 1986 as the "totality of features and characteristics of a software product that bear on its ability to satisfy stated or implied needs". In literature, there are two kinds of common understanding (McConnell, 1996) for professionals to be aware of the software quality. One understanding is the software quality includes all of the possible characteristics that you might consider when you think of a high-quality software product, like usability, efficiency, robustness, maintainability, portability, reliability and so on, which could be measured by the customer feedback or product feedback. The other one understanding of software quality is the low-defect rate of the software products, which is generally

measured by product failures statistics and user feedbacks. Apparently, the latter understanding could be measured in the software testing result.

Therefore, software testing has such tight relationship with software quality. Meanwhile, in the software company, according to the rapidly changing competition, the business strategy has changed from the simply shortening the software development life cycle (e.g. by using the XP programming method) to shortening the software development life cycle but improving the product quality, which is ensured by improving the testing process (e.g. by using the XP programming method and the testing improvement model). The quality is required to be emphasized more and more due to the increased concerns from the customer requirements and market competition. Only the improved software testing process could significantly improve the product quality in a visible sense, which means fewer faults, fewer bugs, more usability and compatibility.

However, in the real software development process, the different stakeholders have the different understandings and measurements to the software quality. So, the quality usually means differently for the different professionals. For instance, from the perspective of software developer, good software quality means the easy maintenance, good reusability and satisfied testability and so on. From the management perspective, software quality prefers to more quality awareness on how to decrease the resource deployment in software development (Chrishnan, 1993), such as cost, and how to increase the software product values and how to optimize the production process, including the software testing process, of course. From the end user’s perspective, high quality software has the easy and friendly user interface (UI), satisfied functionality module, ability to do all they want.

So, how to guarantee the software quality would be the most interesting question when professionals talk about the software product. In this thesis, even though there is no intention to study the software quality issue in very detailed level, it still emphasizes that the perfect quality could not be achieved only depending on the good software

testing, but also depending on many other issues. However, without good and efficient software testing, the product quality definitely could not be enhanced. That is why software testing and quality have the causal relationship in this sense. The software testing ensures the product quality; improved software testing could better ensure the product quality in more positive sides.

The quality improvement topic is the research theme of 90's and there are emerging perspectives on what constitutes high quality of the software. Software quality is more than an attribute that is normally attempted to build into software products (Murugesan, 1994). Strong quality focus is emerging in all the phases of software development and evolution, with increasing emphasis on product quality, process maturity and continual improvement, and this trend is extended to software testing since it is a vital element in software quality assurance (Murugesan, 1994).

1.4 Research Motivation

Software testing is part of software development process. Software testing needs model, so does the software development. When professionals began to research on the software development model, they have been able to accept the fact that the development process spends lot of budget. Therefore, the software developers choose to use the existing software development model when they develop the software, since the software development model can efficiently increase the working efficiency, lower the costs and ensure the optimized production process at the same time. Currently, the most popular software development model is the software Capability Maturity Model (CMM). The CMM was published by Software Engineering Institute in 1991, based on several years of software developing process experience, to measure the satisfaction of software production in organization (Daich, 1996). The model describes the software engineering and management practices that characterize organizations as they mature their processes for developing and maintaining software. The CMM consists of sets of recommended practices in a number of key process areas that have been shown to

enhance software process capability such as Requirements Management, Software Project Planning, Software Quality Assurance, Organization Process Definition, Training Programs and Integrated Software Management (Staab, 2002). Nowadays, a lot of companies choose to use this model assisting their software production and get the satisfied achievements.

Same principle to the software testing, the software testing also has a crucial position and it costs a lot of the budgets. According to John Viega et al. (2000) statistic evidence, during the software developing process, if a bug is caught in the requirements analysis phase, it costs about $139 to fix it. By the time when coding begins, the cost could rise to nearly $1,000 per bug. If the bug is not caught until after the project is completed, the costs rise significantly. Because the software testing is costly, and it is difficult to get improved, but so important to affect the software quality, the professionals naturally have to constantly make research on it for the purpose to find out the more powerful way to optimize the software testing process. The model-based testing process could efficiently offers professionals the points and directions that what and how the professionals should consider in such severe pressure. Therefore, the software testing model with the thorough consideration to the cost effectiveness should be designed and researched.

The SEI claimed, because software testing is a crucial working component, it needs to be allocated in the CMM as in detail as in the real work concerning of the cost-effective and quality-assurance factors. Software testing could make the visibility for software development process by giving an evaluated result for the process based on the testing output. Meanwhile, it characterizes the each process by executing the different objects-based test rounds to divide the baselines, and affect each work process as well. The satisfied testing result could guarantee the software quality and push the software developing process going on. The unsatisfied testing result will hinder the normal software development process and pull the software developing process backwards to find the defects and as a result to affect the project deadline. Therefore, the final aim of software testing process is to control the whole product quality and increase the

software development maturity level. Increasing the software testing maturity level could increase the software process maturity level, thereby to increase the organization overall software engineering process maturity level. So, it is necessary to study software testing process on the ground of the CMM model. However, the truth is the CMM model has been developed for the purpose to control the software development process and get the better performance. It focuses on the whole process, does not give very much detailed information on the testing process individually. It is obviously uncomforted with what researchers originally thought. In order to engage more software testing issues into the CMM, several software testing models right patterned after the CMM. The popular ones are Gelperin's Testability Maturity Model (TMM) (Gelperin, 1996), Ericson's Test Improvement Model (Ericson et al, 1997), Koomen and Pol's Test Process Improvement model (TPI), et cetera. All of those models have the same target, which is to use the less cost and time do better testing process, even though the different model has the different strengths, weaknesses and metrics. As the professionals might already notice, the TMM model is a very spectacular model. It is a complicated model, covering the measurements from the five different levels in the testing work process from the company perspective. However, implementing this model in the daily work for company is rather challenging since it evaluates plenty of the detailed and small aspects. Moreover, professionals find that this TMM is not very practical for middle size and start up company due to the limited human resource and financial resource. From the experienced professionals, the TMM is more like an assessment model rather than an improvement action model (Jacobs et al, 2000). In order to give an improvement action model, Ericson et al (1997) designed the Test Improvement Model to reduce the gap between the state of art and the state of practice for testing in company, which is approved to be suitable for the normal size company without any worry about the company size limitation in Sweden.

Moreover, according to the case research (Ahonen et al, 2003), the TMM has got into some Finnish company. It assesses the software testing process performance well in the big companies and it also facilitates some companies to improve the software testing performance in reality. But, the needed financial support and human resource support are very significant. The Test Improvement Model has not been spread so widely in

Finland so far, even though it has got very good achievements in Sweden and other countries (Ericson et al, 1997). This motivation pushed this study going on. The study mainly focuses on exploring the practicability of the Test Improvement Model in case company, making the possible adjustments based on the literature review and case study.

1.5 Research Objective

Based on the background analysis, the higher quality the software requires, the more attention at every stage of software development process should be given. Since there are several activities straightly affecting the product quality assurance, it is worthy of research those activities in the sense of how to improve them to the better way. Testing is certainly one of those activities (Eickelman and Richardson, 1996). In order to get the better software testing, the testing working process has to be investigated thoroughly, and the improvements have to be defined as practical as possible.

In the software testing process, the most difficult handle issues that affect the testing performance are software testware and test techniques, even though the whole software testing process is also very hard to be modeled and analyzed (Ahonen et al, 2004). The fact shows that without the modeled software testing process, the testing work is impossible to be done well. But, as long as the software testing model is designed, the testing function runs normally, the future testing improvement process could be measured and designed on the basis of the pre-defined software testing model. As a matter of fact, the actual software testing work is tough to do due to the insufficient software testing technology capability and the financial resource capability in company. Furthermore, there are a few detailed factors have to be considered carefully. They are: organization support and arrangement to the testing work, testing project team's planning and tracking, testware usage in the project, test cases and the test result review. All of these factors altogether compel professionals to choose and design the proper software testing process model.

This thesis will not research the every factor that might affect the company software testing performance. The focus would be on how to do the more efficient and optimized software testing process based on the existing software testing process. In order to achieve this research purpose, it is necessary to review the basic knowledge of software testing and software quality, software testing models, and the current software testing work situation. Finally, based on the analysis to the software testing work and the testing models, the research questions can be proposed and the case study would be designed and carried out in the research.

1.6 Structure of the Thesis

Structure of the thesis is important. The thesis is easy to read and understand with the systematical structure. This thesis tries to give readers an easily-navigate view.

The chapter 1 presents the background of the study, the motivation, the objective of the study and the structure of the study.

The chapter 2 proposes the research questions and problems on the ground of the awareness of the critical reasons, which hinder the software testing process improved.

The chapter 3 reviews the literatures concerning of the research topic: software testing, testing models and the improvement models. Through the literature review, the addressed models are analyzed and compared each other.

The chapter 4 further broadens the Test Improvement Model knowledge. All the related contents of the Test Improvement Model are explored, including the Test Improvement Model levels, key areas and the activities. The purposely-expanded knowledge concerned of the Test Improvement Model key areas were commented according to

some recently research and discussions. The possible adjustments and improvements of the Test Improvement Model are proposed here for the later case study investigation.

The chapter 5 describes the case study applied methodology in the empirical part of this study. The different research methods are compared here, the reason why the case study is chosen as the research methodology is explained here as well. The detailed data analysis techniques are given.

The chapter 6 analyzes the case study data. The main research question and the sub-questions are answered in this chapter. The data case analysis shows the positive approval to the chapter 5 about the adjustments to the Test Improvement Model.

The chapter 7 makes the conclusion and states the study contribution. Finally, the chapter 7 gives the further research directions.

2. PROBLEM AND QUESTION

According to Järvinen (1999), value of the research usually reflects on the research topic, research questions and the problems. The research questions define the purpose of the research by clearly identifying the relationship(s) the researcher intends to investigate. Only the research according to some proposed potential problems or questions would be more practical to help developing the existing academic knowledge or industry knowledge. In this chapter, the research problems and questions will be proposed based on the research interests. It is composed of the exploring to the current software testing actuality investigation, and the common understanding to the software quality issue, especially how the software quality is integrated into the software testing. Finally, the research problems and questions are proposed.

2.1 The Main Reasons for Company to Have the Software Testing Model

"In my experience, software testing model is needed at the real work. At the beginning, we do not know how to do the testing, what should be done. But, under the help of the testing model, the target can be made quickly, so that the work has focus. This is the most important point for company work. We need focus and guide."(Software Testing Line Manager, Salo, 2004)

Software testing consumes at least half of the labor cost expended to produce a workable program. It takes a lot of time and labor resources because there is no any easy-defined way to do this job in a definite-right manner. There are a few testing models and frameworks designed previously to guide software testing process done in a relatively unified and understandable manner, with uses the less expense but aims for better software quality. The company needs to implement the software testing model. The main reasons are: 1) In reality, very few software engineers like testing and test design, especially if test design and test execution take longer time than even program design and coding in the software development process. This attitude is problematic, but understandable (Beizer, 1990, 22). The root cause of this kind of emotion is because the company has no reliable software testing methodology to standardize and help the testing work. As a result, the working process is lack of reliable base; the professionals have to spend a lot of time thinking about what they could do and how they should do. In many cases, no suitable software testing model and no enough project documentation work are the most important reasons to baffle the software testing process improvements. Without proper software testing model in software testing team, the software testing process is hard to be analyzed and improved (Ahonen et al 2004). Therefore, the working process is messy, and it is difficult for company as well to assess the software testing process performance. Consequently, it is impossible for company to get the improvement plans. 2) Without the sound knowledge sharing mechanism, and without the enough documentation work, even though the company will organize the necessary training for the employees to develop themselves, the knowledge and experience are still hard to get sharing within the teams, which will directly decrease the possibility to improve the software testing process. 3) Moreover, software testing job is difficult to design due to the two major challenges out of the testing job itself. One challenge is that when the programmers code the software, they have to pre-consider

how to improve the software testability for the testing phase job. But obviously, this is very hard to be considered in advance. As a result, the short of the testability in software increases the difficulty for the testers to test the software. The other challenge is that the tester loses the motivation to test the software. The tester is usually required to find the bugs as many as possible during the testing job. But, what about if the given software snippet has no bug? From software developer side, it is a perfect job. But, from tester perspective, the job has no fun because the tester could not make any achievements by finding lot of bugs. In this case, from the psychological perspective, people do not like this kind of feeling since it spends much time but probably with little or none rewards (Beizer, 1990, 22). It shows again that the proper software testing improvement model should be built and followed in company to review the tester's working performance constantly in order to give them the sufficient motivation to continue the job going on. 4) Having the proper software testing improvement model, the team leader and company manager could always be aware of what is going on in company, how is the current situation and what could be done better, and so on.

Based on the above reasons, the company needs the software testing model. The thesis investigates on what kind of software testing model could be selected to guide the company work, and meanwhile, more importantly, which model can enhance the software testing performance in the most practical way.

2.2 Research Problems and Questions

As an element of software development process, software testing has the objective to allocate the critical defect location and remove it in software development process (Hedger, 2000). However, the current software development process models provide inadequate support for the testing process. Even in the most well known software development CMM, there is short of detailed software testing process management and evaluation criterion (Burnstein et al, 1996). Furthermore, the CMM involves the software testing as the software development process, but it has no specific software testing contexts for test managers and testers. Besides there is no focus on quality

testing as a process improvement neither (Burnstein et al, 1996). However, the reality is that when company is developing software, the professionals usually allocate their software development process according to the CMM. So, the software testing process is unable to be assessed and improved in by using the CMM. As a result, the software testing process has to be separated out of the normal software development process as a sole process to be planned, done and assessed. If continuing using the CMM, without any other assisted testing process model, the worst situation is the software testing process is probably done very roughly or even none.

Therefore, the most critical problem for the company is that they might have the nearly perfect software development model, but they have no good software testing work model to improve the testing function. From this perspective, the software testing process model has to be selected to improve the software development efficiency. This thesis intends to explore the software testing model, ranging from the classic Beizer's testing mental model (Beizer, 1990), Gelperin and Hetzel's testing workflow model (Gelperin et al, 1988), the best-known TMM (Burnstein et al, 1996) and the Test Improvement Model (Ericson et al, 1997, 229-246). However, after the exploring, the thesis plans only concentrate on the Test Improvement Model. The research plans to get the data towards how to do the feasible testing process improvement in Finnish company, how to measure the performance and how they could avoid the commonly happened problems in the real testing work. Thus, the main research question in this thesis would be: How to evaluate the Test Improvement Model performance in Finnish company?

Furthermore, the study intends to get some suggestions from the case study on:

1. How to build up an efficient testing group based on the working project?
2. How to do the testing related documentation work during the project and what should be documented in the testing phase?

3. What is the normal company rule to avoid the usual failures or problems emerging in project, if they exist?

However, there is no panacea for all the software testing process and the problems. Even the most perfect CMM or TMM have also some limitations that people have to pay attention on it, otherwise, it will not help organization to reach the aim, instead of taking disaster to the organization. Therefore, the study plans to answer the research questions from the interviewed company's perspective. The objective is to give a relatively practical experience to the practitioners:

1. How does company use the Test Improvement Model based on their current state of testing?
2. Is there any restriction the companies have to notice when they use the Test Improvement Model?

The study will go following the orders: doing literature review in the field of software testing models and their usage situation; investigating the current Test Improvement Model based on the literature review, studying and choosing the proper research methodology; making the case study, collecting data by interviewing the researchers and practitioners and finally analyzing the collected data.

3. LITERATURE REVIEW

In this chapter, the necessary information on the research subject and research questions is explored by taking literature review. The literature review process intends to achieve the following aims:

1. Learn about the software testing development situation; learn about the software testing improvement in the research domain.

2. Identify the possible gaps between the common testing knowledge and the testing improvement model, which is the research subject.
3. Understand the relationships among the several different software testing models, identify the suitable software testing model for the study use.
4. Develop a theoretical ground for the empirical study.

3.1 Foreword

Software testing process is a crucial process to improve the software product quality since it aims to find and correct the defects in the software development process. However, the principle of software testing is not to only identify the defects inserted in the earlier software developing phases, but to demonstrate, validate, and certify the absence of the defects (Hines, 2001).

The advancement of the software testing technology has shown that the professionals' recognition and understanding to the software testing area is deeper than ever. Currently, the professionals pay much more attention on software testing compared with before. However, testing state of practice is not as good as it ought to be due to the lack of the effective, standard and comprehensive software testing methods. As a result, more and more testing models appear all the time (Burnstein et al, 1996). All of the testing models try to give a suitable executive guide for the professionals when they consider the software testing. Generally speaking, in the case that the existing model could not resolve the emerging problems appeared during the testing process, the new model will come out right after the existing model or even right pattern after the existing model in order to give the more answers to the uncovered problems. Accordingly, there are a few software testing models designed for the purpose of guiding the software testing work and resolving the problems with the different focuses. These software testing models concentrate on the slightly different area of the software testing discipline, but they do not conflicts each other because their aim is the same, which is to improve the software testing process. Therefore, the various software testing

models are uniformed together in the sense of describing the whole testing domain realities, but with the different emphasis in the each model. For example, researcher might like to put the focus on the theory side of the software testing to research how the software testing technology developed in the history and what would be the possible developing direction in the future. In this case, the software testing historical model is useful for this study purpose. Whereas, if the company would like to use some software testing model to assess the current software testing process performance, they might find that the Test Improvement Model is more efficient for this aim.

So, there is no need to be confused when choosing the software testing model. And it is also not wise to say in general that what model is good and what model is bad because the different model has the different function to the different purpose in the different case. In this research, according to the research objective, only the Test Improvement Model is selected from the various models, as the investigated target.

