Matti Matikainen

INFORMATION SYSTEM SUPPORTED PROJECT ESTIMATION AND MEASUREMENT

Master of Science Thesis

11.5.2006

Department of Information Systems
Faculty of Information Technology
University of Jyväskylä
Jyväskylä, Finland

ABSTRACT

Matikainen, Matti Seppo Juhani

Information system supported project estimation and measurement
Jyväskylä: University of Jyväskylä, 2006. Master of Science Thesis, 97 p.

Software project estimation and measurement (PEM) has been researched extensively due to negative organizational and financial consequences of projects that run late and out of budget. PEM is a knowledge-intensive business process and thus critically dependent on effective information systems (IS) support. There is relatively little research on software Project Estimation and Measurement Tools (PEMT) and how to use PEMT effectively in enacting and redesigning PEM. Commercial PEMT exist but they vary significantly in functionality and effectiveness. As a result, the markets for such tools do not know what to expect from the tools and how to evaluate them.

The objective of this thesis is to create a partial Information Systems Design Theory (ISDT) for the class of PEMT. A complete ISDT prescribes both the product and process aspects of a class of IS, that is, what are the meta-requirements, meta-design, and applicable theories for all products within the class and how to build the products. This research focuses on prescribing the product aspects for the class of PEMT. It identifies the connection between the ISDT concept and Information Systems Research Framework (ISRF); is compatible with Dual Information Systems (DIS), an ISDT for building a class of flexible and effective information systems for supporting business process enactment and redesign; and uses those parts of DIS that are most relevant for the ISDT for PEMT. Partial meta-requirements are derived from PEM processes and a commercially available PEMT. Design product effectiveness hypotheses are defined to clarify the expected organizational benefits from using a PEMT instance derived from the class of PEMT. To our knowledge, equivalently holistic information systems design theory building for PEMT has not been conducted prior to this thesis. However, the validation of the results is out of the scope of this thesis.

KEYWORDS: Function Point; Software Measurement; Software Estimation; Project Estimation and Measurement Tool; Design Science; Information Systems Design Theory; Information Systems Research Framework; Dual Information Systems

TABLE OF CONTENTS

TABLE OF FIGURES

1 INTRODUCTION

A common perception of the software industry is that it delivers products late, over budget, and of poor quality and uncertain functionality. Measurement-informed management — an assumed principle of any true engineering discipline — can help to turn this perception around (SWEBOK, 2004, p. 8-2). Still at present, software development projects often run late and out of budget.

Software projects need to be estimated (see Appendix 1) beforehand in order to make project timeline and budget. Software products are abstract. Estimating software projects is a problematic work task. The size of the software can be measured by calculating different functions of the software via the software requirements (see Appendix 1). Function point (FP) calculation is a software measurement method, which is based on software requirements. Delivery rate (see Appendix 1) is typically the effort (see Appendix 1) or number of hours taken to deliver a FP. Factors that affect the development productivity or characterize the development circumstances or situation need to be analyzed in order to precise the estimates (hereafter referred to as "situation analysis"). The rate of reuse and the reusability requirements of the software affects the total effort needed in the software development (hereafter referred to as "reuse analysis"). Software Project Measurement and Estimation (PEM) consist of functional size measurement, delivery rate analysis, situation analysis, and reuse rate measurement. PEM is a subject that needs to be managed in order to stop projects running late and out of budget.

Software requirements may change during the project. These changes may modify the project scope. By this reason, project scope management is needed. Software development progress needs to be measured and managed in order to keep the project timeline and budget up to date. Software project estimation can be strengthened by comparing current software development project characteristics to former software development history data. If the project scope

or any other important factors (e.g., staff or quality requirements) change, re-estimation may be needed. Change management is a process where the development situation changes and the impacts of these changes are analyzed and managed.

Software project measurement needs to be standardized in order to get comparative data from former projects and projects made by other companies. In other words, use of standardized metrics and a standard measurement process enables learning from experience. When project history data is stored into a database, the project performance measures can be benchmarked. Project history data and metrics standardization are thus key elements in PEM.

Benchmarking can be defined as ”a continuous, systematic process for evaluating the products, services, and work processes of organizations that are recognized as representing best practices for the purpose of organizational improvement.” (Spendolini, 1992, p. 9) “Benchmarking is a company-internal process in which the activities of a given company or business unit are measured against the best practices of the best-in-class companies” (Käkölä & Koota, 1999, p. 102 quoting Band (1990), Miller (1992), Sharman (1992), Shepetuk (1991)). PEM benchmarking needs to be based on standardized project history database.

Experience databases can be used for two purposes: delivery rate determination and process improvement. In delivery rate determination, the characteristics (e.g., business sector) of the current development project are compared against the actual data of a group of development projects with comparable development environments. Benchmarking is used when comparing the development project with others found from experience database in order to learn from experiences and improve the software process. Benchmarking can be done by comparing one project or a group of projects against another group of projects with similar characteristics. This process of benchmarking project data

can be done inside one company or against a common experience database. In this thesis, experience database is also referred as Experience® database in the context of Finnish Software Measurement Association (FiSMA) organization.

One way to improve software project management (see Appendix 1) is to strengthen the effectiveness of PEM processes with the support of Information Systems (IS) tools. This thesis defines Software Project Estimation and Measurement Tools (PEMT) to consist of five primary steps: (1) project size calculation, (2) software reuse impact to the effort, (3) delivery rate determination (4) situation analysis, and (5) benchmarking data collection. PEMT is needed in software projects to keep projects on time and on budget throughout the development project life-cycle. Most of the PEMTs support at least two of these steps; the advanced ones support all these five steps. The more complete the estimation process is, the more accurate the estimates are. However, if there is no applicable experience data available or not enough knowledge about the requirements, the estimates are very likely to be inaccurate. PEMT support and are critically dependent of high quality benchmarking data. Owing to that, PEMT require and reinforce standardized data collection.

The estimation process multiplies the results of functional size measurement, situation analysis, reuse analysis, and delivery rate determination. According to OTA (2005), the project size can be estimated by a formula where the estimate E = S * D * T * R where E stands for FP, D for delivery rate, T for situation analysis and R stands for reuse rate. For example, the calculated estimate may 500 fp * 4,5 h/fp * 0,83 * 0,91.

According to tests presented in ISBSG (2005, p.75) "if ten project managers from different business areas try to estimate project effort without a systematic approach, including FSM methods, the ratio between the smallest and biggest estimate is 1 to 6, the worst as high as 1 to 12." According to Forselius (2006)

"the FiSMA situation analysis multiplier may vary between 0,5 and 2,5. Reuse multiplier may vary between 0,5 and 2." (Forselius, 2006) According to ISBSG (2005, p. 103) the median delivery rate in mainframe development platforms in C++ is 29,1 FP/h when by Natural it is 5,8 FP/h. The difference between these two cases is 5. When the FS, situation multiplier, reuse multiplier, and delivery rate are multiplied, the maximal possible value of the result is thus 100 times bigger than the minimal possible value. By these reasons, all these four stages are needed in PEM and they all are included into PEMT described in this thesis.

1.1 Research problem and the objective of this study

This thesis creates a partial information systems design theory for PEMT based on the PEMT related models and methodologies which are derived from the literature review. It is obvious that design is an important topic within the IS discipline. Design theory is based on theoretical underpinning that says how a design process can be carried out in a way that is both effective and feasible. Information Systems Design Theories (ISDT) are intended to produce more effective information systems. (Walls et al. 2004, p. 37-38) This thesis proposes theoretical integration of the ISDT concept of Walls et al and the models and methodologies of PEMT derived from the foundational books describing the knowledge area. Information Systems Research Framework (ISRF) is a concept closely related with the ISDT concept. ISRF is used by IS researchers to create and evaluate scientific artifacts that help practitioners make more efficient and effective information systems. In this thesis ISRF is used to broaden and deepen the readers understanding of the ISDT for PEMT and to strengthen the theoretical evaluation of the ISDT for PEMT. The utilization of ISRF enables better future research possibilities.

Design theory describes the product of design by a set of meta-requirements, which describe the class of goals in which the theory applies. On the other hand, meta-design describes a class of artifacts hypothesized to meet the meta-

requirements. The design product and the design process should be based on kernel theories. Kernel theories are based on natural or social science theories, which govern design requirements. Testable design effectiveness hypotheses can be used to verify whether the meta-design satisfies the meta-requirements. (Walls et al. 2004, p. 45-47)

The different aspects of PEMT are focused on in this research due to the relevance of PEMT for software engineering management and process improvement. ISDT for PEMT is a relevant subject to study because currently there is no scientifically based and validated design theory available in the IS literature that would help practitioners standardize their PEM practices and develop PEMT for supporting these practices. However, this thesis builds a design theory based on the most common PEM practices in Finland. When a PEMT is constructed following the ISDT for PEMT, the quality of the product increases.

Software projects running late and out of budget cause losses for both software supplier and customer organizations. These challenges can be facilitated by PEMT. In the recent years, there has been growing interest in software estimation and measurement. (Forselius, 2005)

The primary goal of this thesis is to theoretically build a partial ISDT for PEMT.

Building an ISDT demands an in depth analysis of the project scope management knowledge area. Theoretical support is presented from foundational books of PMBOK (2004), SWEBOK (2004) and ISBSG (2005) covering the project scope management knowledge area and its definitions. Literature review is based on these books, PEMT research articles, and educational material. Meta-requirements and a part of meta-design are derived from the literature review. They are only partial and may not be the only ones. An ISDT for PEMT demands also an in depth analysis of the methods used in PEM. This thesis creates PEMT from the bases of OTA (2005), Finnish Software

Measurement Association (FiSMA) (2004), (2002), (2000), Forselius (2005), (2004), (1999), Forselius and Maxwell (2000) and ISBSG (2005).

The two basic challenges related to the diffusion of effective PEM practices in Finland are that (1) most organizations have failed to institutionalize the estimation and measurement processes so far and (2) there has been lack of external pressure for software businesses to leverage PEM. There may be other reasons too; for example, software businesses may have needed to improve other (at that time) more essential software engineering processes first instead of management processes. Low level (note, for example, CMMI or SPICE level 1 maturity) organizations don't benefit from management as much as better ones. (Forselius, 2005)

Lack of recognition of the potential of PEM and PEMT might be due to the undeveloped design theory and the lack of scientific evaluation and validation of the benefits of PEMT. One stated challenge in PEMT is that customers, managers, and software engineers may lack conceptual and holistic understanding of the product development processes and the benefits to be gained from using PEMT. ISDT for PEMT has not been published earlier in scientific research. From these reasons the need for an ISDT for PEMT is evident.

The following research questions are derived from the research objective of this thesis:

- What kind of information system best supports project estimation and measurement?
  - What PEM processes should be supported by PEMT?
  - What are the partial meta-requirements for PEMT?
  - What are the design product effectiveness hypotheses of PEMT?
  - What possibilities exist in software project benchmarking?