In the following sections, some software testing models are explored from the general manner, and then the research focus would be moved to the selected model that suits the research context in order to expatiate the points and findings.

3.2 Some Classical Software Testing Models

There are a few software testing models that describe the software testing history, testing process and the most recent testing practice. In this chapter, the software testing models are selected, ranging from the software tester mental process development model to the most recent software testing process practice model. The idea of choosing these models is to see what is the professionals' mental change process to software testing work and what are the testing development achievements from the longitudinal perspective of time. The main purpose of exploring and reviewing these models is to provide the background information of software testing and elicit the useful information for the research.

3.2.1 Beizer's Testing Mental Model

Software testing walked a long path and spent lot of time to get development before the professionals could use the detailed software testing methodology and the process model to instruct their testing work. In the history record, before the professionals had the right attitude to the software testing, this industry discipline really suffered a lot. This is the initial motivation why the literature review starts from the Beizer's testing mental model.

Beizer's mental model describes how did the tester's mental attitude change to software testing. According to Beizer's vision, tester's attitude has very important influence on deciding if the testing job could be done perfectly or badly. Even though software testing has the techniques, methodologies, tools and standards, they can only aid in software testing execution process (Murugesan, 1994). The future technology development and improvement are mostly depended on the tester's working attitude. If testers intend to keep improving the software testing performance, the feasible way is to sustained learn about the software testing needs and summarize the performance from the testing experience with the positive attitude. Nowadays, the testers are educated to have the right attitude to this testing work, but, earlier before, the attitude was not right. Obviously, this attitude changing is not an over-night job.

"Knowing is not enough; we must apply.

Willing is not enough; we must do" *– Goethe, German Philosopher*

Goethe's words prove the application soul of software testing to the real work. Beizer's testing mental model (1990) incisively divided the tester's mental phases as illustration in the following table.

Phase	Symptom
Phase 0	Testing = Debugging
Phase 1	Testing = Showing how software works
Phase 2	Testing = Showing how software does not work
Phase 3	Testing = Reducing the perceived risk of not working to an acceptable value
Phase 4	Testing = Mental discipline that results in low-risk software without much testing effort

TABLE 3-1. Tester's Mental Phases (Beizer, 1990)

This table shows the different tester mental maturity levels on software testing. The phase 0 is the beginning level, existed until the early 1970s, when testing emerged as a discipline (Beizer, 1990). At that phase, there was no real effective testing and no quality assurance either. Tester could not get the enough education and training due to the insufficient development of software testing discipline. Tester was told about just do the very little debugging work to try to find errors in the designated programming code lines in a predefined context. Almost all of the tests were done manually. In the predefined context, if the output of the software is the same with what is supposed to be by the desired input, then, the test is over. And the tested scenario is very limited, designed by the programmer. In this phase, the testing was rather simple, easy and rough. The tester did not spend very much time and think about what they can do and what they can prove. They just finished the pre-designed job. They had no any motive to improve the software testing job. However, it is an understandable lower phase accepted by professionals because it was appropriate to the macro-environment characterized by expensive and scarce computing resources and knowledge, low-cost software (compared with the hardware), lone programmer, small project and throwaway software at that time (Beizer, 1990).

The phase 1 dominated the leading edge of software testing until the late 1970s when its fallacy was discovered (Beizer, 1990). In this phase, professionals noticed the difference between debugging and testing. They began to use the test cases to approve the software performance. However, this kind of testing manner had some problems since one passed test cases could only show the software works in this certain case, instead of proving that the software would work well in any other cases. So, in order to prove that the software works in any cases, the testers need to spend a lot of time to demonstrate it in any conceivable cases. Testers at this phase began to have some idea in mind that what they could test and prove, but the testing costs a lot naturally. This is obviously not realistic.

Not very long time after the phase 1, the testers' knowledge got improved and the software testing technology got improved. The testers began to realize that it is impossible to test every situation that the software works. It is more practical if they could find out how the software does not work, rather than under what condition, it works. The phase 2 was emerging. It is more advanced than the phase 1, but still had limits. If the testing reveals a bug, the programmer corrects it; the test designer designs and executes another test case intended to demonstrate another bug, which leads to a never-ending sequence of the testing (Beizer, 1990), which is still not ideal. In this phase, the testers' work performance was evaluated as the amount of bugs they found.

The phase 3 has a testing perception change. In this phase, the objective of the tester's work was not only to find bugs, but also to decrease the risk by finding bugs. So, it does not matter how many bugs the software has or has no, as long as the software satisfies the software requirements. The testing process has the dramatically improvement in this phase 3 because the testing work has actually the ending point. When the professionals got the enough confidence that the software satisfied the requirements, it can be shipped out.

The phase 4 is a state of mind phase. The professionals' knowledge of what kind of testing can and can not be done, combined with the knowledge of what makes software testable, results in software that does not need much testing to achieve the lower-phase goals (Beizer, 1990). So, testing has not been a "meet-attack" task, it has been a soul in the professionals' mind towards how to increase the testability and decrease the risk for testers. This is the most mature testing mental phase, implemented with the help of the effective testing process improvement models. What this study researches is at this phase. The current testing technology still stands in this phase.

From the Beizer's (1990) tester mental change model, the software tester's mental changes pushed the development of the software testing process forwarding. But, the testers' self-knowledge and education improvement is not ignored at the same time, the enough education and training is the premise. After this model emerged, the software programmers do not work only for programming; they have to consider how to make the programming code more testable for testers. As the same principle, the testers could not only work for their finding-bug purpose, they have to consider that how to test could decrease the potential risks. This change makes the testing process more rationale and more efficient. From the macroscopic situation, the testers' mental change helped very much for software testing technology development. But, certainly, in order to let the change happen, the microcosmic support from every software development company is necessary. In this sense, it could say that the importance of the organization has been officially introduced into the software testing process since the top management attitude also impacts the detailed software testing performance improvement.

3.2.2 Gelperin and Hetzel's Model

If Beizer's tester mental model initialed the professionals to seriously think about the importance of software testing attitude, which is the "soft" side of software testing. Then, the Gelperin and Hetzel's model (1988) gave the "hard" side of what the software testing should be, which is the software testing process workflow guide. This workflow

chart concerns what actually the software testing process is, what software testers could do and how they should do. This model was first designed in 1988, after the Beizer's tester mental model. Gelperin and Hetzel combined the surveyed industry work performance and concluded this basic testing process data flow chart (FIGURE 3-1).

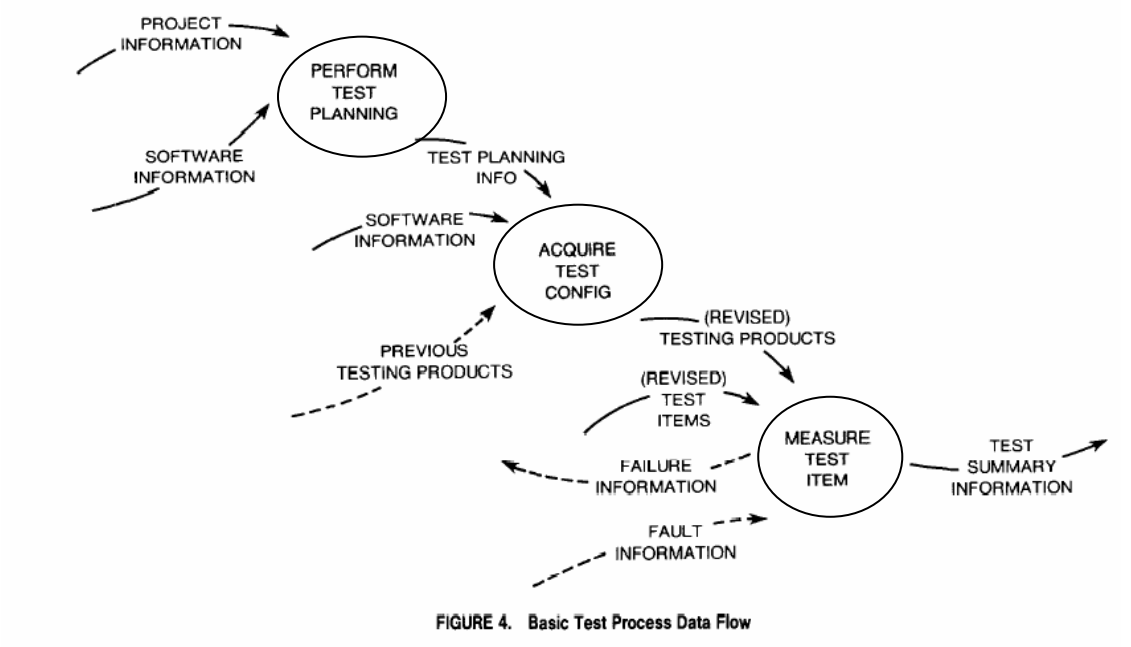


FIGURE 3-1. Basic Test Process Data Flow (1988)

In this model, the basic testing process workflow is described. Testing process is thus divided into three phases – planning, acquisition and measurement. During the planning phase, information about the tested subject and the on-going project are used to develop test objectives and overall testing approach (Gelperin and Hetzel, 1988). The output in this working phase is the test plan and the related documentations, e.g. the aimed testing features and so on.

The second phase is acquisition phase. In this phase, based on the software information, e.g. the software specification and requirements, the test cases are designed. The previous testing documentation could give the instruction on how to set the test configuration. The outputs at this phase are test cases and the test report.

In the last measurement phase, the test cases are executed. The test execution report is compared with the software product requirements and specification. If there is any fault information, the test process should go backwards to the previous phase and iterate the same activity, as described in the second phase. The outputs at this phase are test report, summary reports and the related documentations.

This model gives professionals the relatively standard guideline on how the software testing should be planned and done at the early software development history. This model is famous and classic because it tells professionals what composes of software testing work and how they should be done. After this model, there are a few other models designed based on the idea from this model. Nowadays, the exactly same testing process is still deployed into the practice in some software companies.

However, this model has some limitations. It only puts focus on the objective side of the testing work, which is what jobs should be done in the process. It is very rough conclusion and grouping. For the detailed subjective side of the testing work, which is how these jobs should be done, this model does not cover.

3.2.3 The Software Testability Maturity Model (TMM)

After Gelperin and Hetzel (1988) described the software testing process workflow, the professionals began to put more attention on how to do detailed software testing job based on the workflow model. In the academic field, the researchers agree on that the workflow model is standardized. However, in the industry field, this workflow model is only the ‘read thread’ of the software testing work. It is not enough for the detailed actions. Based on this ‘red thread’, the more specific actions should be defined. The reality is that the different companies prefer to have their own testing workflow model. They usually make some certain tailoring changes to the existing models according to the real situation. No matter what researchers and practitioners do at their work, the

tendency is that after the testing process workflow was model, the next requisite model should be the measurement model to assess if the testing work is valid. Thus, how the software testing work should be measured became to the hot topic.

The Software Testability Maturity Model (TMM) is emerged to satisfy the measurement need. It was first proposed and developed by Burnstein et al from the Illinois Institute of Technology (Burnstein et al, 1996).

In the TMM, there are following statements (Burnstein et al, 1996):

- The TMM is composed of a set of maturity levels, which define a test maturity hierarchy level. Each level represents a stage in the evolution to a mature testing process. Movement to an upper level implies that lower level practices continue to be in place.
- The TMM has a set of maturity goals for each level, and the activities, tasks, and responsibilities needed to support them.
- An assessment model is included into the TMM, which consists of three components: a set of maturity goal-related questions designed to assess test process maturity, a training program designed to select and instruct the evaluation team that is to conduct the maturity assessment, and an assessment method that allow an organization to assess itself based on responses to the questionnaire and interview data.

Hines (Hines, 2001) illustrated the TMM levels in a more understandable way. In the figure (FIGURE 3-2), each level has the different goals. The level 1 is the initial level. In this level, software testing is a chaotic process (Staab, 2002). There is no or very little software testing in the work. This level, therefore, is the worst mature level in the TMM. The most possible reason for this situation is that there is no trained professional

testing staff and testing tools (Hines, 2001). The goal of this level is to show the software works.

The level 2 is the phase definition level. In this level, software testing is identified as a separate function from the programming debugging (Hines, 2001). The basic testing techniques and methods are in place and the goal of this level is to show that the software meets specification (Hines, 2001).

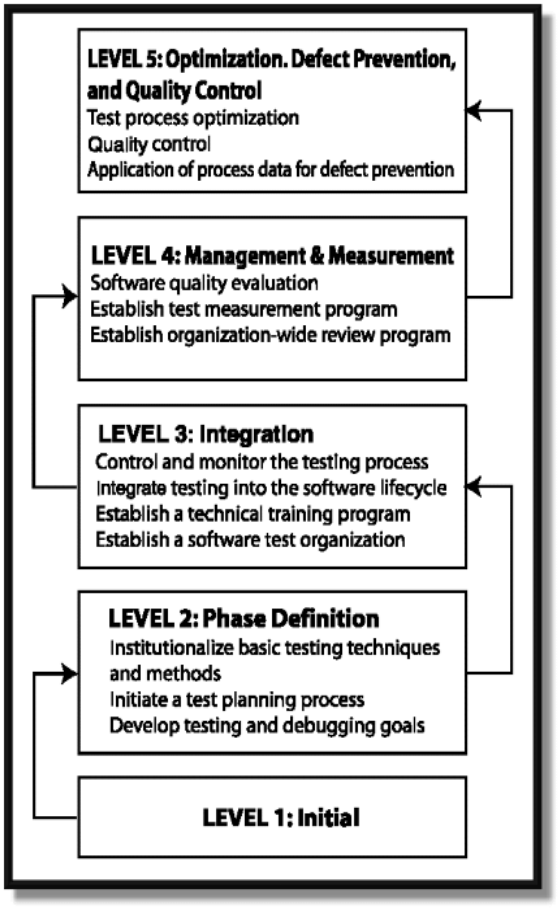


FIGURE 3-2. TMM Levels (Hines, 2001)

The level 3 is the integration level. In this level, the testing has been integrated into the whole software development life cycle. The formal testing function is built in this level. There are trained testers; the planned and controlled testing phases and the necessary

advanced testing tools. Compared with the level 2, this level is more advanced since the organization has implemented the organized testing work environment in it. The goal of this level is to realize testing as the professional activity (Staab, 2002).

The level 4 is the management and measurement level. In this level, the software testing work is measured and qualified. The testing work is evaluated and the software product quality issues are also in consideration. The developed products are now tested for quality attributes such as reliability, usability, and maintainability (Staab, 2002). The test documentation is managed in this level, the documentation, e.g. test cases, test reports and so on are recorded into database for the reuse purpose or regression testing later if it is needed.

The final level is the optimizing level. In this level, the testing is institutionalized within the organization. Testing process is well defined and managed, testing cost and effectiveness is monitored and the testing tools could be selected based on the established procedure.

Assessments of the testing process using the testing maturity model will [Staab, 2002]:

- Document the current level
- Highlight the variances between the imagined level and the actual level
- Provide a road map for making the necessary process improvement.

The TMM is widely accepted by the companies nowadays to measure the company testing work situation and produce the following-up improvement plans. However, it is not perfect. The reason is the TMM itself is only a measurement model to evaluate how mature the organization testing work is. It does not involve very much attention on how to improve the software testing process if the measurement result is that the testing work is not so satisfied. So, the company has to find other clue to design an improvement plan. From this perspective, this model lacks of the practicability. Besides, the TMM

seems including very broad knowledge and side in it, it is a bit too complicated. There is so called gap between the art and the practice.

On the other hand, considering the TMM capacity, this model is probably more suitable for the big company if they would like to measure the software testing function performance due to the model complexity. But, for the small and middle size company, it is not so ideal to use since the testing situation investigation and improvement design need certain time, small company could not afford this time resource. This model covers so much, which obviously need the necessary human resources and financial resources delivered to the project, which is also hard for small company to afford.

3.2.4 The Test Improvement Model

The professionals are appreciated of the excellent ideas from the TMM on how to measure the maturity of the software testing work. Meanwhile, they are still looking for the easier and more practical ones. The software Test Improvement Model is supposed by T. Ericson based on the CMM and the TMM model, which keeps the positive features in the TMM and at the same time, owes more understandable, feasible and effective advantages.

In order to give the practical guide and improvement suggestions to the software testing work in company, the Test Improvement Model was designed aiming to identify the current state of practice in key areas of testing, and after that giving an improvement plan. In another words, the Test Improvement Model can find out where the testing work is ´on the map´ and then, chart a way to ´the destination´ (Ericson et al, 1997). Measuring and synchronously improving are the basic objectives of the Test Improvement Model to decrease the gap between art and practice in software testing improvement.

Test Improvement Model includes four testing process maturity levels. They are: baseline level, cost-effectiveness level, risk-lowering level and optimizing level. Besides the main goal of each level, which has been indicated in the level name, the four levels have their subgoals, as described in the table (TABLE 3-2):

Level	Subgoals
Optimising	Managed, documented overall level of testing corresponding to the organisation's quality policy. Cultural change towards prevention and continuous improvement. Testing is fully integrated with the project life cycle. Decisions are based on facts.
Risk-lowering	Support from all levels of management. Detection of hazardous errors. Risk driven scheduling and resource deployment.
Cost-effectiveness	Minimal cost of rework and change handling. Fast and systematic testing. Tasks are performed in "the right way". Minimal redevelopment of existing artefacts or parts thereof.
Baseline	Documented standards, policies, procedures, methodologies and methods. A basic testing function.