This research is made for both scientists and practitioners. Practitioners can benefit from this thesis through the documentation of FiSMA processes,

methods, and the general requirements and design for PEMT. Scientists can build future research initiatives on the basis of the pioneering creation of the ISDT for PEMT. The ISDT ensures that the PEMT described here can be used in any PEM domain.

This thesis overviews and evaluates PEMT scientifically to improve both the scientific and practical understanding of the usefulness and functionality of PEMT. This is done by defining the basic artifacts of the PEM tools, specifically the meta-requirements, and design product effectiveness hypotheses, and further by investigating benchmarking possibilities.

1.2 Scope and limitations of the study

This thesis builds a partial design theory for PEMT. ISDT partial meta-requirements are identified from the literature review. Parts of meta-design are presented in a figure derived from OTA (2005). Design product effectiveness hypotheses are created for ISDT. The findings of this thesis are not validated.

FiSMA Functional Size Measurement method 1.1 (FiSMA FSM), FiSMA reuse measurement method (FiSMA RMM version 2002), Situation analysis model (Experience ND21) and OTA (2005) are the base sources for ISDT for PEMT. FiSMA provides a guideline for Scope Management Concept for a PEMT. Other measurement methods are not covered in this thesis. It could be argued, if the FiSMA methods are the best ones to describe a PEMT. For example, COCOMO II and VAF are two alternative methods for situation analysis. (ISBSG, 2005, p.14) However, they VAF's coefficiency is 0,65 – 1,30 when Experience ND21 provided by FiSMA coefficiency is 0,5 – 2,5. On the other hand, COCOMO II needs to be calibrated by the user. VAF evaluates 14 factors when FiSMA has chosen 21 factors. Furthermore, FiSMA FSM is the most comprehensive FSM method available. FiSMA methods are the most widespread PEM methods in Finland. FiSMA FSM has evolved to a stage of a theory and has a standard

position. It is conformant with the ISO/IEC 14143-1 FSM standard. From these reasons, alternative methods are out of the scope of this thesis.

"The key differentiator between routine design and design research is the clear identification of a contribution to the archival knowledge base of foundations and methodologies" (Hevner et al. 2004, p. 81). Due to this reason, this research does comprehensive exploration of definitions of project scope management, FiSMA methods, PEMT processes and benchmarking.

PMBOK (2004, p.103) has defined project scope management the following way: "Project Scope Management includes the processes required to ensure that the project includes all the work required, and only the work required, to complete the project successfully." PMBOK (2004, p.103) divides scope management into five sections which are scope planning, scope definition, create work breakdown structure (WBS), scope verification and scope control (see Appendix 1).

One of the original ideas of this thesis was to combine design science, behavioral science, ISDT, and especially dual information systems theory with a PEMT. However, in the context of thesis, all these subjects are not covered completely.

Duality of information systems means that organizations can use information technology as a course of working and learning. This requires that agents can use and modify technology when needed and modify computer supported work practices. Working and learning in the organization should be supported by technology. One key element of the technology is sufficient instantiation. Dual Information Systems (DIS) is a theory base on design theory. DIS does not fully fit to ISDT for PEMT. However, most of the model can be settled to PEMT. DIS model needs to be further developed and modified to fully fit to into ISDT for PEMT. Development of DIS is out of the scope of this thesis. The research is

done in order to gain compatibility with DIS theoretical framework by using some definitions of DIS.

The primary processes of PEMT are analyzed through Design Science artifacts and design product effectiveness hypotheses, meta-requirements and meta-design creation. However, the validation of the theories is out of the scope of this thesis.

The scope of this thesis is exceptionally broad. This thesis discusses the combination of ISDT and ISRF within compatibility with DIS. The theories are linked within the practical FiSMA Scope Management Concept.

1.3 Research methodology

This chapter discusses about the research paradigm and related research. Here the structure of ISDT and ISRF research are introduced. PEM has been under research earlier, but an ISDT for PEMT has never been done before.

1.3.1 Research paradigm

Research paradigm, as a term, can be defined as the combination of research questions asked, the research methodologies allowed to answer them and the nature of the pursued research products. Most academic research in management is based on an idea that the mission of all science is to understand, describe, explain and possibly predict. (van Aken, 2004, p.220) This research problem is extremely important because most projects would benefit from accurate estimates. If systematic estimation and measurement is not used, the project scope may change a lot as stated earlier.

This research uses the design science paradigm as research paradigm. Components of ISDT presented in Walls et al. (1992, p. 44) describe the components of ISDT research. The research method follows the components of ISDT, which is presented in (FIGURE 3, p.27). First kernel theories are defined,

then meta-requirements and meta-design are created, next design product effectiveness hypotheses are created. ISRF intends to make more efficient and effective information systems by creation and evaluation of artifacts. ISRF is used in order to guide design theory creation and to support future research. The design science paradigm supports the ISDT creation. The behavioral science research paradigm can be used to evaluate organizational implications of the artifacts created through the design science research and implemented in organizations. That paradigm is not used in this thesis but it will be very useful in future research to empirically study and validate the ISDT developed in this thesis. Behavioral science can be used for model creation and predicting purposes. Design product effectiveness hypotheses define and predict what consequences the utilization of PEMT should have in an organization. If behavioral science paradigm validation disproves the hypotheses, then the ISDT should be re-designed. However, some other process aspects may affect the behavioral science results such as top-management commitment to PEM processes or PEMTs. On the other hand, these are not parts of an ISDT and they should be described separately by behavioral science theories or the scope of ISDT should be broadened to meet the organizational practices laws.

Design science creates and evaluates IS artifacts which intend to solve identified organizational problems. "The resultant IT artifacts extend the boundaries of human problem solving and organizational capabilities by providing intellectual as well as computational tools" (Hevner et al. 2004, p.76). Design science consists of two activities, build and evaluate. Building is the process of constructing an artifact for a specific purpose when evaluation is the process of determining how well artifact performs. Purposeful artifacts are built to address unsolved problems. The artifacts are specified as constructs, models, methods, and instantiations. (Hevner et al. 2004, p. 75-80)

"Information systems are implemented within organizations for the purpose of improving the effectiveness and efficiency of organizations. Capabilities of the

Information system and characteristics of the organization, its work systems, people, development and implementation methodologies together determine the extent to which that purpose is achieved" (Hevner et al. 2004 p. 76 quoting Silver et al. 1995). ISRF characterizes two paradigms that are used for characterizing the research in the information systems discipline: behavioral science and design science. They are foundational to the IS discipline in their confluence of people, organizations, and technology. The goal of behavioral science research is truth while the goal of design science research is utility. Truth and utility are inseparable when truth informs design and utility informs theory. An artifact may have utility because of some as yet undiscovered truth. A theory may be developed to the point where its truth can be incorporated into design. Design science is described by building and evaluation phases of artifacts. They are designed to meet the identified business need. (Hevner et al. 2004, p. 75-81)

Behavioral science is based on natural science research methods that seek to develop and justify theories that explain or predict organizational and human phenomena surrounding the analysis, design, implementation, management, and use of IS. These "theories ultimately inform researchers and practitioners of the interactions among people, technology, and organizations that must be managed if an information system is to achieve its stated purpose, namely improving the effectiveness and efficiency of an organization." (Hevner et al. 2004, p.76)

Design science is a problem-solving paradigm the purpose of which is to extend the boundaries of human and organizational capabilities by creating new and innovative artifacts. "The design-science paradigm has its roots in engineering and the sciences of the artificial" (Hevner et al. 2004, p. 76 quoting Simon, 1996). "It seeks to create innovations that define the ideas, practices, technical capabilities, and products through which the analysis, design, implementation, management, and use of information systems can be effectively and efficiently

accomplished" (Hevner et al. 2004 quoting Denning 1997; Tsichritzis 1998). When artifacts are built, they can be evaluated.

The build/evaluate loop creates artifacts in design science. "Their creation relies on existing kernel theories that are applied, tested, modified, and extended by the researcher" (Hevner et al. 2004, p. 76 quoting Markus et al. 2002; Walls et al. 1992).

FIGURE 1 describes behavioral science and design science in IS research, and the relation with develop/justify and evaluate/build loop.

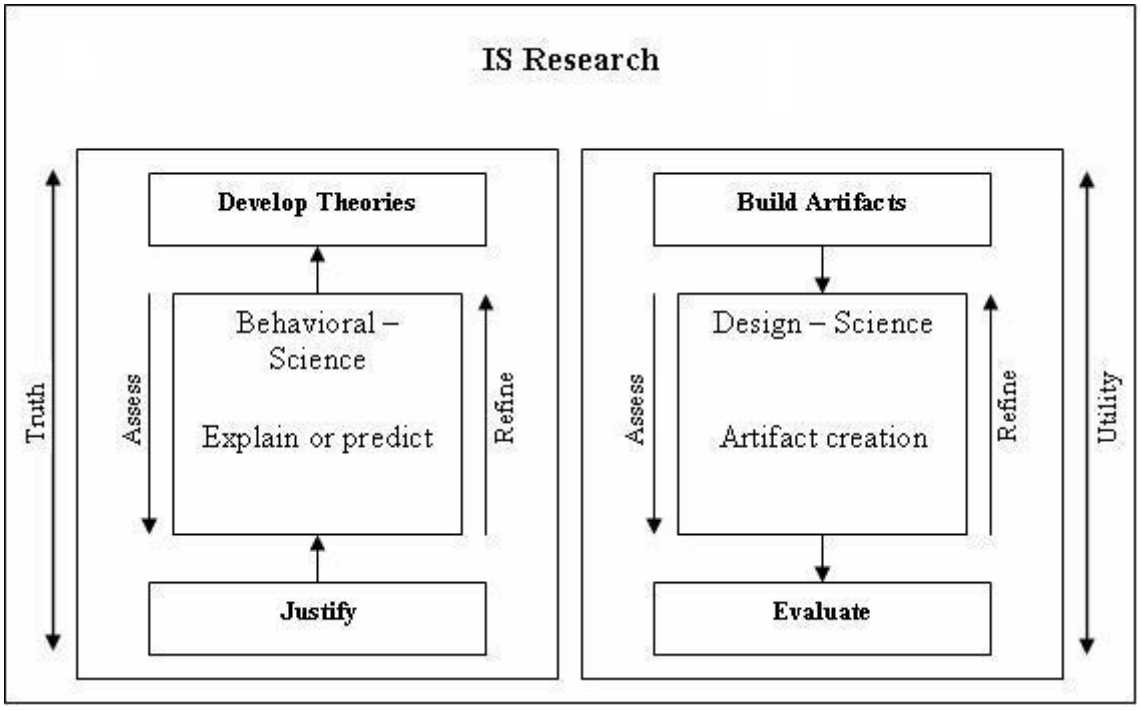

FIGURE 1 Behavioral Science and Design Science in IS Research

The behavioral-science paradigm intends to justify/develop theories. It seeks to find the truth by explaining or predicting. The design-science paradigm builds/evaluates artifacts loop seeks to find utility for artifacts. The theory or artifact assessment/refinement creates a loop while design-science and behavioral-science paradigm can be repeated several times. The utility of the paradigm accrues from the interaction of the loops in design science.