TABLE 3-2. Test Improvement Model Maturity Level (Ericson et al, 1997)

In order to achieve the goals, each level employs some strategies, as described in the following table:

Overall goals	Strategies for achieving the overall goals of each level
Optimising	Knowledge and understanding through experimentation and modelling Continuous improvement Root cause analysis Balancing of testing against product needs and organisation's quality policy Co-operation with all parties of projects, in all phases of development
Risk-lowering	Early involvement Expense justification Analysis of product and process problems Measurement of product, process, and resources Risk analysis and risk management Communication with all parties of projects
Cost-effectiveness	Early detection Computer aid of testing tasks Training Re-use
Baselining	Documentation of standards, methods, and policies Analysis and classification of product problems

TABLE 3-3. Test Improvement Model Strategies (Ericson et al, 1997)

Similar to the TMM, the Test Improvement Model also has Key Area. Currently, the Test Improvement Model has five Key Areas. They are essentials aspects affecting the software testing work. These five KAs are organization KA, planning and tracking KA, Test case KA, testware KA and reviews KA.

3.3 Software Quality and the Improvement

Software testing is always related to the software quality. The quality is the objective, testing is the method. The right quality is achieved by satisfying all the project parties via testing. That is why in order to assure the software quality, it is necessary that as many as possible of the project parties are represented in every phase under the development, including testing (Ericson et al, 1997).

The importance of software quality is emphasized by a recent study (Ahonen and Junttila, 2003, RTI, 2002), which estimated that in the IT field, the costs caused by "inadequate infrastructure for software testing" in the USA alone could be as high as 60 billion US dollars a year. On the other hand, testing is one of the principal means for assuring sufficient quality of software.

Software quality could be controlled by implementing the well-structured software development process model and the testing process model, e.g. CMM and TMM, Test Improvement Model, etc. However, it could be also ensured by the quality assurance methods, e.g. Six Sigma, TQM and so on (RTI, 2002).

3.4 Summary

According to the literature review, arguments and the proposed research questions and problems, this study intends to put the emphasis on the Test Improvement Model. The

other important reason why the Test Improvement Model was chosen as the research topic in this thesis is because there is not so much Test Improvement Model practical information available in Finnish industry. So, this study also intends to provide a guide to Finnish industry that whether the Test Improvement Model is suitable for the Finnish industry or should the Test Improvement Model be modified when implementing it at work?

4. THE TEST IMPROVEMENT MODEL

The Test Improvement Model is used for identifying the strength and the weakness of the company testing work performance. It could identify and prioritize the improvement activities on the ground of the assessed result. By referring to the Test Improvement Model, the company management could increase the visibility into the shortcomings of the company testing work situation and as a result to organize the required 'fight-back' actions.

The Test Improvement Model is composed of two components: framework and assessment.

The framework acts as a testing process guideline for the company, composed of the 4 different levels. Based on the guideline, the company could set up the testing functions that include the different aspects. The company could allocate the work reality in the proper level in the Test Improvement Model and so that choose to do the most suitable effort suggested by the Test Improvement Model. The Test Improvement Model also could be used for the company to find the existing problems according to the advised sub-components in the framework. Different level that numbers from 1 to 4 focuses on the different perspectives of the company testing work areas. Totally, the whole framework covers the all necessarily considered perspectives for the company if the company wants to do the testing process improvement. Consequently, if the company's

testing function satisfies all of the four levels in its daily operation, then, it means the company has a matured Test Improvement Model as ready.

The assessment component consists of a set of checklist. By using the checklist, the current company working status could be determined. And after that, it recommends to make up an improvement plan based on the framework items to develop the testing performance.

4.1 Levels and Goals of the Test Improvement Model

Similar to the TMM, the Test Improvement Model includes four maturity levels. The four maturity levels are followed in the certain order they build upon one another: baseline level, cost-effectiveness level, risk-lowering level and optimizing level (Ericson et al, 1997). The different level represents the different testing performance status of the company. The upper level stands on the ground of the lower level, the lower level is still in its place. For example, if the company's testing work performance only stays at the baseline level, then, it has no consideration on the company testing work area on the cost-effectiveness level. On the contrary, if the company testing work performance stays at the optimizing level, which means it must have satisfied the baseline level testing function, and it also has the sufficient consideration on the cost effectiveness level and risk management issues at the real work.

Each level in the Test Improvement Model has the certain main goal, there are several sub-goals coming along with the different levels. It is reasonable that the higher level the company locates, the higher and possibly more goals it could realize. Therefore, the better testing work efficiency and performance the company gets. In order to achieve the different sub-goals, there should have the paired strategies to assist testing work advised by the Test Improvement Model.

As explained in the previous chapter, the table (TABLE 3-2) illustrates the four maturity levels of the Test Improvement Model and the corresponding sub-goals. It is easy to see from the table that the lowest level is the baseline level. In this level, the testing work baseline is established, from which the deviations can be made and evaluated. The sub-goal in the baseline level is to build up a testing function, i.e. a group of professionals who are dedicated to do testing have to be organized. And the testing work has the documentation standards. This is the basic starting level of software testing function for the company. According to the sub-goals of this level, the fundamental resources should be allocated, i.e. necessary human resources and necessary financial resources.

The second level is the cost-effectiveness level. Usually, it is reasonable for the company to focus on how to save the testing work cost after they have set up the testing group. In order to achieve cost effectiveness and control the testing work cost, the most important sub-goal is to make sure they do the right testing work in the right way, therefore, to save the unnecessary cost. This requires the early detection of software testing since many faults stem from early phases. The cost that is spent on the faults detection and correction grows dramatically quickly when the faults have been migrated and propagated to later phases (Gilb and Graham, 1993). In fact, the cost of fixing a defect is minimized if it is detected at the same phase where the defect is introduced. It is reasonable that a ‘robust scope for evaluation and test must encompass every project deliverable at each phase in the development life cycle’ (Bender, 1996).

When the project tries to save the cost, it has to consider the risk issues inside. So, when it is necessary, using a little bit more budget to decrease the risk is recommended. The third level is the risk-lowering level. It is accepted that when a project is on the process, if professionals put too much focus on the cost-saving aspect, the related incident risk is easy to be increased due to the too much financial resource control. For example, if the project decreases the amounts of the involved professionals in order to lower the budget, the potential risk is the remained professionals could not replace the positions that are originally filled by other professionals. Obviously, it is not a wise decision for the project work. It also happens very often that for the certain project, cutting down the

budget finally might cause the reduction of the necessary workload because project has no enough money to do the job. The result from the case is that the required work is missed. Therefore, the goal of this level is to do the best estimation, control and act to the potential risks in the Test Improvement Model. In this level, the support from the management level is indispensable.

The fourth level is the optimizing level. In this level, the main goal is to optimize the testing work under the condition of fulfilling all the other lower levels' goals. In this level, the software testing performance is more tightly related to the software product quality. The effective testing can give the product quality assurance, and its purpose is 'guarantee a minimum level of quality in the finished product (testing cannot demonstrate quality)' (Mosley, 1993), as Weinberg (1992) claims that quality is the subjective of testing. In this level, the more non-technology related issues are added into consideration. Ericson et al (1997) argues that since during the software development process, there are plenty of different parties involved into the work from the product development beginning to the end. Those different parties might have the different main criteria to the quality issue. Therefore, how to enhance the software quality by optimizing the software testing process may be more difficult. In this case, the communication among the different parties is rather important because the sufficient communication can effectively help the different parties get the "read thread" of how the testing should be done and what is the satisfied software quality. Furthermore, Ericson et al (1997) says, in each step of the software development communication process, e.g. in the product requirements meeting, and in the product release meeting, the understanding to the software quality might be slightly different because they have the different angle to analyze the problem. So, the communication has to be as smooth as possible, and as exact as possible to guarantee the good co-operation between the different parties. The person who takes the responsibility of transferring the communicated information should try to develop a common language and use the common language to spread the testing related information into the different persons as much as possible. By optimizing the testing process and effective communication sharing, the greater consistency between the different parties can be achieved.

From the analysis, it shows that four different levels of The Test Improvement Model do not conflict each other. So, they should be combined together in order to get the big view about what does this model give and how they can be ensured by using this model when implementing the Test Improvement Model into the real company work. The model gives company the basic idea on how to measure their software testing work performance on the ground of the current state. It is to find the right place “on the map”.

Each of these four levels has the certain concentration on the certain perspective, as the description before. To some extent, these four levels also represent the company testing improvement maturity (Staab, 2002). Therefore, it is natural that the conclusion can be made based on what level the company’s testing process locates. The higher level the company locates, the more matured the testing process is. However, only having this Test Improvement Model level is not enough for the company because the situation might happen that the professionals have no idea on how they should take the detailed action, even though they have already got the rough picture about from what perspectives they should consider the testing improvement work issue and what goals they should achieved ultimately. Therefore, Ericson et al (1997) suggests the strategies (TABLE 3-3) followed by the every level to guild people what they might do in order to achieve the goals for the different levels. These strategies are very practical guidelines for company.

4.2 Key Areas of the Test Improvement Model

Furthermore, Test Improvement Model has five Key Areas (Ericson et al, 1997). Each level is considered from the five key areas, which are the necessary aspects covered in the testing work process. The relationship between KAs and Test Improvement Model levels are illustrated in the following figure (FIGURE 4-1).

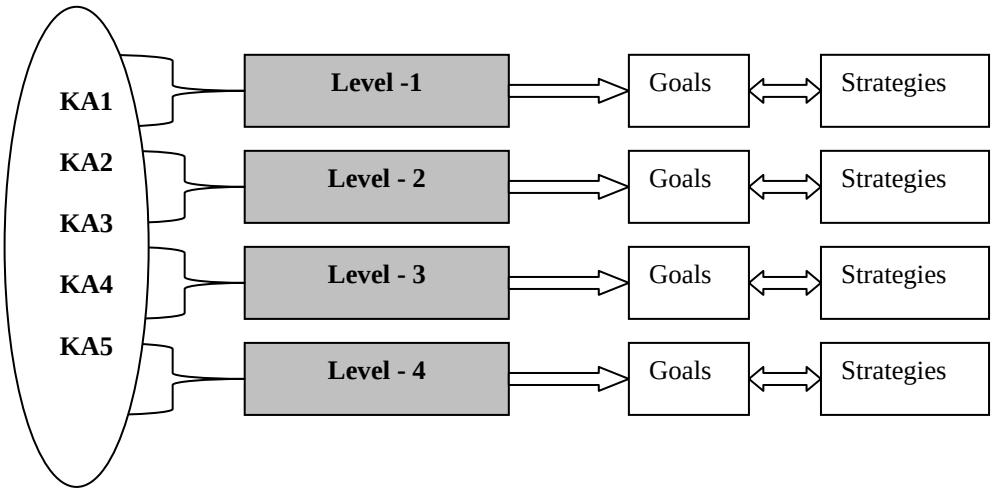


FIGURE 4-1. Test Improvement Model Structures

Respectively, these five KAs are the organization area, the planning and tracking area, the testware area, the test cases area and the reviews area. Each KA represents an important aspect of the company testing work. On each area, there are a few area activities distributing the testing work, but related to the four maturity levels. They are similar to the maturity level strategies, but they do not make any conflict with the strategies. The strategies are meant to fulfill the maturity level goals, whereas the activities are the detailed work actions from the different KA perspective.

4.2.1 Organization

The organization KA mainly concentrates on the overall organization support given to the testing work from the different levels. It also highlights the overall effectiveness of the communication sharing among the different levels. A good physical working environment can facilitate any job to be well done. So does the software testing job. For example, the necessary support from the organization is needed when the project is going on. The organizational support includes a dedicated project team without the communication barriers, the pre-agreed working language for the certain project, and the proper working tools and utensils (Jensen 1996) designated to the project. The psychological environment is also important (Ericson et al, 1997). The psychological

environment reflects on the group members should be able to work in the team environment. They should be open, active and be able to share their knowledge with the group members, etc. The satisfied communication sharing can give the better understanding of testing development, which can result in happier testing process. All in all, the good organization management is the premise to guarantee the success of the software testing process. The following table (TABLE 4-1) gives the more detailed information about the what The Test Improvement Model suggest the organization KA to do during the testing process.

	Organization
Optimizing	Testers are part of multidisciplinary teams Continuous organizational improvement Groups exist for process improvement
Risk-lowering	Test sensitive resource allocation and task planning is performed at all levels Testing function strives after support from all levels of management Tester involvement in all development phases, and development involvement in testing Personnel rotate between development and testing
Cost-effectiveness	Testing function is co-located Key people work full-time on one project and resources are not shared Independent testing function Team structure is based on resource stocktaking and project needs Responsibilities are clearly defined, documented and understood Test group has a dedicated project area, and proper tools and utensils Training to meet organizational needs
Baselining	Testing function is organized according to documented standard There exists a test leader and a core test team

TABLE 4-1. Activities of Organization KA (Ericson et al, 1997)

Except the description listed in the table, there are some more findings. Normally, organization designates the organizational model to the projects. According to Ahonen et al (2003), the impact of the organizational model to project working process is critical. To the same reason, the performance of the testing process work is also engaged into it. The organizational models are referred to the three models according to Ahonen et al (2003), which are respectively team model, interdepartmental model and resource pool model.

The team model is a relatively static model. The independently different project teams are built up separately and each of them works individually with its own project task. All the projects could go in parallel or not, depending on the situation (FIGURE 4-2). The software engineers are involved into the different project team, so as to the test engineer, led by the team leader. In this kind of organizational model, the enough resources should be accessible at any time for the purpose of not impacting any working performance on the possible other project teams, where they are using the same resource. The underlying problem for this kind of working model is that if Team B has several projects at the same time, but by happened, it is short of the project members. The Team B might want to borrow members from the Team A for the temporary help. In this situation, the Team A leader might: 1) Not so willing to give the best member out due to the protectionism of his own team working performance. 2) Have no enough human resource to offer the external help due to their projects working situation. To some extent, it is not good for the knowledge and experience sharing and learning in the whole organization. And another potential problem is that the working module is very rigid.

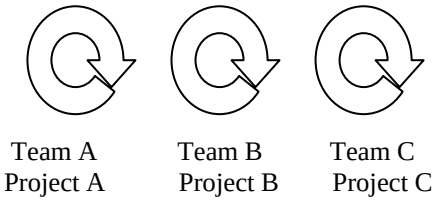


FIGURE 4-2. Team Model (Ahonen et al, 2003)

The interdepartmental model works better in the sense of knowledge sharing and learning. By implementing this model, the project is always divided into the several phases according to the project life cycle, for example, the normal project has project planning phase, project execution phase, project testing phase, etc. In each phases, there is a certain team taking the responsibility of this corresponding phase. As a result, how many phases the project could be separated, how many project teams there should be. All of those teams need to collaborate each other seamlessly until the whole project is

worked out (FIGURE 4-3). By implementing this organization structure, the clear responsibilities should be predetermined in order to make the several teams work clearly well. So, during the project, every team member has the specific task in the specific team located in the specific project phase. However, apparently, the potential problem for this working style is that if there is any person in any team who is not a team-worker, it is easy that the teamwork would be spoiled.

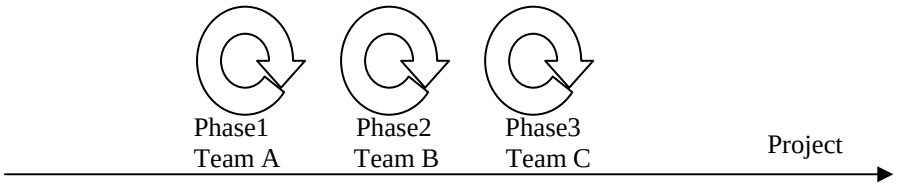


FIGURE 4-3. Interdepartmental Model (Ahonen et al, 2003)

The resource pool model works like a warehouse. All the employees, equipments, facilities with the accessories are collected into a ‘pool’. In case there is project upcoming, the team leader could choose whatever needed from the resource pool, including human resource, hardware resource and software resource, and then build up the temporary project team. After the project is rolling out, the team leader releases the team and returns all the resources back to the resource pool (FIGURE 4-4). It is a very flexible working style.

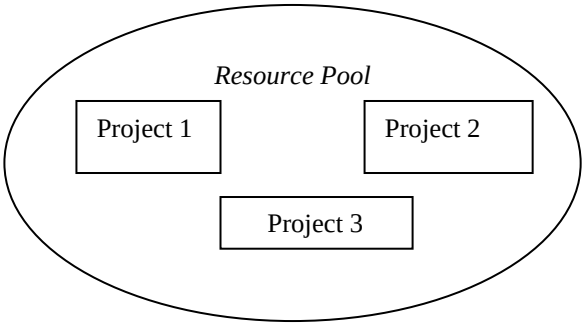


FIGURE 4-4. Resource Pool Model (Ahonen et al, 2003)

If comparing these three organization models, it is easy to analyze and get the conclusion that each model has its own unique advantages and also disadvantages. Based on the real company condition and project situation, the different models might be useful for the different cases. The resource pool model would be the most flexible working model for the start-up and small company since it needs the less resources compared with those two models. Whereas the team model might be only suitable for the big company because it needs the most resources. As a matter of fact, from the literature review and the personal working experience, most of the companies nowadays use the resource pool organization model. It is easy to manage, easy to control the cost and risk. However, to the testing work, which kind of model is better for the common situation and which model has the best help to the company testing improvement, this would be found out in case study later on.

In the organization key area, from the organizational perspective, there is another issue that must be taken into consideration. From the Test Improvement Model, it has the requirement that in order to improve the working efficiency and cost-effectiveness. The testing process should be automated as much as possible. In the reality, the more and more company is directing to change the current testing mode from manual testing into automation testing in order to catch up the fashion. However, the most likely happened embarrassing problem is that in some cases, the management does not make the enough evaluation and investigation before they begin to change the working style, which results that the automated tool and testing work process works far away from the expectation. Therefore, company should recognize that automating testing process is not an easy and over-night plan; it needs to be researched very comprehensively before implementing it, it also needs the well support from the management. It needs the sufficient time and resource to evaluate that if it is worthy of changing the current testing routine into automation testing routine, what kind of automation tools are useful in this case, etc. Changing working mode without carefully thinking, researching and consideration only brings the chaos. Therefore, before trying to automate the testing process, the enough research has to be addressed.