1.3.2 Related research

ISDT is a theory that tells what the necessary and sufficient properties of PEMT are and how it should be designed to best achieve the goals of PEM. Several Design Theories have been published since 1992 when Walls et al. published their article "Building an Information System Design Theory for Vigilant EIS". Walls et al. (2004) have reviewed all the ISDTs they could find based on their extensive review of IS literature. Four research articles could be found using ISDT concept extensively. Stein & Zwass (1995) developed a design theory for an organizational memory information system. Kasper (1996) created an ISDT for decision support system. Markus et al. (2002) present a design theory for systems that support emergent knowledge processes. Hall et al. (2003) developed an ISDT for learning-oriented knowledge management systems. In addition, Liimatainen (2004) wrote a thesis in the University of Jyväskylä about ISDT for integrated requirements management and release management systems. None of the ISDTs published previously describe an ISDT for PEMT. This is the first research that deals with this research area.

PEM processes, methods and benchmarking have been under research earlier. For example, Albrecht (1979) has written about the measurement of application size. Basili (1992) gets into the basics of software engineering, development, methods and tools. Boehm (1981) investigates the major estimation techniques and software cost estimation including COCOMO. Kemerer (1987), Kitchenham (1992), Putnam and Myers (1992) and Jones (1998) have also investigated the PEM research area. (Wieczorek, 2001, p. 39-45, 155-166) Stutzke (2005) published a book "Estimating Software-Intensive Systems Projects, Products, and Processes" describing practical, proven estimating techniques. FiSMA methods and Scope Management Concept have created using PEM research information.

FiSMA network[1] has evolved a scope management concept for PEM, consisting of several separate measurement methods, such as FiSMA FSM 1.1, FiSMA Reuse Measurement Method (RMM) version 2002 and two Situation Analysis Methods, one for new IS development and another for software maintenance (Experience ND21 and MT22). The whole concept has not been published before, except as a large set of training material (OTA 2005). ISBSG has published several books in last ten years. This thesis uses only Practical Project Estimation 2nd edition published in 2005. Overview of PEM has been introduced probably best so far in their book Practical Project Estimation, 2nd edition (ISBSG 2005). Hundreds of articles and conference papers have been published about PEM. The most interesting in the point of view of this thesis are for example Productivity of Software Projects by Business Sector: An Empirical Analysis of Trends (Premraj et al. 2004) and Benchmarking Software-Development Productivity - Applied Research Results (Maxwell K., Forselius P., 2000). These articles, methods and books are closely related to PEM and PEMT, but none of them is covering the whole area, especially the interaction between human actors and information systems. That's why there is a need for a scientific description of PEM and PEMT.

According to IS development specific estimation and measurement training materials (OTA 2005), the five scope management centric PEM processes are 1.) initiating the software development project and target software to be estimated and measured, 2.) estimating development cost and duration, 3.) controlling progress of the development, 4.) managing proposed scope and quality changes and 5.) closing the development project. Domain independent project management standard PMBOK (2004, p.103) divides project management into

[1] FiSMA network consists of (1) member companies, organizations and universities, (2) working groups that develop the FiSMA methods and (3) FiSMA administration. The network cooperates with both international and Finnish researchers.

nine knowledge areas[2]. Project scope management is one of them. It is divided into five processes: scope planning, scope definition, create work breakdown structure (WBS), scope verification and scope control. The PEM processes are developed following the basic ideas of PMBOK (2004) by FiSMA (OTA 2005). Both the available PEM training materials (OTA 2005) and a selected set of foundational books (PMBOK 2004), (SWEBOK 2004), (ISBSG 2005) are used for defining the PEM knowledge area in this thesis. They are discussed more detailed in SECTION 3. From these bases, the literature review is sufficient and comprehensive for theory building.

1.4 Structure of the thesis

After the introduction section, the second chapter introduces the Information Systems Research Framework (Hevner et al. 2004) and Information Systems Design Theory (Walls et al 1992, 2004) and characterizes their interrelations. The main concepts of DIS are also defined. The third section gets into FiSMA Concept of Organizational Learning and PEMT processes. Literature from FiSMA processes is presented, and process benchmarking using Experience® database is discussed. FiSMA Concept of Organizational Learning is discussed with conclusion of the FiSMA methods. Meta-requirements are derived from the bases of the literature. The chapter four gets into an experience database. The chapter five gets into meta-requirements and effectiveness hypotheses of the partial ISDT for PEMT. The results are discussed and concluded. Also topics for further research are proposed.

Definitions of requirements, project scope management functional size measurement, and other PEMT related concepts are also defined in appendixes.

---

[2] PMBOK (2004) divides project management into (1) project integration management (2) project scope management, (3) project time management, (4) project cost management, (5) project quality management, (6) project human resource management, (7) project communications management, (8) project risk management and (9) project procurement management

2 DESIGN SCIENCE AND INFORMATION SYSTEMS RESEARCH FRAMEWORK

This chapter discusses (1) design science and IS design theory in detail, (2) the basics of the ISDT for DIS, and (3) the link between ISDT and ISRF.

2.1 Information Systems Research Framework

Hevner et al. (2004, p.80) presents the conceptual framework for understanding, executing, and evaluating IS research in FIGURE 2. It combines behavioral science and design science paradigms within the context of environment, knowledge base and IS research.

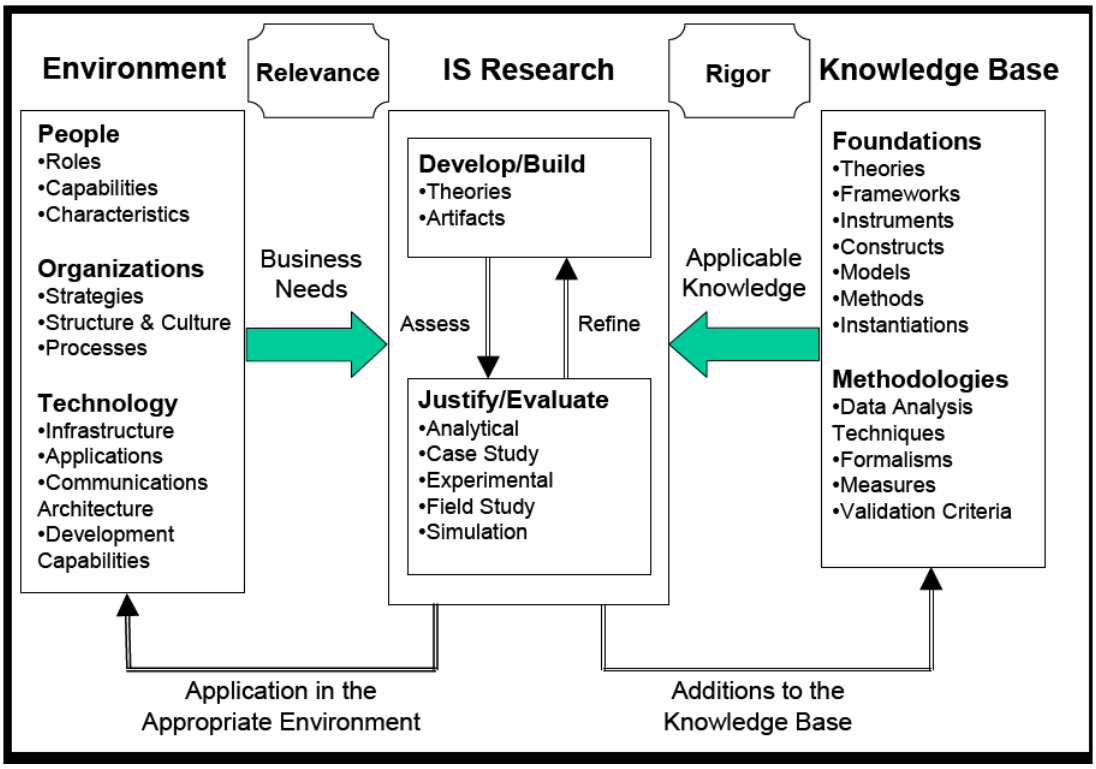

FIGURE 2 Information Systems Research Framework (Hevner et al. 2004, p.80)

According to Hevner et al. (2004, p. 79) "the environment defines the problem space (Simon 1996), in which reside the phenomena of interest. For IS research, it is composed of people, (business) organizations, and their existing or planned technologies (Silver et al. 1995). In it are the goals, tasks, problems, and

opportunities that define business needs as they are perceived by people within the organization." The business needs are assessed and evaluated within the context of organizational strategies, structure, culture, and existing business processes. As shown in FIGURE 2, IS research is conducted in two complementary phases, behavioral science and design science. Behavioral science is described by development and justification phases. Theories that are related to business needs are researched. Knowledge base consists of foundations and methodologies. IS research and results from reference disciplines provides foundations to the develop/build phase. Design science research needs to be differentiated from routine design or system building. Design science research addresses important unsolved problems in unique or innovative ways or solved problems in more effective or efficient ways. A contribution to the knowledge base of foundations and methodologies needs clear identification. (Hevner et al. 2004, p.79-81)

According to Hevner et al. (2004, p. 78) the artifact evaluation provides improved understanding of the problem. Building and evaluating is typically done a number of times before the final artifact is generated. The final artifact answers to the defined business need. "The contribution of behavioral science and design science in IS research are assessed as they are applied to the business need in an appropriate environment and as they add to the content of the knowledge base for further research and practice." (Hevner et al. 2004, p.81)

"Information systems research lies at the intersection of people, organizations, and technology." (Hevner et al. 2004 p. 98 quoting Silver et al. 1995) They represent inputs or business needs to the information systems research. Meta-requirements can be generated from these business needs or problems. After meta-requirements are presented, solution can be derived from the requirements. The build and evaluate loop in (FIGURE 1, p. 17) and (FIGURE 2, p.21) builds the artifact, which answers to the inputs presented by business needs. Artifact development is the development of meta-design. Meta-design is

the required solution for the meta-requirements. This can be seen as the build/evaluate loop that develops the artifact that solves the business needs.

Artifacts, which are generated by design science paradigm, can be evaluated by behavioral science paradigm. Design science and behavioral science uses partly the same methods. Design science can use empirical research methods, but not as comprehensively and profoundly as, for example, contextual design, presented by Markus et al. (2002, p. 185). These ethnographic and other resource-intensive methods do not intend to create general theories or meta-theories for classes of systems. Instead, they develop resources and rules for building one particular system for one particular organization, yielding particular intended consequences in that organization.

ISRF Knowledge base provides the material that is needed to fulfill business needs in IS research. Knowledge base provides raw materials used in research. Foundations are used in the develop/build phase while methodologies provide guidelines used in justify/evaluate phase. (Hevner et al. 2004, p. 80) The knowledge base and the environment are used in the generation of meta-requirements and meta-design. (Liimatainen, 2004, p. 23) The business needs defines the meta-requirements, then meta-design can be drown out of meta-requirements.

"Design science, as the other side of the IS research cycle, creates and evaluates IS artifacts intended to solve identified organizational problems" (Hevner et al. 2004, p.77). This thesis uses the design science of ISRF in order to represent the research paradigm. Behavioral science and evaluation of the artifacts is out of the scope of this thesis.

2.2 Information Systems Design Theory

The design of a system is the main concern in IS research. The design of a system is the design of components and their relationships (Walls et al. 1992

p.42 quoting Churchman, 1971). Design theories address the question of how to combine components and relationship in order to make subsystems and how to combine subsystems and relationship to make systems. (Walls et al. 1992, p.42)

The purpose of design theory is to guide artifact creations. According to Walls et al. (2004, p.48) "An artifact is 'a (hu)man-made object', 'an object produced or shaped by human craft', or 'any object or process resulting from human activity'. The word derives from the Latin words ars (skill) and facio (to make)." The laws of the natural and social world govern the components that comprise an information system.

Design theory improves both the quality of the product and the design of the process. Design is both a noun and a verb. This leads to a circumstance where a design is a product, "a plan of something to be done or produced" (Walls et al. 2004, p. 45). To design is a process "to so plan and proportion the parts of a machine or structure that all requirements will be satisfied" (Walls et al. 2004, p. 45). Design theory must thus have two aspects – one deals with the product of design, the other deals with the process of design. The final artifact answers to the defined business need. Design theory should have testable hypotheses. (Walls et al. 1992; Walls et al. 2004)

Design theory makes the development process more tractable for developers. It restricts the range of effective features and the range of effective development practices to a more manageable set. Restricting developers options improves the development outcomes. (Markus et al. 2002)

ISDTs inform researchers by suggesting testable research hypotheses. According to Markus et al. (2002, p. 181) ISDTs can be split into three interrelated elements: "(1) a set of user requirements derived from kernel theory, (2) principles governing the development process, and (3) principles governing the design of a system (i.e., specifying and implementing its features)." On the other hand, the phase no (1) contradicts with the thinking of

Hevner et al. (2004) who state that business needs are typically driven by design science. Markus seems to be flawed in his aspect.

This thesis creates an IDST for PEMT. ISDT is a theory which tells what the necessary and sufficient properties of PEMT are and how it should be designed to best achieve the goals of PEMT.

ISDT is a prescriptive theory that integrates normative and descriptive theories. The purpose of a design theory is to support the achievement of goals. However, the purpose of the theory is to explain why specific goals exist or predict outcomes associated with goals, not to achieve those goals. (Walls et al. 1992, p. 40)

The following points characterize design theories. (Wells et al. 1992, p. 36-41)

1. They must deal with goals as contingencies. For example, the ISDT for PEMT states that if the business goals (improved project scope management, project estimation, control and change management, measurement and data collection) are to be achieved, then the PEMT artifacts should be designed and used to support PEM processes and improve the quality of PEM.
2. They prescribe both the properties an artifact should have if it is to achieve certain goals and the method(s) of artifact construction.
3. They can never involve pure prediction or explanation. For example, the ISDT explains what properties a PEMT artifact should have and how it should be built and predicts that a PEMT will support the attainment of the business goals to the extent that it possesses the properties and is built using the methods prescribed by the theory.
4. They are prescriptive, composite theories integrating explanatory, predictive, and normative kernel theories from natural and social science and mathematics into the design path that realizes more effective design and use. They involve both the application of scientific theory to design

artifacts and the use of the scientific method to test design theories (usually by building and testing the artifacts empirically).

5. Design theories tell "how to (achieve the goal)/because" whereas explanatory theories tell "what is", predictive theories tell "what will be", and normative theories tell "what should be (the goal)".

Walls et al. (2004, p. 46) defines two aspect of IS research: product aspect and process aspect. The product aspect can be defined by four components and the process aspect with three components.

Product components:

1. Meta-requirements describe the class of goals to which the theory applies.
2. Meta-design describes a class of artifacts hypothesized to meet the meta- requirements.
3. Kernel theories are theories from natural or social sciences governing design requirements.
4. Testable design product hypotheses are used to test whether the meta-design satisfies the meta-requirements.

Process components:

1. Design method describes procedures for artifact construction.
2. Kernel theories of the design process aspect are theories from "natural or social sciences governing design process itself." (Walls et al. 2004, p. 46)
3. Testable design process hypotheses are used "to verify whether the design method results in an artifact which is consistent with the meta-design." (Walls et al. 2004, p. 46)

"Kernel theory enables formulation of empirically testable predictions relating the design theory to outcomes like system-requirements fit." (Markus et al.

2002) This refers to the testable design product and process hypotheses as defined by Walls et al. (1992, 2004) The components of ISDT research are described in FIGURE 3.

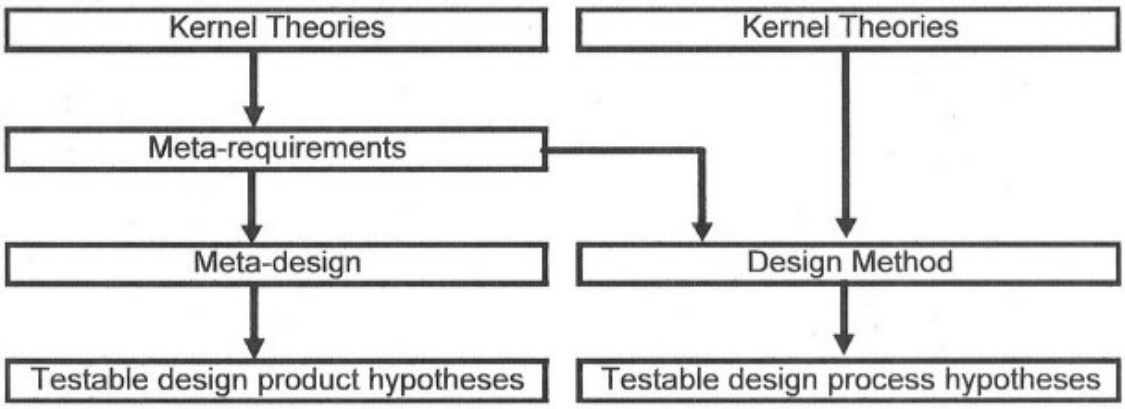

FIGURE 3 Components of Information Systems Design Theory (Walls et al. 2004, p. 46)

The product aspect of ISDT is covered in sections 3 and 4 by determining the characteristics of PEMT. The design product hypotheses are further developed is SECTION 5. However, the product aspect of the ISDT for PEMT is not validated and the process aspect is out of the scope of this thesis. The process aspect is a subject of future research.

2.3 Linking the Information Systems Design Theory with the Information Systems Research Framework

ISRF represented by Hevner et al. (2004) is partly divergent and partly the same as ISDT represented by Walls et al. (1992). Both of them are based on design theory, which purpose is to build more effective and efficient information systems. From this reason, artifact creation by Hevner et al. (2004) may promote the identification of PEMT.

ISDT builds an artifact by defining kernel theories, meta-requirements, meta-design and design product hypothesis. ISRF is divided into behavioral science and design science. Design science creates IS artifacts that "are broadly defined

as constructs (vocabulary and symbols), models (abstractions and representations), methods (algorithms and practices), and instantiations (implemented and prototype systems)" (Hevner et al. 2004, p. 77).

By default, ISDTs must give additions to the ISRF's knowledge base. ISDT artifacts answers the business problems defined by the environment and also make additions to the knowledge base.

ISDT produces one artifact. According to Walls et al. (1992, 2004) a development cycle can not be found from the development of ISDT. ISRF behavioral science and design science emphasize a cyclic research where the research method can be re-evaluated and repeated. ISRF creates a loop where build/evaluate artifacts and justify/develop theories are repeated. This is a repetitive process. The behavioral science can test the design science artifacts. ISDT can be fulfilled by the build/evaluate loop of ISRF. On the other hand, design science is not as complete as ISDT. On other words, ISDT is more complete and more precise compared to design science. However, ISDT is not a cyclic model when ISRF is. As a consequence, both ISDT and ISRF are needed when IS design is considered. This thesis does not validate the theories. However, it is a topic of future research.

2.4 Dual Information Systems

DIS denotes the IS architecture providing services defined in the ISDT for DIS (Käkölä 1996). The services defined in the meta-design of the ISDT for DIS (1) conceptually unite manual and computerized aspects of work, helping actors to understand their work holistically; (2) let actors zoom in on the details of their work practices and check shared databases for mistakes in order to deepen their knowledge and handle exceptions locally; (3) help actors draw on their improved knowledge to enter and interact in redesign project teams where they can share best practices and crystallize them into work process redesigns; and

(4) store the created design knowledge in the organizational knowledge base for later reuse.

The ISDT for DIS is used in this thesis in order to gain compatibility with ISDT for PEMT. Concepts used in the ISDT for DIS (derived from the underlying kernel theories of the ISDT) are also used to create the design product hypotheses for the ISDT for PEMT. Due to that reason, some concepts of DIS need to be defined.

Käkölä & Koota uses term "agent" instead of "user" to refer to people whose work is computer supported. Orlikowski (2002) sees information technology as constructed by human agency and as institutionalized in structure. This she calls "the duality of technology." Here duality implies that organizations can effectively utilize information technology if agents, working in organizations, can use and modify the technology whenever it is necessary to redesign computer-supported work practices, and if the technology can be institutionalized as a legitimate component of the organizational working and learning environment. Today, organizations and information systems they use are excessively institutionalized, that is, agents can not see the dual, both institutionalized and malleable nature of the information systems they use and thus alter the systems when necessary. Käkölä & Koota (1999, p. 88) draw upon the work of Ciborra & Lanzara (1994) to state that excessively institutionalized IS

> "(1) limit lateral communication, coordination and knowledge sharing; (2) provide little feedback to agents (especially in the lower echelons of an organization) on work arrangements and on the coordination and communication patterns which emerge from their use; (3) limit the agents' to reflect and inquire within the social and technical context in which the agents are embedded, restraining them from creating, questioning, and modifying practical knowledge when problems

> emerge; and consequently (4) endanger the process of reinvention that any complex technological artifact should undergo when put to use."