4.2.2 Planning and Tracking

Test planning and defect tracking is crucial to software testing. The good testing planning and tracking management could reduce the test cycle time by providing a structured testing approach; improve the testing efficiency by facilitating the communication, collaboration and information sharing among the team members; increase the visibility of the testing process by allowing team members to monitor and track the defect and gauge the application readiness.

In the planning and tracking KA, it is required that after the moment when the testing plan is made, the plan should be watched and tracked by the certain person from the beginning to the end of the testing process. This KA crosses over from the baseline level in the Test Improvement Model to the optimizing level since this area is a very important and basic area. The well-scripted testing plan could improve the testing work based on the research and the project experience. Guided by those activities, the results of the planning effort are preferably documented in a test plan, which is required to make the verification to other parties in a project and to inform the ones that are going to perform the actual testing. The plan should be documented according to a standard, e.g. the IEEE Standard for Software Test Documentation (IEEE, 1991). Moreover, at the cost-effectiveness level, it is strongly recommended that a trained tester should do the planning because the trained tester could ensure the planning is decently required. According to Kaner et al (Kaner et al., 1993) evolutionary planning facilitates:

- *Learning before thinking*—as testing proceeds, new knowledge of the product is gained. More knowledge facilitates better understanding, leading to better testing. The traditional approach requires a complete test plan before testing is initiated. The consequence of this approach is that most of the thinking is performed before most of the learning. The evolutionary approach allows planning and designing as the testers learn.
- *Consistency with requirements*—a large amount of the testing is based on the requirements. Normally, the large projects are perhaps spanning several years. The requirements are likely to change a number of times

very often. The evolutionary approach enables less reworking of test plans since the plans are written with respect to the factual requirements.

- *Best testing effort with respect to resources*—since the test plan is written on a 'need to' basis, no unnecessary planning is performed. That is, if management cuts back on testing resources, the testing performed up to that point is the best possible with respect to consumed resources.

For the more detailed activities description, the following table (TABLE 4-2) gives more items.

	Planning and tracking
Optimizing	Models are made of costs/resources, risks, etc. Planning and tracking activities are continuously improved based on analysis metrics and experiences Post-mortems are performed and results are duly distributed
Risk-lowering	Risk analysis is performed and influences planning and resource deployment Affected parties approve plans, including test objectives Fulfilment of overall test objectives is monitored Planning is performed by an experienced tester/planner Plans change with factual circumstances
Cost-effectiveness	Planning and tracking is computer aided Generic plans are used The choice of test methods and test levels is balanced against test objects and test objectives Progress is measured, and compared against plans Planning is performed by a trained tester or planner Planning is performed in an evolutionary manner Resource prediction is done according to documented policy
Baselining	Basic planning is performed Entry and exit criteria are defined for all testing levels Test results are documented, and duly processed and distributed Plans are developed following standards and deviations are documented and motivated Planning is performed according to documented policies Plans change with requirements Changes to the test plan are made according to a documented policy

TABLE 4-2. Activities of Planning and Tracking KA (Ericson et al, 1997)

In addition, the testing team must be properly structured, with defined roles and responsibilities that allow the testers to perform their functions with minimal overlap. The tasks have to be divided uncertainly regarding of which team member should perform which duties. All of these have to be done in the planning phase.

The more important part of testing planning work is to decide that what should be done in the test round, what should be documented and how to make the documentation work in the more understandable and standardized way. Over the several years, a lot of documentation techniques have been invented to support the control of testing plan. The documentation techniques are varied according to the different companies with the different names and different purposes. The IEEE developed a series of 829 Standards for software test plan in order to try to standardize all types of software testing. It considers from the different perspectives that how to make the test plan. The clauses are included in the test plan according to the IEEE829 standards are (IEEE 829):

- Test plan identifier
- Introduction
- Test items
- Features to be/not to be tested
- Approach
- Item pass/fail criteria
- Test deliverables
- Testing tasks/responsibilities
- Staff and training needs
- Schedule
- Risk and contingencies
- Approvals

As a result, the documentation policy and test subjects involved in test plan could be settled down by using the standard easily.

Test tracking is also crucial for the testing process since the software testing is a repetitive task. The test process usually includes several test rounds. The test tracking can give the possibility for checking the test logs in the previous test rounds. At the same time, the guaranteed test tracking method could provide the sound traceability to the testing work. The test engineers can trace forward/backward the original/desired requirements or starting/ending point of the test cases that where the testing result comes from/to, as you can see from the following image. For example, in the test case Z, if the state A is the starting state. Then, when the state A satisfies the requirement X, it can change to state C via transition 4, with the additional requirement YY. When the state C satisfies the requirement X, XX and YY, it C can continue changing to state B via transition 5, with the additional requirement Y. Finally, the state B can still change back the state A (starting state) via transition 1, with the requirement X. In this case, the good traceability enable the test engineers make the analysis when they meet the test cases fails about what is the required requirements, and if there is anything missing. When the test cases indict inconclusive, the test engineers can trace the state of the test cases and make sure what is the desired test result.

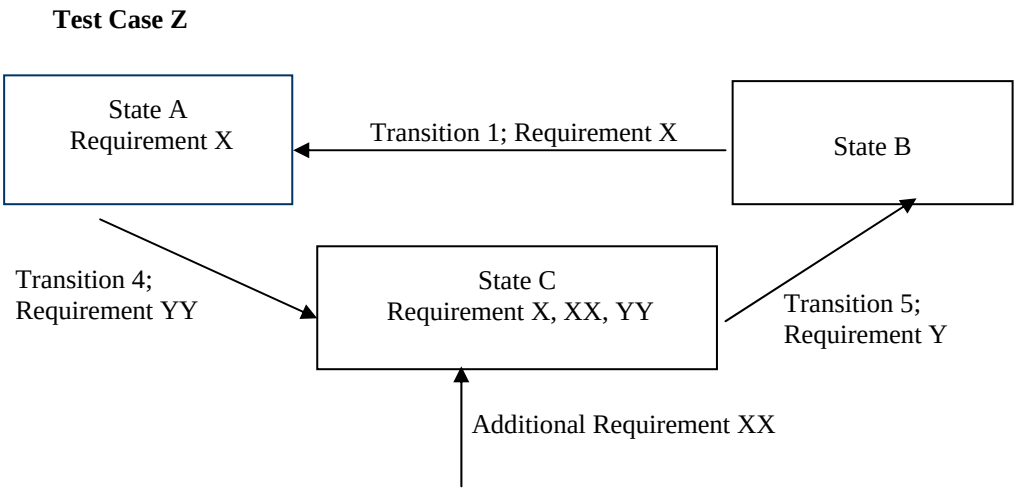


FIGURE 4-5. Traceability Example Model

Unstructured testing without any following-up standard process can only result in the creation of tests, designs, and plans that are not repeatable, and cannot be reused for the future iterations of the test. It is difficult to locate and track decentralized test documents. The so-called structured testing means that the whole test rounds are organized and executed as a predefined process with the contingence plans. If there are any changes happening every now and then, from the beginning to the end of the job, the contingence plan can help the emergency situation. Even though it might be true that in the reality, for many small companies, without predefined software development process could still work well, including the unstructured software testing process works well as well. But, it is still recommended that the structured working element process could definitely optimize the whole working process from the long-term run, especially for big company.

4.2.3 Test Case

Testing should be executed with the test cases help designed before the test initiation. The test case KA concerns the test cases design and the documentation. In this KA, the focus of the improvement is to do the better change management and test cases reuse. Change management includes the change of the specification, the product requirements, or the product updating that straightly leads to the change of the test cases. Test cases reuse means the same test cases can be reused into the different testing rounds or different products. If the company could control these two issues very well, the cost could be significantly saved and the risk could be efficiently decreased. Thereby, the optimizing of the testing improvement could be achieved. The more descriptions are in the following table (TABLE 4-3).

	Test cases
Optimizing	Test case design techniques are measured, reviewed and improved
Risk-lowering	Test cases are ranked and selected according to criticality Test cases are designed in response to risk areas
Cost-effectiveness	Test cases are designed using documented techniques Testability aspects influence requirements and design Test cases are recorded and re-used Test cases are organized with respect to testing levels, requirements, test object, objectives, etc.
Baselining	Test cases are revised when requirements change Test cases are based on requirements Test cases are designed and documented according to a documented policy, prior to execution Test cases are executed according to documented instructions

TABLE 4-3. Activities of Test Case KA (Ericson et al, 1997)

The test cases are completed after the test design is done. Test cases should specify the each testing requirements, therefore, is based on the requirements. With the testing technology developing, professionals have noticed the derivation of the test cases are getting more and more since it could significantly improve the testing process efficiency.

However, beforetime, the test engineers always took the task to design the test cases and approve the certain function unit and then discarded after the tested software deployment. If the tested product has the thousand of object units, the test engineers have to design the thousand of test cases to test all the units, repetitiously. This testing work process is the most original process. The product line and product family methodology take some fresh ideas to broaden the test cases usage. From the higher-level perspective to look at the test cases, the focus is moved to the test cases functionality, concerning of how to minimize the function overlaps between the different test cases. If the software design technology could be reused between the different products in the same product line, then, the most likely improved testing process between the different products is the improvement of the test cases design process. The test cases benefit from the product line specificities, which is to reuse the

test cases between the different products in the same product family. Therefore, the reusability is highlighted more and more nowadays. Once a test case is defined, it should be reused as much as possible until the corresponding application is discarded. As the illustration of the following example figure (FIGURE 4-6): in the product-engineering domain, the different product with the same application implementation (sending/receiving short message function) can achieve the same product function. In this case, the test cases designed from the same requirements can be reused. There is no need to write and design the different test cases for the different product, but testing the same function. Whereas, for the different products (mobile products) with the different application implementation (sending/receiving short text message and camera capture function), it is impossible to reuse the test cases designed from the different requirements. Therefore, from the perspective of how to bridge gap from the product-engineering domain to the application-engineering domain, and how to improve the test cases design has remarkable meaning to professionals now. And the requirement family also is proposed, as a new idea in the industry field.

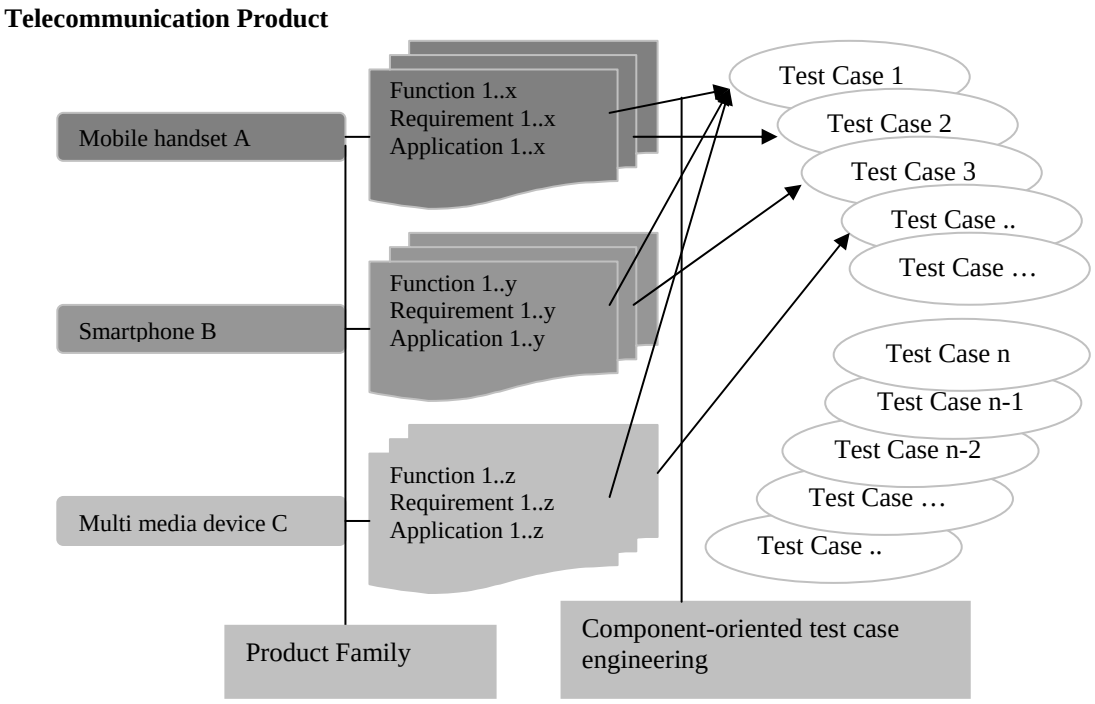


FIGURE 4-6. Test Case Reuse Based on Requirement Relationship

Using the UML methodology to abstract the high-level use case scenarios is the possible solution in the industry field to extract the requirements and design the test cases as reusable as possible. The use case scenarios elicited by the UML could derive the test cases pattern, and then generate the concrete test cases by using the software assistant tools. This design idea simplifies the test cases design phase, and meanwhile, save the design time by using the software tools. The test cases generated by using this method could be executed automatically under the assistance of the software, which is another benefit.

The other concerned issue of test cases is how to do the efficient documentation during the testing work process. The documentation policy and standards should be decided beforehand in the testing planning phase. Test cases form is the one of the documentation contents, in where the test related elements should be written down for the further use. The commonly-used test cases form template includes the following points: test date, tester, system requirement, test cases version, test environment requirement, test function description, test pre-requirement/condition (test configuration), test steps, test expected result and test actual result, etc.

4.2.4 Testware

The next KA is the testware area. Testware in testing work process includes the actual 1) test plan and test cases, 2) the supported software, 3) the data sets that are used to run the tests, 4) necessary documentation (Ericson et al, 1997). In this KA, it primarily includes configuration management and the usage of the testware. The involved aided tool in this KA could differ in the different company, which means it is possible that the same CM tools have to be customized to the local company situation. In addition, the file system, content management system and database management system should be also covered into the testware KA to assist the testing work. The more detailed activities in this KA are described in the following table (TABLE 4-4).

	Testware
Optimizing	Computer aided result checking and evaluation Computer aided testware design Test environment is integrated Experimentation with tools
Risk-lowering	Computer aided risk analysis Testware and software is under effective CM Computer aided regression testing Only mature technology is used
Cost-effectiveness	Computer aided problem tracking, code tracing and test execution Computer aided test case result gathering, recording, processing and reporting Coverage analysis tools are used Hardware and software systems are simulated Databases are used for CM of testware Test tools are evaluated before they are bought and put into action Static analysis tools are used, e.g. to prepare for code reviews
Baselining	Problem reporting is computerized File systems are used for 'configuration management'

TABLE 4-4. Activities of Testware KA (Ericson et al, 1997)

Since the testware consists of a lot of test data and materials, how to find an efficient configuration management solution based on the computer software assistance is rather important in order to increase the working efficiency.

Using database to store the test results is a good software resolution to improve the testware management. The centralized test results are easily for further analysis. Use ready software program to write the test cases and a UML program to design the overall test suites and testcases, for example with U2TP (Schieferdecker et al) is nowadays the popular way for company to try to design the testing process as automatically as possible.

However, here, it is emphasized that professionals should use as much as possible standards to improve the testware performance. As the element of the testware, for instance, professionals could use the IEEE 829 standards to guide the test plan. For test cases and test report, there are ISO 9126 and IEEE 829 to provide guideline on how to guarantee the quality.

4.2.5 Reviews

The last KA is the reviews key area. Reviews is only a generic name for this area, embracing all techniques involved readings or the visual inspection of the software documents (Ericson et al, 1997). The examples of the reviews techniques are inspection, technical review, walkthrough, peer review and code reading (Freedman and Weinberg, 1990). The different review activities, techniques are used in the different level of the Test Improvement Model, as the following table (TABLE 4-5) described.

	Reviews
Optimizing	Review techniques are continuously improved Selection of team and technique is based on facts
Risk-lowering	Testware and test cases are reviewed Review techniques are evaluated Review processes, products and resources are measured
Cost-effectiveness	All review staff are provided training in review techniques Checklists, scenarios, etc. are customized and used Design and code documents are reviewed The organization has some review techniques to choose from
Baselining	Requirements are reviewed Review leaders and management are given training in review techniques Training and support are bought in for at least the first project Standardized, documented review techniques are used

TABLE 4-5. Activities of Reviews KA (Ericson et al, 1997)

It is worthwhile to highlight here the importance of the test plan review. The test plan review action could correct the test action in time, and help project team to collect the successful experience and failed lessons in time. Therefore, there must be positive influence to the later project upcoming jobs. From this perspective, it is natural that the Test Improvement Model KAs have sort of relations, not only to the test job, but each other.

Test cases review action done by the project team members and customers facilitate the better understanding to the customer requirements. It is easy to communicate the test cases to the team because of the project interface. This will verify the test cases developed by the test team and improve them if it is required, before the actual testing begins.

The artifact review action is not all for the review work. The test operator review is also necessary. In any team, continual evaluation of the effectiveness of each test team member is important to ensure a successful and worthy test effort. Experienced professionals perform the test operator review. The review examines a variety of areas, including the types of defects generated, and the number and types of defects missed. It is never good practice to evaluate a test operator's performance by using the numbers of defects generated alone, since this metric by itself does not tell the whole story. Many factors must be considered during this type of evaluation, such as complexity of functionality tested, time constraints, test engineer role and responsibilities, experience, and so on. Regularly evaluation by using the valid criteria makes it possible to implement improvements that increase the effectiveness of the overall effort.