DIS can be divided into the following layers:

> "(1) agents on the business layer continuously draw on *the business layer of DIS* to learn, enact, and coordinate activities in business units; (2) self-organizing project teams on the project layer use *the project layer* and *the knowledge sharing server of DIS* to produce innovative work and IS (re)design that can be enacted on the business layer; (3) *the knowledge sharing server of DIS* is a repository of explicit work and IS design knowledge in the knowledge-base layer of a hypertext organization. In addition, there is *the breakdown layer of DIS* that agents on the business layer can use to zoom in on the details of their work and to deepen their understanding of the computerized aspects of work in order to handle unexpected (coordination) breakdowns" (Käkölä & Koota, 1999, p.97).

The knowledge sharing server (KSS) of DIS is outside the boundaries of the business units. It stores both current and historical work process knowledge of the organization. These main parts of DIS are presented in FIGURE 4.

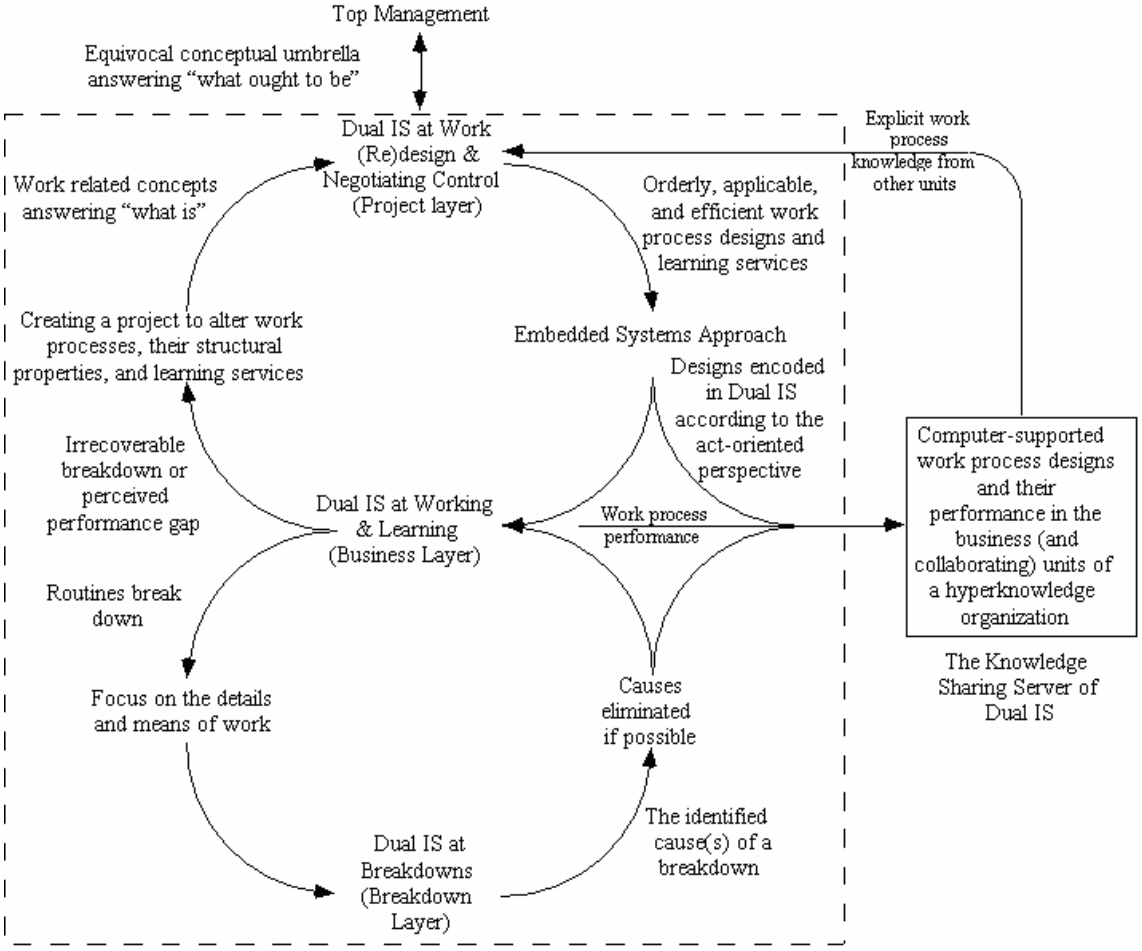

FIGURE 4 The conceptual design of a DIS in a business unit of a hyperknowledge organization (Käkölä & Koota, p. 98)

One important reason for agents' lack of awareness is that the conceptual and material structures of computer software typically reflect the design/use-dualism of technology. When using the institutionalized IS, the constructed nature of the IS is masked by the software. If agents are restricted from using the functions of the software, they face difficulties monitoring their actions. This decreases their ability to control the actions they make. (Käkölä & Koota, 1999, p. 91)

Due to the prevailing structural properties of organizations and the psychological motivations of people, the role of IT in most organizations is to help meet short-term productivity objectives. (Käkölä, 1996, p. 37) The project

risks materialize when activities cannot be completed due to ineffective processes or lack of resources (Käkölä & Taalas, 2005, p. 17). They can be reduced by improving the motivation and work-related knowledge of personnel (Sherer 1997) and by enhancing the level of coordination of sub-processes (Kraul & Streeter 1995).

Because of their limited ability to control all aspects of work, including computerized tasks, the actors cannot necessarily be responsible for their work as a whole (Käkölä & Taals, 2005, p. 17). DIS deepens the shared understanding among actors of the organizational context of tasks by providing actors information about work of other participants. The better understandability of work tasks reduces the number and severity of breakdowns. (Käkölä 1996, & Käkölä & Koota, 2005)

Drawn from the bases of DIS, PEMT needs institutionalization. PEM needs institutionalization not only when the estimated project is created but also in change management situations. The institutionalization of PEMT structures demands incorporation of the organization. Hereby the institutionalization of PEMT requires acceptance of organizational top management. However, utilization of DIS institutionalization is out of the scope of this thesis.

2.5 Conclusion

This section discussed the design science paradigm adopted in this thesis in detail. Both ISDT as defined by Walls et al. (1992) and (2004) and ISRF as defined by Hevner et al. (2004) have the same goals, that is, to help developers build more effective and efficient information systems and to develop theories endogenous to the IS field, thus helping the field as a whole mature both from practical and scientific viewpoints. This chapter got into ISRF and design science in detail. The concepts of the ISDT for DIS are discussed in a broad level to support the creation of the ISDT for PEMT.

3 SOFTWARE PROJECT ESTIMATION AND MEASUREMENT FOR ORGANIZATONAL LEARNING AND PROCESS IMPROVEMENT

This chapter characterizes why PEM is needed and how PEMT can support PEM processes. The chapter three discusses the FiSMA Concept of Organizational Learning, FiSMA Actors, PEMT Effort Estimation, PEM Processes and FiSMA Measurement Methods, FiSMA Calculation Processes. From the bases of FiSMA, PEMT meta-requirements are identified.

3.1 FiSMA Consept of Organizational Learning

FiSMA Concept of Organizational Learning identifies the main parties involved in PEM. Top management, project office and project management are parties from the development organization. FiSMA network supports the utilization of PEMT. FiSMA Concept of Organizational Learning is presented in FIGURE 5. It defines the roles of top management, measurement network, project office and the projects performed.

Figure FIGURE 5 introduces the main parties of FiSMA and their functions while FIGURE 6 describes their relations and interaction. In other words, it describes what the necessary and sufficient organizational structures of FiSMA are.

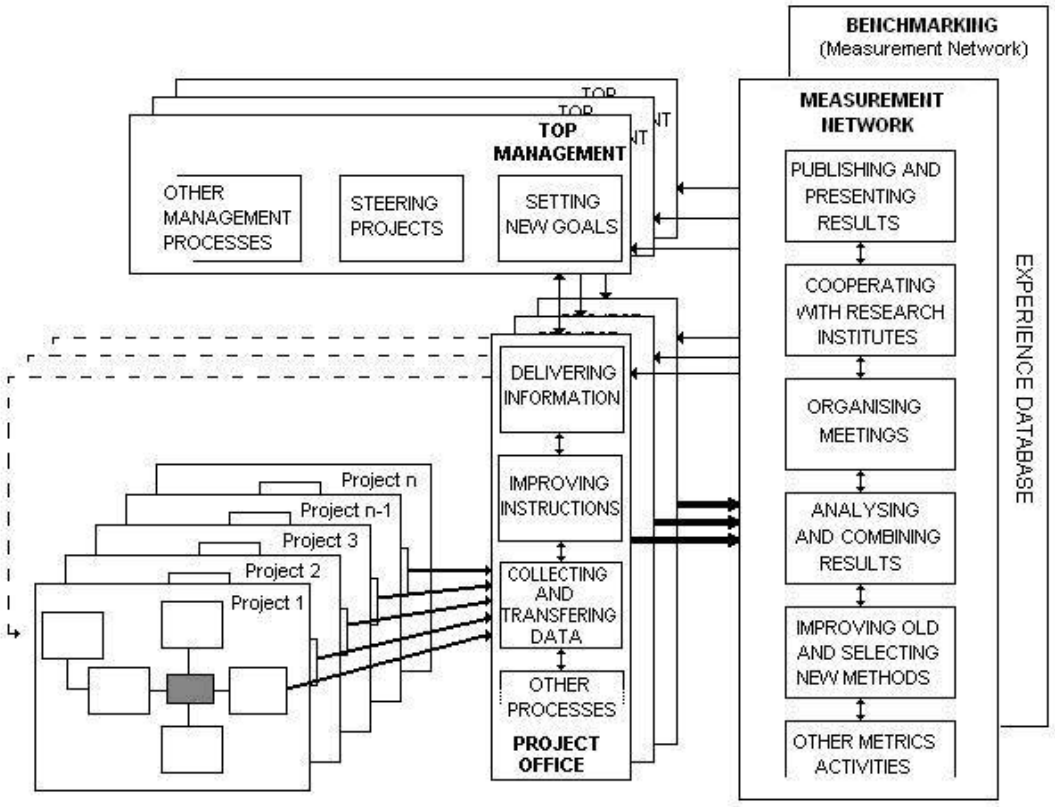

FIGURE 5 FiSMA Concept of Organizational Learning

The project managers manage the projects. It is usual, that there are several parallel projects at same time. Project office controls the information and data collection, transferring and sharing. Project office is also responsible of the instructions, information delivery and company data collection. The measurement network manages the Experience® database and further develops the methods. The methods development and improvement is done by the measurement network. It also provides benchmarking services.

Top management can follow the results of the past projects after the project office has collected and submitted project data. New goals can be set from the bases of past project performance results and reports. From these bases, the projects steering can be made. The top management support and attention is essential in organizational learning. The software process improvement and

organizational improvement can be assisted by the use of experience data. These processes of organizational measuring and new goal setting can be performed by benchmarking the results against an experience databases.