4.3 The General Assessment of the Test Improvement Model

It seems complicated that these five KAs have their own different activities; however, those activities are just suggestion lists to facilitate company work in the theory. In the practice, those activities are possible to be tailored towards the company needs, which will be discussed more in the case study chapter in this thesis.

For Test Improvement Model assessment process, the different company might have the different tailored process. But, as the suggestion from the Test Improvement Model, the assessment is basically done through the interviews, audit and analysis. This is the procedure to determine the status of the company testing work. If there is anything not good enough, the possible development plan will be made. So, the basic workflow of

this activity could be illustrated as the following figure (FIGURE 4-7). The assessment and the improvement processes are two iterative processes until the satisfied extent. As Erison et al (1997) proposed, the development of an improvement plan includes:

- Solution identification—give the maturity profile. Test Improvement Model suggests strategies that can be used to improve the organization. The strategies form a general solution space;

- Solution analysis—the solution space is combined with the needs and visions of the organization. Considering what strategies are relevant, important and cost-effective for the organization reduces the solution space. The possible strategies are discussed and ranked, according to capability, resources, timing and management opinion.

- Presentation—the proposed improvement tasks are combined with the results of the assessment.

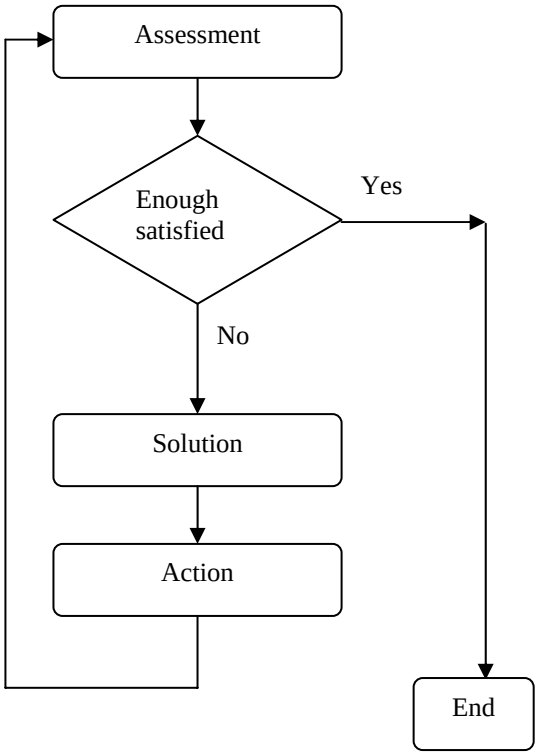


FIGURE 4-7. Assessment Work Flow

Except the Test Improvement Model key areas, there are several aspects tightly related to the software testing, but not mentioned very much in Test Improvement Model. It is no doubt that based on the real work experience; some of the good ideas can be picked and adopted into the Test Improvement Model. In this chapter, the aim is to give some fresh ideas concerning of the Test Improvement Model KAs based on the in-depth review of this Test Improvement Model. The further feasibility research would be approved in the case study phase.

4.4 Summary

Even though the Test Improvement Model has comprehensively covered all the technology aspects related to the testing issues. It was still found that the human-related issues are omitted in the Test Improvement Model. However, in the real work, the human-related issues do have the big impact affecting the testing work performance. The only difference is that in the different working environment, according to the local situation, the human-related issues might be emphasized more or less. However, as a universal model, it should include every possible aspect, so that the users can have a relatively whole guide to help them when they use it.

All the points pointed out above are from the literature review and my own personal experience. Therefore, they are prepared for the investigation in the case study and the further research.

5. RESEARCH APPROACH

This chapter gives the introduction on how to choose the research approach and how to utilize the chosen research approach to conduct this study. It consists of the background introduction to the research approach and the selection of the research approach, research design, data analysis and the study validity research.

5.1 Background of the Chosen Research Approach

According to Järvinen (1999), research approach is divided into theoretical-conceptual analysis approach and empirical study approach. Theoretical-conceptual analysis approach mainly concentrates on the theory creating and testing based on the propositions and the theory axioms by using the logic or rules. Therefore, it is deductive. Empirical study concentrates on the theory testing by using the empirical generalizations, which is called as inductive.

Daniel Moody (2002, 15-18) said,

> *"Empirical research methods are a class of research methods in which empirical observations or data are collected in order to answer particular research questions. Empirical research normally starts with some a priori theory, which the researcher develops to try to explain and/or predict what happens in the real world. The purpose of the research is to test the theory and possibly refine it."*

The research topic of software testing is not new, however, since the professionals give the special attention on this topic only in the recent decades (American Society for Quality, 9, 2000), the available information is rather limited. Especially, the information concerning of the software Test Improvement Model is less in this sense. For the Finnish industry, the available experience and literature towards the Test Improvement Model is even less. Therefore, the research mainly explores the possible existing company experience involved in the implementation of the Test Improvement Model at work on the ground of the theoretical Test Improvement Model model. As a result, it is natural to choose doing empirical study and collecting the in-depth empirical data to analyze the existing Test Improvement Model.

As the explication before, the objective of this study is to review the existing theoretical Test Improvement Model and the related testing information, then explore the feasibility and the utilization situation of the Test Improvement Model in the chosen case company. According to the data collected in the case company, the thesis intends to answer the research questions and analyze how the conclusion would affect the Test

Improvement Model in Finnish industry. Therefore, the study goal also determines that it is more suitable for this research to do empirical study.

In this study, the empirical study starts from the prior theory, in which the researcher tries to explain and discover what happened in the real world based on the prior theory. It might be possible to generalize some new ideas or comments based on the prior theory. Therefore, the research aims of empirical study are to review and possibly refine or complement the prior theory (Moody, 2002, 15-22).

Generally speaking, empirical study includes the qualitative research method and the quantitative research method. The qualitative research method refers to the method to collect qualitative data, like text data, image data and video format data and so on. Whereas the quantitative research method refers to the method to collect the numerical data and analyze the data by using statistical methods (Järvinen, 1999).

Referring to Moody (Moody, 2002, 15-22), the most common qualitative methods are:

> • *Case study: observations carried out in a real world setting (e.g. a software development project, an operating theatre in a hospital). The objective is to immerse you in the situation and gain a holistic understanding of the phenomena in its natural setting.*
>
> • *Action Research: apply a research idea in practice, evaluate results, and modify idea (cross between an experiment and case study)*

It indicates that it is more suitable to use the case study methods to the research natural attributes.

Early research of Yin (Yin, 2003) also stated that there are five different data collection methods when doing empirical study. Each method has its own way on the ground of its own logic. The five data collection methods are experiment, survey, case study, archival analysis and history. Different data collection method answers the different research

questions types. According to this categorization, case study corresponds to the “how” and “why” type questions, which focuses on the contemporary events.

In this research, most of the research questions involves “how” and “why” types questions, besides the research subject is the contemporary event, therefore, the case study suits for the purposes and research questions of my thesis.

5.2 Research Design

A research design is crucial to an efficient research study since it is the logic to link the collected data, and the conclusion has to be drawn to the initial questions of a study (Yin, 2003). In this study, the design of the case study was chosen to take the logic role in the thesis to link the collected data and the proposed research questions. The main reason is that the case study approach was adopted for long time as a data collection approach when carrying on the empirical study because it generalizes the theory from the cases, which is more understandable, practical, describing and explaining.

Based on Yin’s research (Yin, 2003), there are five components should be allocated in case study when conducting it:

- *A study's questions*
- *Its propositions, if any*
- *Its unit(s) of analysis*
- *The logic linking the data to the propositions*
- *The criteria for interpreting the findings*

In this research, the most important research question is “*How the Test Improvement Model works in company (Finnish company)”*, around with this key question, it includes

several sub questions towards how to conquer the usual-happened problems in the testing work, and so on. Among those research questions, most of them are about “how” and “why” type questions. In this research, there is no proposition, the reason is because the current literature to the Test Improvement Model practice is very less, and the study likes to put more concentration on exploring the situation in deep depth level in case company, rather than propose and prove any propositions. The unit of analysis in this research is about the several strategies and activities implemented in the Test Improvement Model. How are they actually done and evaluated in the real company domain, especially in Finish company domain. Towards the criteria for interpreting the findings, the theoretical Test Improvement Model will be used to guide the empirical research, and the data get from the empirical study will be used to compare with the theoretical model. Finally, the aim of the research is to try to evaluate that whether the current Test Improvement Model is suitable for the Finnish company, is there any improvement could added to the Test Improvement Model in order to make it more suitable for Finnish company based on the empirical study and the literature review. Thus, the research is aiming at exploring and defining the generalization.

The case company was chosen in the case study is because of the following reasons: 1) it is a typical Finnish middle-size company; 2) it has many years working experience in the software testing area, which is the researched area; 3) the company has the testing improvement model, which is very similar to the Test Improvement Model. Due to the facts that 1) in Finland, most of the companies are the middle size company; 2) Finland is famous of its high-tech and telecommunication technology, most companies are related to software development and testing; 3) most of the software development and testing companies are seeking for the software testing improvement process. Therefore, it is very natural and practical that the case company is chosen as the representative to be interviewed. However, the Test Improvement Model is a big model covered the different testing maturity levels, in where the different, but main testing areas are involved as well. Thus, basically, this research would prefer to concentrate on one or several aspects of the whole Test Improvement Model in the company case study. The collected data could be used to make generalization based on the representative case company. The data are collected from interview in this case study. As Adams R. G

(Adams, 2003) said, the interview data collection method could support the necessary interactional effects between interviewer and interviewee that can influence both the quality as well as the quantity of the interview results.

On the ground of the reality that the current literature about the research subject is very limited, the interview was planned to set as semi-structured interview. The strength of this interview type is that the interviewer could first of all explain the purpose of the study, discuss the interview and respond to the interviewee questions they might have. On the other hand, the semi-structured interview could provide a central interview soul. Around the soul, the data could be collected as much as possible for the convenience for the later data analysis and generalization.

The major strength of the data collected from the case study is that it has possibility to use different sources of evidences (Adams and Schvaneveldt, 2003). Therefore, besides the data collected from the interview, the amount of secondary data was collected as well as from the literature review, articles, papers, online resources and lectures. All these data together facilitates the research on my subject.

5.3 Data Analysis

Data analysis is the rather important step when conducting the case study. The following steps are adapted in the interview process. First, the interview summary has to be done right after the each interview with the interviewer. This is the necessary step to analyze and identify the validity of the interviewed data. Secondly, transcribing the interviewed tape into the real text, and re-send the transcribed data to the interviewer for the authenticity. In order to support the analysis, several quotations from the interviewees are included into the case study chapter.

The interviews were conducted in English according to the real language situation of the interviewees. All of the interviews were recorded into tapes.

The interviewed data are confidential and not included in this thesis according to the requirements from the interviewees, except there are some necessary quotations.

5.4 Quality of the Study Design

According to Yin (Yin, 2003), the case study must maximize four aspects of the quality of the design: construct validity, internal validity, external validity and reliability.

Towards the construct validity, Yin (Yin, 2003) suggests three tactics to test it for making the case study more objective, they are: 1) use multiple sources of evidence, 2) establish chain of evidence and 3) have key informant review draft case study report. In this study, in order to avoid the subjectivity from the interviewed data, the multiple sources of evidence were picked. The resources include the literature review, empirical study design, which includes the interviewed questionnaire, and secondary data that are relevant to the Test Improvement Model study exploration and research. First, based on the theoretical model and the current Test Improvement Model information, the interviewer designed the interview questionnaire. Then, the interviewer's supervisor approved the questionnaire before the interview going on. Secondly, according to the Test Improvement Model key areas, ranging from the management level in company to technical level, the interviewees were chosen according to their working experience, current working position, and the knowledge scope. Thus, the interviewees are determined to cover the testing line manager, testing site manager, project leader and the testing engineers. Finally, about the interviewed report, the interviewer send back the report to interviewees after interview to verify its accuracy and validity.

Internal validity concerns the issue that causal analysis and explanation offered by the theory reflect the reality at the moment of the case study. Yin (Yin, 2003) suggests the following tactics to ensure the internal validity of the case study: 1) do pattern matching, 2) do explanation building and 3) do time series analysis. In this research, it tries to guarantee the internal validity by following the rules: 1) every interviewee was asked the almost same questions, except the management-level questions and company history related questions only went to the manager since his working experience in this case company is the longest among the interviewees; 2) the theoretical Test Improvement Model was used as the basis to explain the interview aims to interviewees and all the related questions are based on the Test Improvement Model.

External validity refers to generalization validity from the case study could be efficient to other cases. In order to do this, Yin (2003) guides researchers to do the following tests: 1) use replication and 2) logic in multiple case studies. In this research, the external validity is ensured because the generalization was concluded from the representative Finnish company. Besides the same questions went to the different interviewees and the answers were collected to make comparison. Moreover, the multiple data resources were used in the research. So, to the most extent, the external validity is achieved.

Reliability claims the most minimal bias and the most maximum accuracy from the data. Yin (2003) asks to use the case study protocol and develop case study database. In order to perform this, I used recorder to record all the conversations during the interview process. In addition, all the records are kept as the database for the further needs. The interviewees were chosen from the different position in the case company is also considered from the perspective of reliability of case study.

Therefore, I believe the quality of the case study is valid.

6. CASE STUDY— THE TEST IMPROVEMENT MODEL USABILITY AND IMPROVEMENT

This chapter presents how the case study was planned and conducted. It is made up of the case study background introduction, case study phase description, data analysis and the summary.

6.1 Background

In this study, a single case study was chosen as the data collection method for the two major reasons. The first reason is that the available resource in the research field is very limited; the second reason is that this case study intends to make the explanatory investigation on the Test Improvement Model. Case study is suitable for this intention. However, considering the research topic is wide and the duration of the case study might take, the single case study was planned. The case study was designed to give answers to the research questions. If needed, the similar, but multi-case study will be planned in future research.

The case study process was designed based on the literature review and the Test Improvement Model. The case analysis was carried out according to the suggestions from the research methodologies (Yin, 2003, Moody, 2002, Järvinen, 1999, Adams and Schvaneveldt, 2003). The case data analysis intends to give the conclusions that if the Test Improvement Model suits for the Finnish company or not. If the model is not suitable for the company, what kinds of additions have to be considered?

6.2 Case Study Phases

The company *BH* Oy was chosen as the case company. It is a middle-sized Finnish wireless technology company with some operations in Europe. *BH* Oy has more than 15

years operation history in the wireless technology and software production service area. *BH* has separate wireless technology development and service lines. In the development line, the business concerns telecommunication software development, wireless technology research and related areas. Its service line focuses on telecommunication technology testing, including testing of wireless protocol, systems and radio signaling. This company was chosen because of its extensive years in this industry, its software testing knowledge and its professional experience in the research area.

The interview was designed following the suggestions from Creswell (Creswell, 1997). First, identify the respondent based on the research purposes and the respondent's knowledge. In the case study, five respondents attended the interviews. They are, respectively, a software test line manager, a test site manager, a project leader and two testing engineers. The five respondents were required to have the extensive management knowledge and technical experience in the same software testing area. Second, determine the interview type. The interview was performed in a semi-structured style. It aimed to get the information as much as possible via the open discussion style. Because the available factual case information on the Test Improvement Model is fairly limited, the interview was designed and performed in the semi-structured style. The respondents do not need to stiffly answer the interviewed questions. They could talk their ideas as many as possible based on the heuristic questions from the interviewer (Yin, 2003). This is helpful for the interviewer to get the useful information to answer the research questions. Third, determine to conduct one-to-one or focus group interviews by using the enough record equipment. During the interviews with the respondents, the interview recorder was used during the process of the conversation between the interviewer and the interviewees. So, the original information resource was saved for the further needs. The interview was conducted one by one. The reasons to take the one-to-one interview are: 1) the respondents have working experience in the different testing areas. Taking one-to-one interview could be more practical to get the more exact information from the different aspects; 2) there are a few questions about the Test Improvement Model strategies and activities in the questionnaire. These questions are different to manager and engineers because the experience from manager and engineers might different. The one-to-one interview could more effectively collect the information from them; 3) if

take many respondents together to have interview, someone's opinion might affect the others' idea; 4) one-to-one interview is easier to be arranged than group interview. The company has to keep only one person absent from the work every time. Therefore, the one-to-one interview was chosen and carried out based on the research design and the real busy work situation. Fourth, design the interview questionnaire. The questionnaire was designed beforehand on the ground of the research methodology instructions (Yin, 2003, Creswell, 1997). Then, the list was approved with the supervisors. All of the questions are open-ended questions. Fifth, choose the interview place. In order to give convenience for the respondents' work, the interviews were taken in interviewee's working area after getting the permission from the case company. Sixth, during the interview, stick to the interviewed questions and complete the interview within the specified time. All the interviews were finished within the pre-determined time scale, 1.5 hour. Seventh, translate the citation. The conversations were translated from the tape to paper. Except the small citations, all the translated data were emailed back to the respondents to ask for checking and approval. Any following-up questions after the interview were communicated by emails and phone calls between the interviewer and the interviewees. Finally, the summarized data were re-sent to the respondents at last for the final checking and content approval.