Organizations should motivate project managers to collect project data. If the executive managers don't see the value of collecting data, project managers and project office will not collect it. (OTA 2005, Forselius, 2005)

FIGURE 6 illustrates the organizational learning in a simplified way. It introduces the PEM actors and their interrelations. The main parties are the customer company and STTF Ltd (Software Technology Transfer Finland), which represents the experience factory. The actors are the project manager, project office, the PEMT and the repository manager of STTF. Other actors are the project team members, the project office of the organization and the measurement network. (Forselius, 2006)

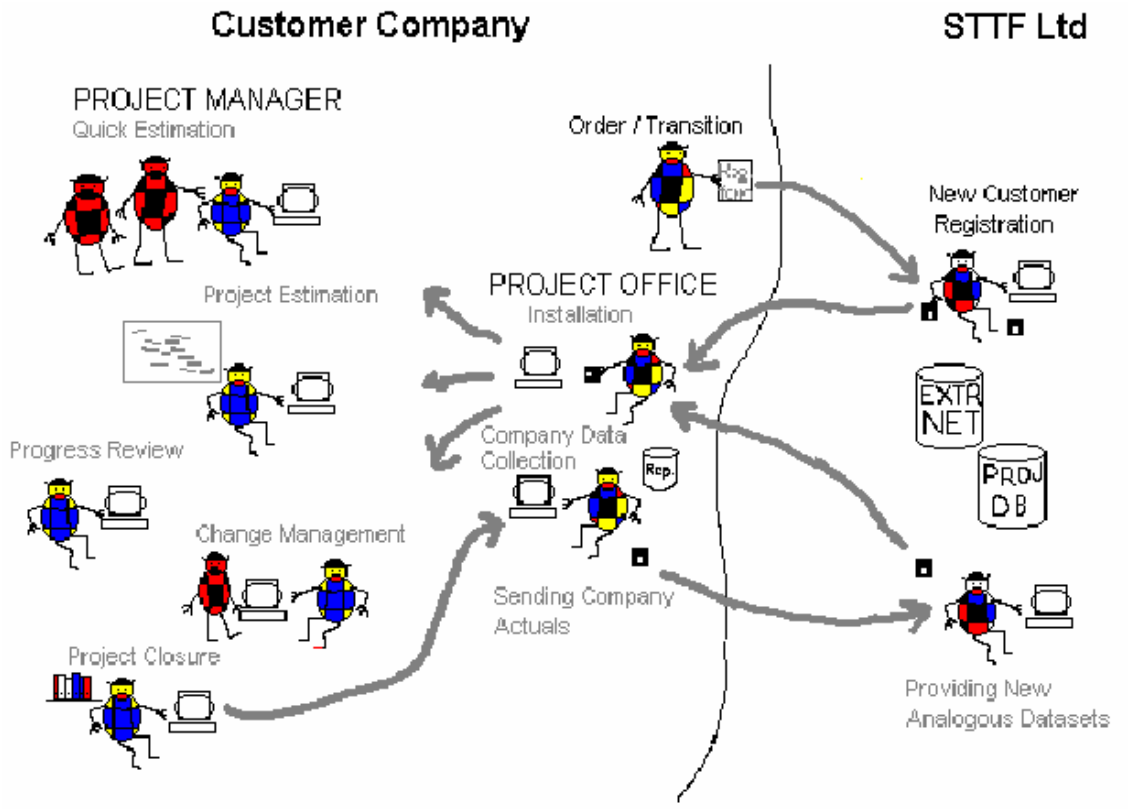

FIGURE 6 PEM actors and PEMT use cases (OTA 2005)

STTF Ltd stores and manages the Experience® database (PROJ DB). It collects data from all customer companies and creates new analogies for them and extends the common Experience® database. The Rep. in Project office represents the customer company specific Experience® database. When the data is collected, the quality of the data is evaluated and quality feedback returned together with new analogies. This process ensures better data quality and better benchmarking results.

The standards of collecting data and the quality of data may vary between different PEMTs. When benchmarking, the experience database must be compatible with the PEMT used. The bigger the experience database is the better and more exact the estimates are. However, if the quality of the experience data is poor, the reliability of benchmarking results is questionable. (Forselius, 2005)

From the bases of FIGURE 5 and FIGURE 6, PEMT should have centralized experience database, which is updated and managed. It should support data collection, quality management (see Appendix 1) and database distribution. PEMT should define and support software measurement network, project manager, acquisition management and top management. They are organizational structures that all organizations using PEMT should have.

Meta- Requirement: PEMT artifacts shall support network creation to gather actual project information and to distribute new versions of experience databases

3.2 PEMT effort estimation

This chapter discusses PEM processes, the FiSMA Scope Management Concept and presents meta-requirements for the ISDT for PEMT. The common effort estimating process (FIGURE 7a) is based on four simple axioms:

1. The smaller the target is, the smaller the effort needed in average.
2. The better the development circumstances are, the smaller effort needed in average.
3. The more reusable components available, the smaller development effort needed in average.
4. For each type of development the typical productivity level can be derived from previous experiences by measuring the actual effort and the size of result.

This process works as kernel theory for the FiSMA estimating process. When the common theory is applied at software development domain, the estimating process input information needs can be specified (FIGURE 7b).

Another truism behind the PEM concept is that if you want to improve something, you have to measure it. For PEM it means that we have to use same metrics and measurement methods in all scope management processes, throughout the development life-cycle.

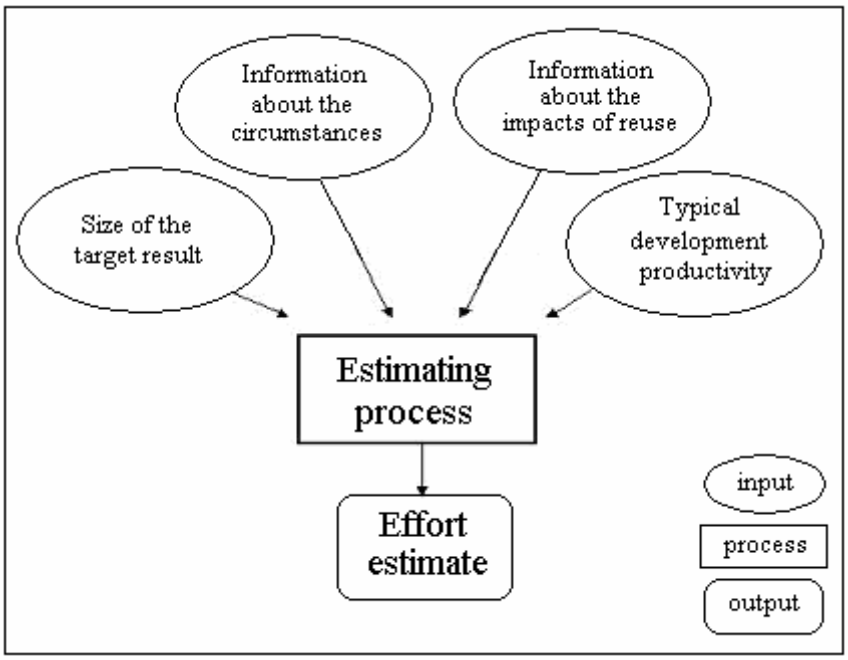

FIGURE 7a Common effort estimating process (Forselius, 2005)

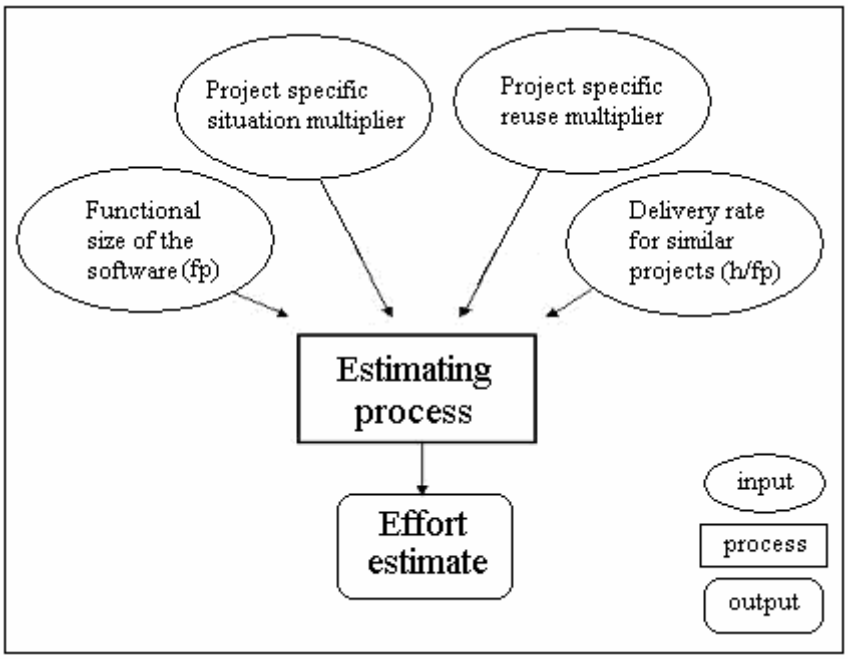

FIGURE 7b Effort estimating process for software development.

The input needed for estimating process of software development consists of Functional Size (FS) (see Appendix 1), Situation Multiplier, Reuse Multiplier and Delivery Rate. All the input factors are results of different measurement processes or determination processes. Both the situation and the reuse multipliers either increase or decrease the total amount of effort estimate. The values of these multipliers vary around 1,0. Although the calculation methods are important parts of the meta-design of the ISDT for PEMT, the calculation processes and their counting rules are not included in this thesis because they are publicly available (refs) and would add unnecessary detail here.

The risk analysis is often connected to the project planning and effort estimating (Boehm, 1989, Stutzke, 2005, p. 443-444), but it is not included into the design theory for PEMT because there is no well-established and validated theory for refining software project estimates based on risk analysis. The estimated risk is simply an informal estimate used in the project plan and contract management to remind about the potential need to change the estimate when the risks occur.

The main functions of PEMT are estimation and benchmarking. First estimation is discussed. The CHAPTER 4 discusses about benchmarking, in other words software process improvement and organizational improvement. The effort estimation process of PEMT consists of FSM (FS), situation analysis (Situation Multiplier), reuse analysis (reuse multiplier), and delivery rate determination. The estimation process of PEMT should be automated by software.

From the bases of the introduction section, the experience database shall supports data collection, data retrieving and processing data. This enables the analyses of the data and storing the estimates and other results of the analyses into the experience database.