6.3 Case Data Analysis

In this thesis, the main research question is how to evaluate the Test Improvement Model in Finnish company. The case data analysis aims to answer the main research question. At the same time, the sub questions proposed in the previous chapter will be analyzed. In order to follow the thesis structure and keep the research focus, the data analysis will take the order that all of the Key Areas of the Test Improvement Model will be analyzed one by one. It is known before the interview that the case company has its own testing improvement model. But, the interviewees read the provided Test Improvement Model papers, found that the Test Improvement Model is pretty much like the model designed by them. Therefore, the Test Improvement Model and all the KAs' activities were checked in the interviews. The exciting finding is that many of the

aspects in the case company's testing improvement model are similar to the issues mentioned by the Test Improvement Model. This obviously gives weight to the case study that the investigation of the Test Improvement Model in the Finnish company will be a good reference case study in this research area. Hence, it is worthy seeing closely to the Test Improvement Model in the interviews and case analysis about if the Test Improvement Model can be used to the real work. Meanwhile, during the data analysis process, there are several interesting points found out, such as there are some more KAs used in the case company, whereas they are not addressed in the Test Improvement Model.

The interviews were carried out systematically according to the pre-defined procedures. The interview objective was around the Test Improvement Model KAs feasibility and reliability in the case company. During the interviews, the interviewees checked all the activities of the Test Improvement Model KAs, and compared with their own model's activities (the results are given in the later chapter as the form of table). They assessed the practicability of the Test Improvement Model in their company environment. During the interview process and the after-interview analysis process, two extra key areas were found. They are environment KA and tester KA. They are new KAs from the Test Improvement Model perspective, but the case company is taking these two KAs into the real work. Because the emerging of the additional two new KAs, some new activities are emerging as well. If the two new KAs can be added into the Test Improvement Model with the corresponding activities, the whole model will be slightly changed. However, it certainly needs to take the further research and prove that the new KAs can be generalized from the case study.

6.3.1 Discussion of the Research Questions

Because the case study means to answer the research question, the research question discussion is put at the first place in the case data analysis chapter. The interview data gave some of the research questions very straight answers.

How to avoid the usual failures or problems emerging in project, if they exist?

At real work, according to the working experience statement from the interviewees, the big headache of the company testing work is that the testing engineers do not know what are the usual problems that the testing work will meet before the work starts. So, they have no clue that if it is possible to avoid the problems before they actually happen; if they happen, they have no idea how should they resolve them.

To this question, the interviewees' answers give very good resolution on how the case company handles this problem. The answers are from the interviewees, who have the different positions in the testing line. Their answers elaborate the testing "problem and fight-back" working procedure in the case company.

The test engineer said there are sort of 'easily-meet' project specification related problems at the testing work. For example, when the test engineers execute the test cases, they might find the problematic test cases. How are they aware of those problematic cases before they execute them? The team members try to identify them and put them into a separate pre-agreed log file saved in the company server during every testing project. So, the test engineers should check from the log file that if the cases are valid or not before executing them. So, before the project starts, the team members are aware of the possible problems and resolutions.

The project manager said they have a "problem inspecting and resolving" procedure running among the project teams. For example, in the mobile testing project, it is easy to find that the DUT (device under test) information is missing from the test database. How to resolve this problem if it happens? The manager continued saying, the communication efficiency is fairly important in this case. Sending emails, making phone calls, walking to find the responsible persons are recommended strategies when they need to communicate with project teams. After the problem was resolved, the related documentation has to be done as a log file with the detailed date. Moreover, it is understandable the test engineers sometimes meet the problems made by themselves.

For example, the DUT information is in somewhere, but they could not find it for some reason. In order to avoid this kind of problems happened, improving the test engineers' experience, skills and knowledge is rather important. The necessary DUT information checking and even double-checking has to be done before the project starts. The routine meeting in every week and the constant project training also guarantee to decrease the problem occurrence rates.

The software test site manager said from the perspective of the whole site, the easily happened problems normally comes from the two factors. The first factor is that the information from the testing subject is wrong, which cause the testing work is done in wrong way. Or the project information is not updated in time, which causes some waste of project time and work. The second factor is that the test engineers sometimes have the wrong operation, which causes the test problems. In order to avoid the problems, they usually emphasize that before and project starts and during the project going on, the communication between the customer and project team and the communication running inside the project site should be as smooth as possible. The necessary information related to work should be put somewhere, where is accessible to the related persons at any time.

The software line manager said, in the company level, they have designed the certain problem resolve process. But, the project leaders should decide the detailed process implementation in the project team. The process is:

Recognize Problem-->Report Problem-->Fix Problem-->Prevent Problem.

How to evaluate the Test Improvement Model performance in Finnish company? Is there any restriction the company has to notice when they use the Test Improvement Model?

This is the main research question in this thesis. The detailed analysis is made in the following sections.

The software test line manager mentioned, "*I have noticed that in the Test Improvement Model, the human side issues are not covered very much. But, I would hold my opinion that in Finland, this is not an ignorable issue. We have to take care of our people, try to offer the best service and benefits to keep them motivated all the time. The money could not buy the loyalty. Only by providing the best working environment, the people would work together and lead the company going forward…*"

6.3.2 Organization

When investigating the organization KA, every interviewee checked the Test Improvement Model organization KA activity list. The comparison result is illustrated in the following tables. The red mark means the item is in plan to be or has been implemented in the case company. The item without red mark means that company has not implemented the item. After checking all the activities, the interviewees gave their own opinions and comments on some special and additional points.

	Organization
Optimizing	Testers are part of multidisciplinary teams ✓ Continuous organizational improvement ✓ Groups exist for process improvement ✓
Risk-lowering	Test sensitive resource allocation and task planning is performed at all levels Testing function strives after support from all levels of management ✓ Tester involvement in all development phases, and development involvement in testing ✓ Personnel rotate between development and testing ✓
Cost-effectiveness	Testing function is co-located ✓ Key people work full-time on one project and resources are not shared Independent testing function ✓ Team structure is based on resource stocktaking and project needs ✓ Responsibilities are clearly defined, documented and understood ✓ Test group has a dedicated project area, and proper tools and utensils ✓ Training to meet organizational needs ✓
Baselining	Testing function is organized according to documented standard ✓ There exists a test leader and a core test team ✓

TABLE 6-1. Investigated Organization

The case company implements a set of pretty good organization activities during the testing process. The necessary support from the company level is given as much as possible. Compared with the Test Improvement Model organization Key Area activities, there are some more activities that the interviewees commented. It was found from the interviews that the extra organization activities that the BH is doing are very culture-oriented, which means that it might happen only in Finland. But, because this study is done in Finland, and those activities are also belonging to the testing related issues; as the additions to the Test Improvement Model, these comments were analyzed comprehensively. The collected data mainly reflected the following four points: organization prestige, organization model, organization cost control management and organization human resource management.

Organization prestige. In Finnish ICT field, there is a commonly accepted business rule that if company has the international accreditation or certification on its business field, the company has the better competence and reputation in the same business field. This business rule affects all the companies try to get the certification while they are working. As a result, arranging the corresponding auditing and applying for the certification from the authorized institution become popular. During BH's 15 years operation history, the company has got several international certifications in the different technology and business areas. One of the accreditations that BH got from the FINAS is for the BH software development and testing line, which accredits that the BH's software development and testing process is conformed to the ISO17025 standards. Meanwhile, because the certification demands very much, it keeps the BH's leading prestige and competence in the international market.

The interviewee stated:

> *Because the certification is fairly important in the sense of keeping company reputation and competence, we got some in the wireless technology area. Meanwhile, while we are working, we try to do everything based on the related international quality standards (ISO17025) to improve the company working efficiency. We have had a very good scope in the software development line, we are continuing developing it in other lines as well. (Software Line Manager of BH, Salo, 2004)*

To the same issue, the Site Manager said:

> "*All of our labs here are working according to the rules of the quality standards ISO 17025. The working procedure here is followed by the customer's requirements, which includes the customer quality standards and the pre-defined international standards.*" *(Software Test Site Manager of BH, Salo, 2004)*

As the conclusion, if a company intends to improve the software testing process, taking the international standards as the reference to improve the working process is necessary. "The certification will not fool market and customers", this is a very popular slogan of many Finnish IT companies. Therefore, in the Finnish company, getting the certification is taken as an important additional activity to the original Test Improvement Model Organization KA activity.

Organization model. According to the literature review in the previous chapter, organization model has the certain impact on company working efficiency. However, if the organization model also has significant impact on the testing process? The organization models proposed by Ahonen at el (2003) were investigated in the case company.

> *In our company, the organizational model works more like sort of mixture of the team model and the pool resource model. Based on our business needs, we think this ways is the best way for us. From the company scope point of view, we have the different project teams, which are responsible for the different kinds of projects, like hardware design, RF testing, etc. Whereas, in the customer site premise, the BH site works more like the resource pool model. (Software Line Manager of the BH, Salo, 2004)*

> *"BH customer site, especially the software testing group works as the resource pool model. The reason is for better meeting the customer requests all the time. The resource pool model let us work more flexible and free compared with the rigid team model." (Software Test Site Manager, Salo, 2004)*

From the information got in the interviews, the advantages for BH to pick the team model are: 1) each team has the certain technology competence and project experience. It makes the further development on the same competence easier and more focused. The team model has big influence on helping the company develop its competence and expand in the same area; 2) the cooperation in certain team is fluent by choosing the team model because the team members know each other very well; (*Project Manager of BH, Kaustinen, 2004*) 3) each team has the certain scope of the tasks, which helps company standardize and simplify the working procedure easier. The management complexity is decreased because each team has its team leader. To some extent, the testing efficiency is improved because of the simpler management. However, as the experience competence is more important than the technology competence on software testing area, the team model is helpful to improve the test engineer's experience quickly and specifically, which is naturally helpful to improve the software testing efficiency.

But, the limitations by using the team model were also clarified. In small company, the team model is not so easy to run smoothly, because small company might be lack of the enough human and financial resource to build and keep the independent teams all the time. Moreover, although the team model is good for developing the each team's competence, it might baffle the company information exchanging and sharing at the same time. If the team has the conservative idea to keep its leading competence, the team members might not be able to share the information with other team members. Another disadvantage of the team model found in the interview is that *"according to my 10 years working experience, it is possible, that things are done in certain pattern in certain team, lack of the innovation and flexibility, which is obviously not a good thing for company lasting development."(Project Manager of BH, Kaustinen, 2004).*

Team model has been used in many companies. But, the resource pool model is also widely used in the Finnish companies. The main advantage is that company can work very flexibly by choosing the resource pool model. All the resources are put into the resource pool, which gives the project team possibility to rapidly and smoothly adjust the resource at any time according to the project requirements and the demands from the

customer. Meanwhile, resource in the pool requires being movable at any time. In details, the human resource in the pool is demanded that every candidate has to have the wide knowledge, which can enable the quick implementation that any candidate can do the resigned task in short time. It is very normal that in the current business environment, everything is rapidly changing. The human resource has to be motivated enough that they can take any tasks in the short time to meet the requirements. During the investigation, it seemed that the resource pool model works very well in the case company, especially, for outsourced testing group. In the testing group, every test engineer knows more than one often-used testing technology. So, they can be arranged and delivered to the different testing platforms and taking different testing responsibilities at any time. The disadvantage of using resource pool model might be that every candidate in the pool has the broad knowledge, but none of them has the expert knowledge in one specific area. Organization model helps improve the software testing process. Therefore, the organization model is an addition to the Test Improvement Model Organization KA activities.

Organization cost control. Cost control is a very sensitive issue during the company working process. Company always plans to use the less money, but do the more and better things. The Project Manager stated as following:

> *"In our company, each project team has its own project budget before the project beginning or in each account period. During the project going on, if we need the more financial resource out of the budget, we would make the application to the line manager… "(Project Manager of BH, Kaustinen, 2004)*

> *"If the project manager applies for the more financial resource, we take it as investment. If the investment is below the limit that I can manage, I could make the decision if I permit the application or not. If it is beyond the limited extent, I would need to make the application to the company board meeting for the discussion… " (Software Line Manager of BH, Salo, 2004)*

The certain hierarchy relationship ensures that the company working procedure is based on the certain company rule. For the different office site, the site manager is authorized by the company the certain rights to manage the cost. The manageable cost could be fluctuated in the agreed extent. The Software Test Site Manager stated:

> *"In our working site, the biggest cost is the employee's salary. All the labs' equipments are belonging to the customers. They will take care of all the financial expenditures related to the labs. So, to us, the financial pressure is not much, we just try to keep finishing our work…"*

BH sends out one of the testing groups as the outsourced service to the customer company, where BH only takes care of setting up the human resources pay-roll issues. After the site begins to work normally, all the extra costs brought by the working issues will be charged from the customers. This kind of "body-shopping" work style is more and more popular in the Finnish industry due to its easy management and less cost. However, there are still some problems found by using "body-shopping": 1) how to keep the employee working in the customer premise with the mental happiness. In reality, the employees who are working in the customer site often complain that they are mentally lost. They do not know which company they are actually working for. They are confused because they are paid by BH, but working in different company. They have no access to get information from the customer intranet. Moreover, they could not get information from the BH intranet either since there is no BH intranet access in the customer site. So, they are blind of communicating within BH. Meanwhile, they are external employee in the customer site, saying in the working badge the red marker "external". There has always some restriction to "external". Even in the small issues, as if lunch, they have to pay more because they are external employees. Such kind of issues usually happen, which make the BH persons confused that if they should consider to change job. In this sense, BH feels embarrassed all the time because they always claim they care their people. It is still in negotiation and planning, BH has to listen to the employees, and try to find some compromise in this issue.

Organization human resource management. Human resource management is a fairly difficult task in any company. In the Organization KA statement of the Test Improvement Model, training is taken as an important activity to improve the testing work efficiency and human resource management. The same conclusion is also summarized from the interviewees. They explained how the organizational effort could contribute to improve the testing work efficiency.

> *"We organize the different trainings for the employees at every certain time. These trainings could be tightly related to the ongoing or upcoming projects, they also could be the general working area knowledge trainings. For example, for our software testing group, we organize the beginning level testing related trainings for the new employees and the higher level trainings for the old employees. For the design group, we mainly organize the mechanic design related trainings for them. For the management group, we concentrate on how to develop and shape their management capability to meet the company working needs." (Project Manager of BH, Kaustinen, 2004)*

When the company organizes the training, it has to consider giving all the employees as many as possible the useful trainings. The trainings are better in the related, but slightly different testing area. But, the company does not expect the employee to know every detail. The purpose to arrange the training in this logic is that in case the work needs, the employees have already mastered the basic knowledge as the starting point to take over the new job or do the backup of the new job. This is especially important for the testing work. From the BH wireless protocol testing experience, the testing requests usually have to be done in the different testing platform systems. So, when the certain platform is lack of test engineer because of the sick leave or some other reason. Another test engineer could do the same job to backup the vacancy in time.

Human resource management also includes that each employee has a personal archive in the company server, where the employee's CV, competence list, evaluation report and the personal development plan are saved. This personal archive effectively gives the company the background information of the each employee. Beside of this personal archive, the company has personal discussion once a year. In this discussion meeting, the chosen candidate will discuss directly with the line manager about the work summary in the past year and the future personal development plan. By doing this, the employee has the certain goal in the future certain time period. It helps develop the employee very well. It is called as Competence Map Methodology inside of the company.

In the thesis research questions, there is a question towards how to build up an efficient testing group. The interviewees answered this question: 1) according to the project

requirement, the project leader determines what kind of project group has to be built; 2) according to the project group properties, the project leader should choose the team members based on the personal archives. BH's philosophy about how to compose the project team is: put the beginners in the project team, who are led by the senior level professionals. Meanwhile, the senior level professionals could help the beginners to quickly get used of the company work. Properly mix the members, who have the different experience background together can somehow give the different gifts, skills and fun to the project team.

> *"To my experience, I personally like to mix the employees who have the different education and working experience background together since the mixture team can give the team work more fresh ideas and gifts, which helps working better and fun." (Software Test Site Manager of BH, Salo, 2004)*

During the interview process, how to motivate testing engineers is a hot topic. Software testing is more like an experience-oriented job. In this area, how to motivate and keep the old employees working are rather important for ensuring the testing work well done. Compared with the company that is always losing the people and seeking for new employee, BH prefers to pay more attention on develop and improve the existing employee's skills. In order to do that, the company needs to motivate the employees all the time. *"For me, I believe money can not buy the employee loyalty. Working in a relaxed, enjoyed environment is much more important. So, I prefer to motivate my people by letting them more engaged into the project work all the time and letting them really can consider what they would like to do next."* The Project Manager stated.

Other interviewees continued by saying:

> *"I motivate them by giving them more trust, which shows in letting them attending the project management more and more, changing their working contents every now and then, keeping them always having the passion to the work. And of course, if the customer has the very positive feedback to the individuals, I will tell them directly." (Software Test Site Manager, Salo, 2004)*
>
> *"We have the personal development discussion (called as PI) in company once a year. It happens between the employee and the line manager. Based on his/her work achievements, the line manager will decide if the employee gets the promotion or not, if the employee is qualified to change the working position*

from this role to that role after the discussion with him/her."(Software Line Manager, Salo, 2004)

Based on the interviewees' statements, the BH's testing process maturity level is located on somewhere between the baseline level and the cost effectiveness level. *"Even though we do not have the Test Improvement Model as the testing process improvement model, we believe our current working model is more or less very much like the twin of the Test Improvement Model. It is very good to know the Test Improvement Model and we will consider to use the Test Improvement Model since it is so similar with our working model and it is more professional and focused."* The Software Line Manager said. The findings indicate that BH has implemented a very practical working model to improve the testing process. Compared with the Test Improvement Model description in the article, the activities mentioned by the Test Improvement Model and meanwhile implemented in BH are proved to be very practical and useful for the company. For those higher maturity level activities, which are not been implemented yet in BH, the Test Improvement Model gives a very good guide on how they should be considered.

From the investigation of the Organization KA, it is easy to find that in BH (as a Finnish company representative), the company pays a lot of attentions on the human side issues. The company considers how to make employees feel happier, how to make employees more competitive and how to motivate employees working for them. This is a rather tricky rule that company has to consider in Finland.

BH has its own organization related activities. Some of those activities are exactly same with the activities in Test Improvement Model. Some are not. After the investigations, the interviewees also checked all the activities of the Organization KA mentioned by the Test Improvement Model. They all agreed on the conclusion that the suggestions from the original Test Improvement Model are helpful for them as well. The following list is the collections of the gathered activities from the case company:

Optimizing:

- Keep checking and updating the company certification.
- Keep motivating the testers
- Let testers engage in the real testing planning and management work

Risk-lowering:

- Get the international certification from the authorization bureau
- Daily work is based on the standardized working procedure
- Keep motivating the same testers at work, do not let your people consider changing company

Cost-effectiveness:

- Have the certain cost management methods and procedures
- Have the certain person set the budget plan and master the budget
- Track the budget all the time

Baselining:

- Decide the certain organization model
- Train the testers get to know the real work by practice.

6.3.3 Planning and Tracking

Testing planning and tracking is claimed to be very important before any real testing work starts. Good testing plan usually ensures the work to be done systematically and precisely. The encountered problems during the testing process can be resolved efficiently by using the pre-defined methods mentioned in the testing plan. The test result and test defect can be tracked at any time as necessary to find the cause root.

The investigation of the case company's planning and tracking activities and the Test Improvement Model's planning and tracking activities is showed in the following table:

(The red mark shows that the corresponding activity is implemented in the case company, as the activity of the case company's own activities.)

	Planning and tracking
Optimizing	Models are made of costs/resources, risks, etc. ✓ Planning and tracking activities are continuously improved based on analysis metrics and experiences ✓ Post-mortems are performed and results are duly distributed
Risk-lowering	Risk analysis is performed and influences planning and resource deployment Affected parties approve plans, including test objectives ✓ Fulfilment of overall test objectives is monitored ✓ Planning is performed by an experienced tester/planner Plans change with factual circumstances ✓
Cost-effectiveness	Planning and tracking is computer aided ✓ Generic plans are used ✓ The choice of test methods and test levels is balanced against test objects and test objectives ✓ Progress is measured, and compared against plans ✓ Planning is performed by a trained tester or planner ✓ Planning is performed in an evolutionary manner Resource prediction is done according to documented policy ✓
Baselining	Basic planning is performed ✓ Entry and exit criteria are defined for all testing levels ✓ Test results are documented, and duly processed and distributed ✓ Plans are developed following standards and deviations are documented and motivated ✓ Planning is performed according to documented policies ✓ Plans change with requirements ✓ Changes to the test plan are made according to a documented policy ✓

TABLE 6-2. Investigated Planning & Tracking

During the comparison process, it was found that BH has two candidate testing planning and tracking resolutions. Which one is the suitable resolution can be decided according to the real project situation. One resolution is that BH uses the BH self-designed testing planning and tracking method. Usually, the project manager makes the testing plan, which covers the testing schedule, testing financial estimation and so on. Before the project starts, the testing manager evaluates the validity of the plan. The testing tracking is enabled under the assistant of the testware.

> *"We have the certain person to take the responsibility of building the testing plan since drafting the testing plan needs certain experience. It can not be done by anybody. We usually nominated the testing manager to draft and evaluate the testing plan before the project can be put into the kick-off meeting." (Software Line Manager, Salo, 2004)*

The other resolution is uncertain, which means according to the project needs and customer request, the testing plan will be roughly decided by the customer. Then, the project manager gives the detailed plan proposal. The customer checks the validity of the testing plan proposal before the testing starts.

> *"We have our own daily work plan, but, of course, our own plan follows the general requirements from the customer's request. If the customer request has changes, we will also correspondingly change our internal plan as well." (Software Testing Site Manager, Salo, 2004)*

The interviewees emphasized that the professionals should join and make the testing plan because the professionals have more experience on how to make a more accurate testing plan. The professional plan could significantly help save the project time.

> "*However, we do find that lacking of the systematic training for the testing planner is a disadvantage. They usually have no any clue how they should start to do when they meet some new case, whereas the old experience is unable to let them start quickly. " (Software Testing Site Manager, Salo, 2004)*

The site manager indicated that if company does not organize the enough specified testing planning training for the testing planners, the problem will occur that the testing planners do not know how to handle when they meet the new cases.

During the interviews, there are a few points were pointed out by the interviewees as the important issues for company to check before and during the testing work going-on.

Documentation. How to do the documentation work? What should be documented and how to manage the documentation? In a company scope, all of the testing related inputs and outputs should be documented during the testing process, including the man-made documentation and the machine-generated documentation (for example, the testing plan, testing report, testing change requests, testing approval, test result, test defects, test cases). All the documentations should be kept in a safe, but accessible place. The necessary software should be used to assist the management of the documentation.

> *"In our company, we have the special project folders, where we put all the project related documentations, from the project request report to the project delivered report. Project folder is open to everybody in the company, but the different rights are authorized to the different users in the company. According to the company policy, each project folder should be kept in the company server for at least 10 years."(Software Line Manager, Salo, 2004)*

The same interviewee supplemented:

> *"The documentation itself has the different template. If the customer has no special request, we will do as the BH internal uniformed template and standards. For instance, the testing planning documentation is based on the IEEE829 standards. "(Software Line Manager, Salo, 2004)*

It answered the research question proposed earlier that *How to do the documentation work during carrying on testing project and what should be documented* in a practical manner.

Tracking. Test results and defects tracking are fairly crucial when executing the test cases. For instance, if the test case execution result is inconclusive, it is hard to know what is the failed reason by only analyzing the test process. At this time, the test result tracking can enable the test engineer trace the test case running procedure. It is helpful to find the cause root. Obviously, this tracking process has to be done under the assistance of the relevant software. As well as the test results need to be tracked, the test plan also needs to be tracked all the time. The test plan has to be scheduled based on the agreed standards or rules. On daily basis, the certain persons should check the test schedule against the real test plan. The aim is to track the testing process. The software used to trace the test cases running process is usually working on at the same time with the test cases execution, but generating a tracing log file. From the log file, the test engineer can tell what and where is the problem. The test planning and tracking work flow summarized from the interviewees is:

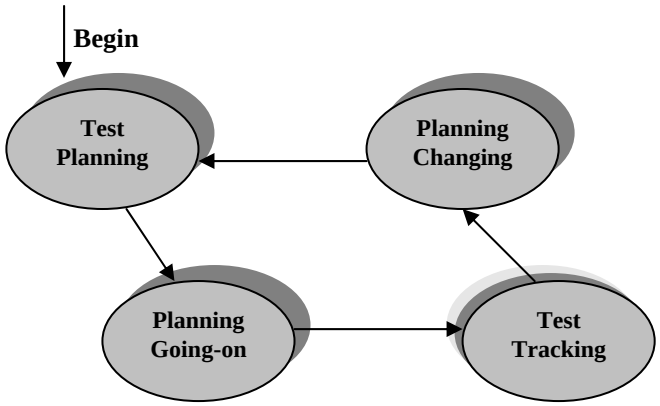


FIGURE 6-1.Testing Planning and Tracking Working Procedure

The shadow area around the each step presents the iteration self-checking and adjustment process. The test tracking step has more shadow than other steps is because it includes more elements, the test planning tracking and test result tracking.

So, the activity list covered by the case company of the Test Planning and Tracking KA is:

Risk-lowering:

- Keep tracking the test plan against the real situation
- Keep all the documentation for certain time before any disposal

Cost-effectiveness:

- Customer project documentation standards and procedures have to be discussed and decided altogether with the customers.
- Company has the internal documentation standards, templates and procedures for internal use.

6.3.4 Test Case

The most difficult, but principal Key Area in software testing process is to design and execute the test cases. Nowadays, test case design technology has been developed very well, represented by the test case reuse technology. For instance, in the mobile software testing field, especially, for the UI (User Interface) software testing, most of the functions have the ready-made commercial test cases. They are designed based on the standards, which enable the specific test cases can be reused to test the specific functions for many different mobiles. What the testing engineers need to do when they need to test the different mobiles is to “localize” the test cases into their working platform, which means changing the possible parameters, changing the interface and so

on. In the mobile network emulation environment, it is common that the test case parameters have to be changed due to the slightly difference between the emulated network and the real network. The test cases reuse also happens in the application software area. Sometimes, the different software might have one or two similar function units. In this case, the similar functions should be able to be tested by just one test case. The engineers shall change the related parameters to adopt the test cases into the certain environment. Reuse the test cases enables the company saving much more time and money. It provides big convenience for the software testing technology. The test case reuse is not valid in the situation, where the new software function or new mobile features are designed. The company has to design the new test case or buy from the third party.

The interviewees checked the Test Case activities of the Test Improvement Model and the activities done in BH. The result is showed in the following table:

	Test cases
Optimizing	Test case design techniques are measured, reviewed and improved ✓
Risk-lowering	Test cases are ranked and selected according to criticality Test cases are designed in response to risk areas
Cost-effectiveness	Test cases are designed using documented techniques ✓ Testability aspects influence requirements and design Test cases are recorded and re-used ✓ Test cases are organized with respect to testing levels, requirements, test object, objectives, etc.
Baselining	Test cases are revised when requirements change ✓ Test cases are based on requirements ✓ Test cases are designed and documented according to a documented policy, prior to execution ✓ Test cases are executed according to documented instructions ✓

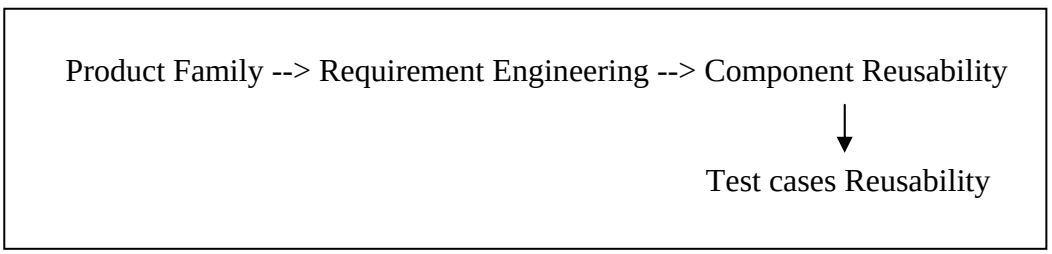
TABLE 6-3. Investigated Test case

BH designs the wireless products. Meanwhile, BH offers the wireless product design and testing services for the customers. This business model determines that BH has its own stand-alone test cases for its products and the ready-made commercial test cases for the wireless product components and its testing services.

> *"The test cases we choose to run are strictly conformed to the 3GPP standards. At the same time, we have the special persons to check GCF standards every day, updating the necessary information in order to let our test cases running exactly right."(Software Testing Site Manager, Salo, 2004)*

It is found from the investigation table (Table 6-3), the case company did not pay much attention on the risk-control side of the test cases design. The reason is because *"we did not have any actual activities to the test cases risk lowering side is not because we were not aware of it. The real reason is that we noticed it, but, lacking of the capability to implement it. But, fortunately, we have the very good researchers and engineers, we are making effort to do it better." (Line Manager, Salo, 2004).*

The Line Manager also agreed that the test cases design is a difficult job. Especially, it is very hard to improve the test cases design. If you intend to connect the test cases with the risks analysis and rank criteria, it is even harder. The test cases reuse is also a hot research topic in the case company. The manager stated that for the mobile signaling and wireless protocol testing, because they are using the special wireless equipments and the attached software, all the test cases are provided by the equipments' vendor. What they need to do is changing the wireless protocol parameters when they are running the different wireless products. In this sense, the test cases get the possibly biggest reusability. This guarantees the test cases can be reused all the time. This idea is surprisingly coincidental with the component reusability idea (Sommerville, 2000, Chapter 14). If the component can be reused from the product family idea by using the certain design pattern, the test cases can be also reused from the product family by using the certain test cases design pattern and application.



Another interviewee continued saying:

> *"We have our own mechanics design and RF products. We began to do this at very early time. So far, we have got enough experience in this field. In this area, we design our own test cases according to our own products features and properties…"(Software Line Manager, Salo, 2004)*

The same interviewee continued:

> *"We have our own self-designed test cases design technology. We are enabled to design the test cases. Sometimes, customer asks us to give the ranked list of the test cases, but we have not developed to do that. We believe it is necessary to give these kinds of criteria."(Software Line Manager, Salo, 2004)*

6.3.5 Testware

The configuration management tool represents Testware in the real company testing process. In the case company, Synergy™ is chosen to combine with certain database system, served as the main CM tool. All the testing related data are put into this CM, the testing data includes the testing planning, testing request, testing result, test cases, test report, etc.

	Testware
Optimizing	Computer aided result checking and evaluation ✓ Computer aided testware design ✓ Test environment is integrated ✓ Experimentation with tools ✓
Risk-lowering	Computer aided risk analysis Testware and software is under effective CM ✓ Computer aided regression testing ✓ Only mature technology is used
Cost-effectiveness	Computer aided problem tracking, code tracing and test execution ✓ Computer aided test case result gathering, recording, processing and reporting ✓ Coverage analysis tools are used Hardware and software systems are simulated ✓ Databases are used for CM of testware ✓ Test tools are evaluated before they are bought and put into action ✓ Static analysis tools are used, e.g. to prepare for code reviews ✓
Baselining	Problem reporting is computerized ✓ File systems are used for 'configuration management' ✓

TABLE 6-4. Investigated Testware

During the interview with the test site manager, he mentioned, "*In BH premise, we are using our CM tool. In customer premise, we choose to use customer-specified testware…*" The file system is simply implemented by database system, without any complicated third-party software. It aims to simplify the management process. All the test-related data are put into the database. Test engineer can access the database, fetch the latest test cases information, including test device information, software information, hardware information and so on. The routine weekly meeting is organized to help the testware information sharing and management among the test engineers and the management. The necessary testing-related information has to be stored in the database for certain time, "*we usually store the project information for 10 years before destroying them…*" The line manager stated.

6.3.6 Reviews

According to the Test Improvement Model, the reviews activities should be done systematically, which means the professionals should use the professional review methods to do the professional effective reviews.

	Reviews
Optimizing	Review techniques are continuously improved Selection of team and technique is based on facts
Risk-lowering	Testware and test cases are reviewed Review techniques are evaluated Review processes, products and resources are measured
Cost-effectiveness	All review staff are provided training in review techniques Checklists, scenarios, etc. are customized and used Design and code documents are reviewed The organization has some review techniques to choose from
Baselining	Requirements are reviewed Review leaders and management are given training in review techniques Training and support are bought in for at least the first project Standardized, documented review techniques are used

TABLE 6-5. Investigated Reviews

In the practical work, company usually let the test engineer or test managers do the review work.

"We have the cross-reviewed method to review our testing work. The test engineers review their work each other. The project manager and the quality manager will take responsibility of the final review work." (Software Line Manager, Salo, 2004)

"We have no specific review technology. We use the inspection review at this moment. But, company is developing and research the better review technology. We hope it can be implemented in the near future." (Project Manager, Kaustinen, 2004)

During the interviews, it was found that in the case company, the review includes the testing work review and non-testing work review. The testing work review means the test cases review, test result review and the testing work audit etc. The non-testing work review means the test engineer review, which should be done by the test engineer, test manager and the quality manager altogether. In BH, since the software testing group is mainly doing wireless product testing, the test engineers' reviews are required very strictly. The test engineers' work affects the testing process in the wireless product testing field. Sometimes, the wrong operation from the test engineers can cause the test cases fails. But, if the test operators do not notice that, they might conclude the test cases has problem or tested device has problem. The case company reviews the test engineers' operation in order to improve the testing efficiency.

"In order to improve the testing work, we think the operator review is necessary. For the different engineer, we can have the different development plan, this has to be done with the personal discussion every year." (Line Manager, Salo, 2004)

6.3.7 Two Additional Key Areas

From the interviews data analysis and the case company working experience, it was found that there are some activities, which are belonging to additional two Key Areas. They are not addressed by the Test Improvement Model, but are implemented in the case company. They are called as Tester Key Area and Environment Key Area. According to the interviewees, especially, in Finland, when the employees are choosing

employer, they care very much if the employer respects and cares them. The human-factors are presented to be very crucial, which affects the career decision. Therefore, the human issues can not be skipped at all. Meanwhile, it was found that environment parameters also have unforgotten influence on the actual testing work, especially important to the wireless technologies. For instance, the wireless component testing sometimes needs the highly precise environment parameters, such as temperature, humid, etc. Therefore, it is summarized that the Tester aspect and the Environment aspect could be added into the Test Improvement Model as the two KAs.

Tester. Tester here means the test engineer, who runs the test cases. Tester is the direct test cases operator. Since the Finnish company takes the human-factor issues very seriously, it is worthy listing the tester as an individual Key Area. Moreover, It is the common knowledge that tester is a principle role during the testing process. So, the work performance of the testers affects the performance of the testing process. For example, some times, it might happen that the failed test case result is actually caused by the tester's wrong operation, rather than testing object problem. Sometimes, it might also happen the bad mood of the tester impedes the tester's work. All the tester-related activities could be elicited and added into the tester key area.

In order to keep the consistence of the new key area descriptions with the original key area descriptions in the Test Improvement Model, the new Tester Key Area is also categorized to the same maturity levels. Some of the items might have been listed in the other key areas, but locating in the different levels. The purpose of listing them here again is to get the consistence between the items and the key area name. The same item has the different function when it appears in the different levels. For example, "tester is part of the multidisciplinary team" appears in Tester KA "cost effective" level is because if the tester can do more than 1 project at the same time, then, ideally, it can save some cost for the project to recruit new person. However, in Organization KA, same item appears in "Optimizing" level. The reason is from the organization perspective, if the same person can be in multidisciplinary team, it is easy for company

to manager and control. Such as this kind of different considerations make the same items appeared in the different KA, but different levels.