Meta-Requirement: PEMT shall support automated effort estimation through

- **functional size measurement of software**
- **reuse analysis**
- **situation analysis**
- **the determination of the delivery rate based on an experience database**
- **retrieving methodical knowledge related to functional size measurement, reuse analysis and situation analysis from the experience database**
- **performing the estimations and analyses and**
- **storing the estimates and the results of the analyses in the experience database.**

3.3 Functional Sizing / Macro- and micro-estimating

The following section is based on ISBSG (2005, p. 6-7). Software estimation can be divided into two approaches, Macro- and Micro-estimation. Macro-Estimating can be further divided into equation use, comparison and analogy. Micro-Estimation holds for work breakdown.

Equation use method is based on a depth of historical data. The project size equation is derived from experience database. Indicative estimate is ideal in early phase of the project. However, the estimation is not accurate and the equation used must be relevant to compared project. Comparison “involves

finding a group of completed projects with similar project attributes to those of the proposed project, assessing the median project productivity delivery rate and speed of delivery for each attribute, taking the average of the medians and using the result to provide a guide for the estimate of the effort and duration for your new project." (ISBSG, 2004, p. 6) Productivity can be described as Project Delivery Rate, as described in (SECTION 3.6.5, p.61). Product delivery rate can be indicated "as the number of hours taken to deliver one unit of functional size" (ISBSG, 2004, p. 6) Analogy stands for method, which is based to benchmarking a project with very similar attributes. The project delivery rate and speed of delivery are used to guide the estimate of the effort and duration.

Functional size is related to both macro- and micro-estimating approaches. This is described in TABLE 1.

Approach	Use of Functional Sizing
Macro-Estimating	Functional Size is a key input to most estimating equations and project comparisons.
Micro-Estimating	The Functional Size allows you to calculate the 'expected' project delivery rate for comparison with past achievements.

TABLE 1 Use of Functional Sizing in macro- and micro-estimating (ISBSG, 2004, p. 9)

Meta-Requirement: PEMT shall support automated macro-estimating and micro-estimating

3.4 PEM Processes

Software PEMTs are built to support software development and maintenance projects. The FiSMA Scope Management Concept is discussed in the context of ISDT for PEMT. It consists of the five PEM processes integrated with the information services provided by PEMT artifacts. This is discussed in more detail in this section and in (SECTION 3.4 p. 41).

According to OTA (2005), FiSMA Scope Management Concept can be divided into initiating project and software development, estimating cost and duration, developing software, controlling progress, managing changes and closing development. FIGURE 8 presents FiSMA scope management processes. These processes are general and derived from the bases of PMBOK (2004) project scope management. FiSMA Scope Management Concept is used for defining PEM processes. These five FiSMA scope management processes are partly parallel and cover software development from the requirements specification phase to the installation of the software. Initiation and estimation of PEMT needs to be performed before the development, controlling and managing processes. (OTA, 2005) “When the initiation process meets the closing process, the customer satisfaction increases” (Forselius, 2005).

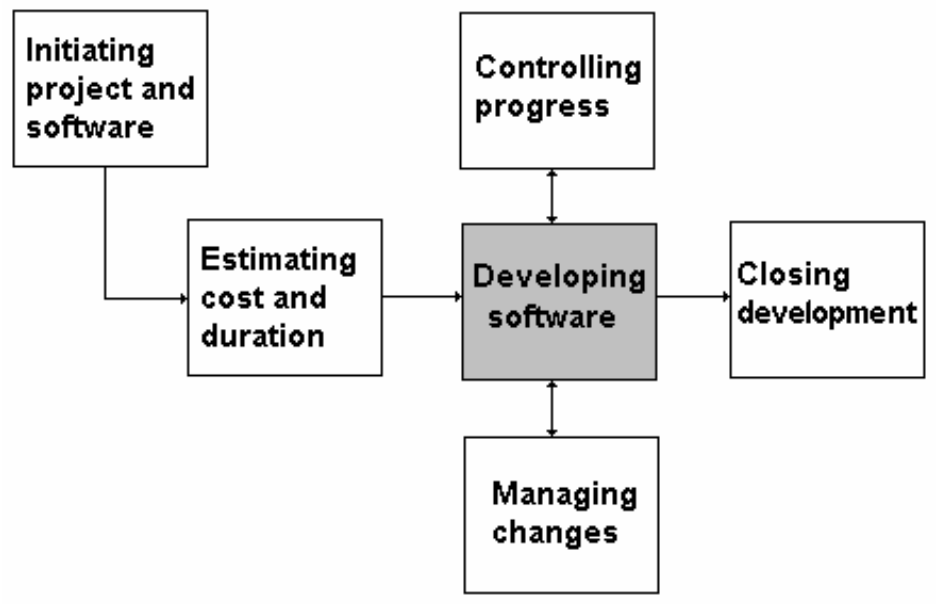

FIGURE 8 PEM processes and PEMT use cases (OTA 2005)

Initiating project and software

Initiating project and software starts from the bases of requirements specification. The requirements of the software are required input information for the project. Initiating project and software covers initiating the project to be managed and initiating the software to be developed. Both of these should be simplified to keep the management of the project and measurement of the software comprehensible. The initiation process should be repeated till the software scope is compact. FiSMA initiating project and software define the project and target software and selected measurement method. The result from the initiation process is an updated version of the project data. (OTA 2005, Forselius, 2005)

The definitions of scope planning and scope definitions respond to the initiation phase. The scope management plan should guide the initiation phase in order to support project management team by defining how the scope should be defined, verified, controlled and how the WBS should be created. The tools, data sources, methods, processes and procedures, and other factors that affect

the project's size, complexity, and importance should be defined. Scope definition describes what is included and excluded in the project, while a scope statement defines major deliverables, assumptions and constraints. It also describes the project scope and objectives. Scope definition's major task is to develop a detailed project scope statement as the basis for future project decisions. Project scope statement support PEMT by setting the scope and boundaries of PEM by defining the project and target software.

In initiation phase, PEMT should support the methods, tools and the used Work Breakdown Structure (WBS) to be selected. It should also adjust classification questions and available methods. Experience database should support and standardize their selection. As a consequence, PEMT should support updating experience database.

Meta-Requirement: PEMT artifacts shall support the initiation of project scope and the storing of project data in a standard format by

- **selecting methods and models,**
- **utilizing classification questions**
- **adjusting available methods**
- **defining the target software**

Estimating cost and duration

The cost and duration estimation process depends on the size of the software, available knowledge of the project, and the quality of requirement specification. The better and more exact the initiation process is, the better success factors the estimating process has. If the documents are well prepared and if similar types of projects have been run before, it is relatively easy to estimate the effort and duration of the project. The cost and duration estimation of the software is done from the bases of the initiation phase. The main function in estimating cost and duration is to collect and analyze all required information related to effort and duration of the project. FiSMA process of estimating software development cost and duration consists of functional size calculation, reuse rate calculation,

situation multiplier calculation, recommended applicable target delivery rate determination from the bases of similar projects, risk level and risk index estimation, data storing and report delivery. The FiSMA process of estimating software development cost and duration is shown in (FIGURE 9 p. 51). (OTA 2005, Forselius, 2005)

The PEMT estimating process starts with the FS calculation, which is done from the bases of functional user requirements. The next phase is reuse rate calculation. It is made from the bases of reusability requirements. The next phase is situation analysis, which is done from the bases of available resources and non-functional requirements (see Appendix 1). The target delivery rate is estimated based on the knowledge of external databases and recent projects. PEMT support standardized methods to calculate the FS, reuse rate, situation multiplier, and delivery rate. The review and results needs to be combined from the bases of scope management plan and scope definition. PEMT should support data storing into the experience database and report delivery to and storing in other project management systems that support activities and task that beyong the scope of PEM such as detailed coordination, communication, and collaboration of actors within and between projects. (OTA, 2005)

The deliverables of the PEM process "estimating cost and duration" are the estimated total cost and duration of the project as well as the following factors used to calculate the estimates: functional size of the software, the impact of reuse, the situation multiplier based on the analysis of situational factors, and the target delivery rate. Combined and calculated result can be drawn from there bases. In addition, appendixes of project contract and project plan are developed. Scope verifications should be done from the bases of estimating cost and duration. Scope verifications include activities of measuring, examining, and verifying to determine whether work and deliverables meet requirements.

The delivery rate is a micro-estimation process where the current project is benchmarked against a group of projects with similar development circumstances. When the analysis of the development circumstances is done, the delivery rate needs to be defined. The Experience® database can be sort according to selected criteria. "Projects have been categorized by development type, business area, target platform type and software development tool" (Forselius, 2005). If a company specific database is available, the results may be more reliable. "The delivery rate is expressed in hours per functional size unit, h/fp." (Forselius, 2005)

"You should never rely on a single estimation method for a project. The more cross and sanity checks you can employ the better." (ISBSG, 2005, p.7) Additional databases are provided by COCOMO, IFPUG, ISBSG and FiSMA. "Software delivery rate is based on past experience, which may not be relevant at present" (ISBSG, 2005, p.7).

The estimation process of PEMT must be based on standardized data collection methods and standardized calculation processes. If the data stored to experience database is not standardized, the past projects are not comparable and can not be used in PEM delivery rate calculation and benchmarking. From these reasons, PEMT must store standardized questions to identify stored data related to functional requirements, reusability requirements and reusability requirements (see Appendix 1). From the bases of standardized data, PEM calculation may provide reliable results. The questions defined in PEMT should reinforce the functional requirements, non-functional requirements and reusability requirements into a level which is needed for precise project data calculation.

These phases can also be seen in the (FIGURE 9, p. 51).

Meta-Requirement: PEMT artifacts shall support cost and duration estimation by

- **storing and collecting data related to project size, cost and duration in a standardized format**
- **identifying the requirements of the development environment**
- **collecting functional user requirements**
- **determining non-functional requirements**
- **estimating reusability requirements**
- **calculating the estimated size, cost and duration**
    - **target delivery rate determination from the bases of functional size**

One result of estimating cost and duration is the delivery rate calculation. This is done from the bases of experience database. This can also be seen in the (FIGURE 9, p. 51).

Meta-Requirement: PEMT artifacts shall support delivery rate determination from the bases of experience database and external databases

Developing software

The development of the software is not covered in PEMT because it is not a part of project scope management. Instead, the development process is a product process. During the development of the software, progress is controlled and changes are managed. These will apparently affect the estimates and measures of the software made in the estimation cost and duration process.

Controlling progress

Controlling progress is a repetitive process. If the total duration of the development project is several months, typically the progress reviews are done monthly. (Forselius, 2005) According to PMBOK (2004) it is a parallel process which objective is to get verification that each deliverable is completed satisfactorily.

The controlling process of FiSMA starts from progress review. New version of the development project need to be estimated and new calculation needs to be drawn from values of variables. The readiness rate is calculated and a comparison report is developed comparing the new and old version of the project. The readiness rate should also define finished project deliverables. (OTA, 2005)

The control process of PEMT supports scope control. New version and new values calculation. Readiness rate calculated should be included into PEMT. As a consequence, it should support changed progress reporting.