Baseline:

- Tester has the individual position role, separating from the other positions.
- Tester is organized according to the testing work needs before the work starting.

Cost effective:

- Tester is part of the multidisciplinary team.
- Tester is trained before to do the real work, under the control of the tutor.
- Tester gets training constantly.

Risk lowering:

- Tester is approved before he/she can take any responsibility.
- Tester has his/her specific testing working task.
- Tester is highly motivated.

Optimizing:

- Tester is trained in training program and real work, approved in real work and authorized to take the stand-alone test.
- Tester masters several testing skills for the different testing platforms.
- Tester has long working experience on the specific testing environment.

Environment. Environment here means the test cases execution environment, including the testing equipment and facilities (hardware and software); testing environment parameters, including testing temperature, humid, noise and so on. It was found in the interviews that the wireless technology testing requires the testing environment

temperature, humid, noise, the equipment calibration and so on. Since the case company tests very many mobile devices and the embedded software, it is highlighted by the interviewees the importance of the testing environment. However, the original Test Improvement Model targets strictly software testing, so that it does not consider very much in the test environment factor. As the conclusion from the case study, it is claimed that the environment KA should be listed as an individual test key area added into the Test Improvement Model, for the purpose to brand the Test Improvement Model covering range.

Baseline:

- The suitable testing environment is set up.
- The necessary hardware and software are accessible.

Cost effective:

- Test environment facilities can be reused for different testing labs.

Risk lowering:

- Test environment is approved before the testing; all the software and hardware are maintained and calibrated at every certain time.
- The testing environment parameters are checked every time.

Optimizing:

- Test environment (temperature, humidity, noise, gravity, etc) is controlled automatically according to the testing environment needs.
- The test environment is approved at every certain time.

6.4 Summary

The case study analysis represents the BH testing function work and improvement situation. At the beginning of the interview, the interviewees have pointed out that they do not use the so-called Test Improvement Model to guide their work, which once made me worried. But, after the interviews, it was found that all the interviewees had read the original Test Improvement Model paper and assesses the suitability of Test Improvement Model to their company and compares it to the company's existing approach. They have their own testing work improvement process, which is pretty much similar to the Test Improvement Model according to our investigation discussion and result analysis. Meanwhile, it was found that many good comments and points are proposed by the case company and have been proved to be useful in the case company. Therefore, due to the principles are consistent between the BH working framework and the Test Improvement Model, it is claimed here that the Test Improvement Model should be also practical to the Finnish company, represented by BH. The more investigation and empirical study are welcome. They can prove that if the Test Improvement Model and the additional KAs are suitable for every kind of Finnish company and other international companies in the future study.

7. CONCLUSION

The main purpose of this thesis was to investigate the Test Improvement Model and analyze if the Test Improvement Model is a practical working model of the Finnish IT company. Meanwhile, this thesis intended to check if there are any additional issues the company needs to take into consideration before they implement the Test Improvement Model to improve the testing process. Based on the research purpose, the research questions and the sub-questions were proposed, and the case study was conducted and analyzed.

First, the comprehensive literature review helped provide the theoretical background to software testing. The software testing process is similar to the software development process and it needs the assistance of proper models. So, the software testing models were researched at the beginning of the thesis. From this, the Test Improvement Model was selected as suitable to make a case study.

Second, based on research methodology analysis, the empirical study was chosen as the right research method in this thesis. The interviews and their data analysis gave the research questions understandable answers in the real industry environment.

It was concluded last that the Test Improvement Model is a standard and relatively complete testing model during the software development process. It is a good and workable model in the Finnish case company. However, the necessary supplements should be made before implementing the model into the real work. On the other hand, the investigated Test Improvement Model is not suitable for all companies. Because of the different scale and different business orientations of the company, it is hard to say that the Test Improvement Model works for every case. For example, for a small-scale company, because it has no need to implement the very strict CMM model when they develop software, the Test Improvement Model will not work well. In another case, if a project team uses XP software development process, the Test Improvement Model does not work either. The Test Improvement Model is supposed to be used under the condition that the company uses the traditional and detailed software development process when they produce software. Furthermore, since the Test Improvement Model requires the demanding and professional support from the management level, the normal company might not have such strong capability to implement this model in the real work.

7.1 Major Contributions

This thesis intended to contribute and improve the software testing process based on the real company case investigation. The in-depth literature review revealed two testing and

improvement models. The comparison and analysis among the models helped choose the Test Improvement Model as the proper model to be studied in the Finnish industrial environment. The literature review not only displayed a few software testing models, but also provided the implications on more issues that might fall under the Test Improvement Model umbrella. Such analysis can be added into the Test Improvement Model after the detailed research in future to approve its generalization.

The literature review of software testing and the empirical case study helped give a better understanding of the Test Improvement Model key areas and activities in the following ways:

- It contributed to give a better understanding of the Test Improvement Model and its key areas in the real practical working environment.
- It contributed to a broader view of the Test Improvement Model in the perspective of key areas and the activities.
- It contributed to help prove the practicability of the Test Improvement Model to the Finnish company. At the same time, there are two additional Key Areas and activities mentioned.
- It contributed to help the company to improve its software testing process through using the Test Improvement Model.

Therefore, the Test Improvement Model was investigated in both theoretical and practical levels. The case study also serves as a comprehensive reference for the Test Improvement Model researchers, who can use the findings presented here instead of performing a new investigation.

7.2 Limitation

Because the literature concerning the Test Improvement Model is quite limited, the opinions and ideas from the Test Improvement Model might not be fully objective. In

order to keep the objectivity of the study, the case data analysis was done on the ground of the full objective truth.

The single case study was chosen as the data collection method in this thesis. It might not be typically enough to generalize the conclusion of the study. Especially, it might not be fully enough to generalize the conclusion that two additional Key Areas need to be added to the Test Improvement Model. In order to try to avoid subjectivity, the following measures were included during the case study conduction: 1) the original Test Improvement Model framework is formatted according to the published literature review; 2) the case study was designed and the questionnaire were planned as universal as possible; both of them were verified by the supervisor; 3) the amount of approved literature were referenced when eliciting the useful information for the Test Improvement Model; 4) the case study was designed avoiding any possible subjectivity; 5) the case study process was taped and verified by the case company before, during and after the case study.

During the empirical study, only a part of the items in the Test Improvement Model were directly covered by the detailed interview questions and discussion. The reasons are mainly: 1) the full investigation and discussion about the Test Improvement Model would have been too big for a master's thesis and; 2) it would also have taken more time and effort from the interviewees than the company was willing to spend.

Due to the request from the case company, all the issues related to the actual competitive testing technology were covered up in this study. Therefore, this study concentrated more on the non-technology related issues of the Test Improvement Model. Moreover, even though the case company is a typical Finnish company, not all Finnish companies will give the same investigation results. However, the main idea should be consistent due to the universality of the case questionnaire design.

7.3 Further Study

This research has investigated the practicality of the Test Improvement Model in the Finnish company. After the investigation, the study proposed two new key areas, which could be added into the Test Improvement Model. In Finland, companies pay a lot of attention on the human-side issues in any industry area, which gave the evidence that the new Key Areas should be added into the Test Improvement Model. It might be interesting to continue this study by conducting more investigations on Finnish and other international companies. Due to the current published information, the Test Improvement Model is successfully implemented in some Swedish companies in Sweden. As far as the thesis was written, there was no additional information on whether this model is working in companies in the other countries. However, the experience and development suggestions from Test Improvement Model implementation in any country are welcome.

In addition, it is found that the case company does not consider very much the risk related issues. Especially, the budget risks and project deadline risks were considered very narrowly. But the risk issues are addressed in details in the Test Improvement Model. The discovery seems to show that the case company is not aware of the risks. What is the reason? In the further study, this question could be considered more. Moreover, other risks that are not mentioned in the Test Improvement Model, such as security and technology leakage risk, are certainly worthy of future study.

The company financial situation directly determines if the company can afford the necessary resources to implement the Test Improvement Model. The case company is a middle sized company, which has enough human resources to support the Test Improvement Model research and implementation. Some small companies do not have this capability. So, what are the minimum capability requirements to research and implement the Test Improvement Model? Can the small size company afford to use the Test Improvement Model or part of the Test Improvement Model? All of these Test Improvement Model feasibility and applicability research questions could be done in the

future for the purpose of letting the Test Improvement Model work better for the company.

REFERENCES

Adams R. G, Schvaneveldt J. D., 2003, Understanding Research methods, 4th edition. Pyrczak Publishing.

Ahonen J. J., Junttila T., 2003, A Case Study on Quality-Affecting Problems in Software Engineering Projects, IEEE International Conference on Software-Science, Technology&Engineering, Nov.04-05, 2003, Israel.

Ahonen J. J., Junttila T., Lintinen H., Sakkinen M., 2004, An Evaluation of Testing in Object-oriented Software Engineering Methods, Proceedings of BIS 2004 (7^{th} International Conference on Business Information System), Apr.2004, Poznan, 449-460.

Ahonen J. J., Junttila T., Sakkinen M., 2003, Impacts of the Organizational Model on Testing: Three Industrial Cases, Proceedings of EASE 2003 (Evaluation and Assessment in Software Engineering), Keele, Apr.2003.

American Society for Quality Forum: Software Quality Professional: 9. 2000, http://www.asq.org/pub/sqp/past/vol3_issue2/.

Beizer B., 1990, Software Testing Techniques, International Thomson Computer Press, 2nd edition.

Bender R., 1996, Proposed Software Evaluation and Test KPA, revision no. 4, http://www.softtest.com/pages/kpabai.htm.

Burnstein I., Suwannasart T., Carlson C. R., 1996, Developing a Testing Maturity Model, Proceedings of the Ninth International Software Quality Week Conference, SanFrancisco, CA, May.1996.

Chrishnan M. S., 1993, Cost, Quality and User Satisfaction of Software Products: An Empirical analysis, Proceedings of the 1993 conference of the center for advanced studies on collaborative research: software engineering – Volumn 1, Toronto, Canada. IBM press, 400-411.

Coulter A. C., 1999, Software Testing Methodology Embedded Software Testing Technique. Lockheed Martin Missiles and Fire Control, Orlando, Florida.

Creswell J. W., 1997, Qualitative Inquiry & Research Design, SAGE Publications, ISBN 0-7619-0143-4.

Daich T. G., 1996, Emphasizing Software Test Process Improvement, http://www.qaiindia.com/Resource_Art/journal_emphasizing.htm.

Eickelman N., Richardson D., 1996, An Evaluation of Software Test Environment Architectures. Proceedings of ICSE-18, Berlin, Germany, 353-364.

Ericson T., Subotic A., Ursig S., 1997, Test Improvement Model – A Test Improvement Model, Software Testing, Verification and Reliability, 7(39), 229-246.

Fewster M., Graham D., 1999, Software Test Automation, ACM Press, 1st edition, ISBN 0-201-33140-3.

Freedman D. P., Weinberg G. M., 1990, Handbook of Walkthroughs, Inspections and Technical Reviews—Evaluating Programs, Projects, and Products, 3rd edition, Dorset House, New York, U.S.A.

Gelperin D., Hetzel B., 1988, the Growth of Software Testing, Communication of the ACM, Volume 31, Number 6, June 1988.

Gelperin D. (1996) 'A testability maturity model', Presented at Fifth International Conference on Software Testing, Analysis & Review, Orlando, Florida, 13–17th. May.

Gilb T., Graham, D., 1993, Software Inspection, Addison-Wesley, Wokingham, U.K.

Hedger D., 2000, Testing Research, Critical Systems Research Group. University of Minnesota, http://www.cs.umn.edu/crisys/spin/talks/hedger-jan-00/.

Hines N., 2001, The problem with testing, Journal of Defense Software Engineering.

IEEE 829 Test Plan, http://www.coleyconsulting.co.uk/IEEE829.htm.

ISO 8402, 1986, Quality-Vocabulary ISO, Geneva.

Jacobs J., van Moll J., Stokes T., Nov. 2000, The Process of Test Process Improvement, Xootic Magazine, Eindhoven.

Jensen R. W., 1996, Management Impact on Software Cost and Schedule, http:// http://www.stsc.hill.af.mil/crosstalk/1996/07/manageme.asp.

Järvinen P., 1999, On Research Methods, Tampereen Yliopistopaino Oy, ISBN 951-97113-6-8.

Kaner C., Falk J., Nguyen H. Q., 1993, Testing Computer Software, 2nd edition, Van Nostrand Reinhold, New York, U.S.A.

Marick B., 1999, New Models for Test Development.
http://www.testing.com/writings/new-models.pdf.

McConnell S., 1996, Software Quality at Top Speed, Software Development Magzine,
http://www.stevemcconnell.com/articles/art04.htm

Mizuno O., Shigematsu E., Takagi Y., Kikuno T., 2002, On Estimating Testing Effort Needed to Assure Field Quality in Software Development Graduate School of Engineering Science, Osaka University, Japan.

Moody D., 2002, Empirical Research Methods.
http://www.idi.ntnu.no/~ekaterip/dif8916/Empirical%20Research%20Methods%20Outline.pdf

Mosley D. J., 1993, The Handbook of MIS Application Software Testing: Methods, Techniques, and Tools for Assuring Quality Through Testing, Prentice Hall, Englewood Cliffs, New Jersey, U.S.A.

Murugesan S., 1994, Attitude Towards Testing: A Key Contributor to Software Quality, Software Testing, Reliability and Quality Assurance, IEEE.

RTI, The Economic Impacts of Inadequate Infrastructure for Software Testing. NIST, May 2002. Prepared by RTI for National Institute of Standards & Technology.

Schieferdecker I., Dai Z. R., Grabowski J., Rennoch A., The UML 2.0 Testing Profile and its Relation to TTCN-3, http://www.swe.informatik.uni-goettingen.de/publications/IS_ZD_JG_AR/TestCom2003_UTP_Final.pdf.

Sommerville I., 2000, Software Engineering 6th Edition, Chapter 14.

Staab C. T., Oct. 2002, Using SW-TMM to Improve the Testing Process, Journal of Defense Software Engineering.

Viega J., McManus J., 2000, The Importance of Software Testing, Cutter Consortium, http://www.cutter.com/research/2000/crb000111.html.

Weinberg G. M., 1992, Quality Software Management—Vol 1: Systems Thinking, Dorset House, New York, U.S.A.

Whittaker A. J., 2000, What is Software Testing and Why It is So Hard? Practice Tutorial, Florida Institute of Technology, IEEE.

Yin R.K., 2003, Case study research: Design and methods, Sage Publ., Beverly Hills Ca.

APPENDIX—THE QUESTIONNAIRE GUIDE

All of the questions in the list are open-ended. The expected answers from the respondents are “yes”, “no”, “I don't know”, “I don't quite understand the question” and “This information cannot be revealed”. Besides these, any reasonable ideas and opinions concerning the interview questions from the respondents are welcome.

A. General questions about the interviewee/respondent:

- What is your name, what is your position in BH?
- How many years have you been worked for BH and in the testing area?
- Before you began to work for BH, do you have similar testing working experience in other company? Was it mainly in technology side or management side?

B. General company background related information questions:

- Does BH have any kind of approval or authorization from the testing organization?
- In your opinion, what issues from the management side and the technology side are the main points that BH testing group cares the most?

C. Questions about the Test Improvement Model key areas:

1.Organization

- Do you often arrange the test related training programs for the group members, and do you attend the training programs?
- What is the most costly part in the testing group work? How do you usually control the cost on that part?

- Some researchers have recently identified 3 organizational models in software companies. They are called as team model, interdepartmental model and resource pool model, as you can see from the *appendix*. In your opinion, are you using any one of these models at work? If yes, what are its advantages and disadvantages to the testing work?

- Continue with the last question, for the testers, do you think is it better that every tester masters only specific technology for the specific testing area or each test masters several specific technologies for the several specific testing areas? What is the normal BH situation in this sense?

- If you have to set up the project team, how do you usually choose team members from the testing group?

- Do you find any problems that most easily happen in the testing work? If yes, have you have any idea/ways to avoid?

- Have you planned to continuously improve the working efficiency of the testing group? From which perspective, you think that it might be improved better?

- Do you have certain rules to evaluate and motivate your team members?

2.Planning and Tracking

- For the test planning, do you have certain way to do it? Who does usually make and maintain the test plan?

- During the testing work, is the testing progress measured against the plan? If there is nonconformance, how do you do? Who will make the decision?

- During or after testing work, do you usually evaluate your team member's working performance? What are the main issues in the evaluation?

3.Testware

- How do you manage the test data? (Here data means, test plan, test cases, error report, cost, schedule, etc.)

4.Test case

- How do you normally maintain the test cases? Are they following some rules, standards? Who will make the approval of the test cases?

- If test cases change, do you keep the traceability? How do you usually do?

- When running the test cases, is there any priority between them? What are the criteria to set the priority?

5.Reviews

- According to the Test Improvement Model reviews key area, what does review mean in BH?
- If you have review activities, are they done by review specialists, by testers or some other third party? What will be usually reviewed?

- How do you approve your test engineer before they could really start to work without the tutoring?

- Do you have test development plan for each tester? If so, usually, does the plan have the different levels? What are those levels? If no, what are the main points of the development plan?

- About the testing work environment, including the software environment and hardware environment, do you consider to improve and maintain them? How do you usually plan to do? (software environment means the HR, welfare, salary; hardware environment means the equipment, tools, etc.)