Meta-Requirement: PEMT artifacts shall support progress controlling through

- **change control, estimate version control**
- **readiness rate definition of the project deliverables and new readiness rate calculation**
- **updated measurement data storing**
- **new version progress reporting**

Managing changes

It can be said that changes in the scope of the project make changes to the time, cost and quality of the project.

Managing change process is needed during the execution of the software development. Changes frequency depends on change requests. It may be organized to perform at for example regular monthly meetings. (OTA, 2005) The change management manages the factors that create project scope changes and controls the impact of these changes in order to avoid project scope creep. Scope control defined in PMBOK (2004) supports the change management process of PEMT.

FiSMA management process starts from changing requirements, gets into new version estimation and FS calculation. New non-functional requirements may change the values of variables of situation analysis. It needs to be recalculated. FiSMA change management provides comparison report of the old and new version of the project. (Forselius, 2005)

PEMT support new functional requirement and new non-functional requirement changes to new/updated functional size calculation and new/updated situation analysis. Support for old and new software version comparison and control should be provided. New report support is also needed.

Meta-Requirement: PEMT artifacts shall support change management and its linkage to the estimation process through

- **analyzing the impacts of changed or new requirements on the project**
- **calculating new version of the estimate based on changed functional size, situation multiplier, reuse rate**
- **comparing previous and changed versions of the estimate**
- **delivery of a comparison report**

Closing development

Closing development phase includes definition of the final version of the development project. Closing software development is important owing to information transfer to the organizational memory. The closing process includes gathering and packaging statistic and measurement information related to the project. The data storing to the experience database is an essential stage because it supports organizational learning and process improvement. (OTA, 2005)

FiSMA closing development starts from the definition of the final version of the project. This includes functionality closure and final estimate version identification. Duration and still reminding errors are defined, however, they

are not stored into Experience® database. The actual productivity and delivery speed etc are determinated. The details of methods used are stored. Reports are defined and delivered. (OTA 2005, Forselius, 2005)

The closing development process of PEMT supports functionality closure and final version definition of the project. Reminding duration and errors should be defined. The final project data needs to be entered to PEMT. After the final data is entered, the final productivity and delivery speed can be determined. From these bases, final reports are created.

Meta-Requirement: PEMT artifacts shall support development closure through

- **functionality closure and final version definition**
- **reminding duration and error definition**
- **final data entry**
- **final productivity and delivery speed determination**
- **delivery of final report**

3.5 Actors of FiSMA

FiSMA actors, processes and their relations are described in this subchapter in order to define, what the organizational structures are that ISDT for PEMT should support and how they affect the design of PEMT (Meta-design). FIGURE 9 presents the effort estimation process of FiSMA. The actors of FiSMA were discussed and characterized in (FIGURE 5, p.34) and (FIGURE 6, p.36). The FIGURE9 show also a partial meta-design for PEMT by defining the structure of the estimating cost and duration.

The following chapter is based on OTA (2005) and Forselius (2005).

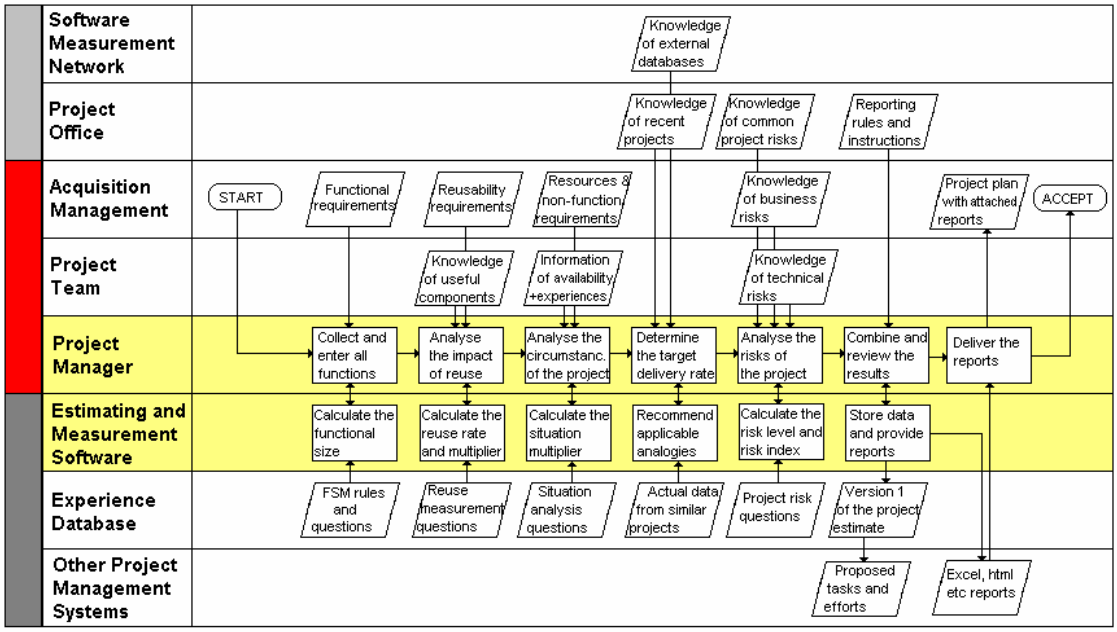

FIGURE 9 FiSMA Effort Estimation Process (OTA 2005)

Software Measurement Network

As described in the introduction section, FiSMA network consists of (1) member companies, organizations and universities, (2) working groups that develop the FiSMA methods and (3) FiSMA administration. The working group consists of international and Finnish researchers, and organizational members

Software measurement network, which can be seen in (FIGURE 6 p, 37) shares software development related information via Experience® database. Project team consists of project manager and project office.

Project Office

Project office is a part of project organization that collects data to experience database and shares project related knowledge. (OTA, 2005)

Acquisition management

Acquisition management is considered as the stakeholders (see Appendix 1) and broad spectrum of acquisition. The customer can also be internal, form the

same organization. Acquisition management is the starting point for software development, fitting a software or software implementation (Forselius, 2005).

The FiSMA Effort Estimation Process starts from functional requirements, reusability requirements and resources & non-functional requirements. Acquisition management initiates the project and the software. It defines the input information of the project. Functional requirements works as an input for FSM, reusability requirements for reuse management, resources & non-functional requirements for situation analysis, and knowledge of business risks for risk analysis.

Project team

Project team is defined as the development team, which characterizes the input for situation analysis. They provide information about technical risks of the project.

Project Manager

Project manager collects, analyses, combines project related information and determines the boundaries of the project. They are responsible for estimating cost and duration, controlling progress, managing changes and project closure.

Estimation and Measurement of Software

Estimation and measurement phase provides reports of calculation and recommendations. The measurement software automates the calculation and estimation processes.

Experience® database

Experience databases store project information of FSM rules and questions, reuse measurement, data from projects, project risk and version information of the project.

3.6 FiSMA Measurement Methods

This thesis creates PEMT from the bases of FiSMA methods and techniques. Next, the FiSMA methods are taken into in depth exploration. FiSMA FSM is the most commonly used FSM in Finland. The Experience ND21 situation analysis method is unique. It is critical when talking about benchmarking and new benchmarking possibilities. FiSMA RMM is used because it is related to the other methods. Delivery rate is determined from the bases of similar projects in an experience database. An in depth analysis of all FiSMA methods is needed to explain the PEMT field. In addition, exploration of only some of FiSMA methods would not give a comprehensive understanding of the research area of PEMT.

FiSMA has created software measurement and estimation methods: FiSMA Functional Size Measurement method 1.1 (FiSMA FSM), FiSMA reuse measurement method (FiSMA RMM version 2002), Situation analysis model (Experience ND21) and Situation analysis model (Experience ND22). They are further discussed in the following subsections.

The following sections 3.5.1 - 3.5.6 are based on the methods published by FiSMA 2004, FiSMA 2001 and FiSMA 2002 and OTA.

3.6.1 FiSMA Functional Size Measurement Method 1.1

Functional size is used for measuring software development activities and development alternatives. This section discusses the definitions and concepts of the FiSMA Functional Size Measurement Method 1.1 method. Functional size is not only used in estimating and productivity analysis but also in project planning, tracking, control and contracting. The purpose of FSM is to provide the size of the software in change management, measuring reuse rate, estimating effort of a project, determining the price for maintenance, the price of a software product, the productivity after a project and other purposes.

FiSMA FSM is used to measure the functional size of a piece of software. It was developed from the bases of 700 software development projects measurement information. FiSMA FSM conforms to the ISO/IEC 14143-1 international standard.

The primary users of FiSMA FSM are associated with the acquisition, development, use, support, maintenance and audit of software. FSM works best with a complete list of functional user requirements and services. When they are involved, FSM makes scope management and change management more effective and reliable. The correctness of counting parameters and thus the usefulness of a FSM method can be evaluated based on the correlation between functional size and effort under similar environmental and technical circumstances, and quality requirements (FiSMA 2004). The evaluation of these parameters may specify a need to take a closer look to the counting parameters used to derive functional size.

Each peace of software shall be measured separately by FiSMA FSM and then combined to total functional size. It is the sum of the functional sizes of the pieces of the software.

FiSMA FSM is built on the bases of assessment of the Functional User Requirements. Functional User Requirement is a function that works as a base functional component to the FSM method. A Base Functional Component (BFC) is an elementary unit of the functional user requirements. They are defined by FSM method for measurement purposes. FiSMA FSM BFC's are named as functional services.

FiSMA FSM categorizes BFCs to seven classes. Each class consists of a defined group of BFC types. A BFC type is a category of BFCs that may include several different types of functional user requirements. FiSMA FSM method's seven BFC classes are composed of 28 BFC types. The identified BFC classes are: interactive end-user navigation and query services, interactive end-user input

services, non-interactive end-user output services, interface services to other application, interface services from other applications, data storage services and algorithmic and manipulation services. These classes are discussed more detailed in the following paragraphs.

Interactive end-user navigation and query services is a procedure which specifies interactive user interfaces which have not got maintenance of persistent data storage(s) of the system. They are divided into seven BFC types which are icons, log-in and log-out dialogs, menus, selection lists, data inquiries, generation dialogs and browsing lists.

Interactive end-user input services, in turn, specify interactive user interfaces where maintenance of persistent data storage(s) of the system exists. Those are the business tasks which change the data contents of the system. This procedure considers different data items, needed reading and writing references to entity types.

Non-interactive end-user output services is a procedure which specifies user interfaces which are non-interactive and do not maintain persistent data storage(s). This includes data items of the BFC's and the number of needed reading references to entity types. They are divided into four BFC types which are output forms, reposts, e-mails and text messages and monitor screen outputs.

Interface services to other applications specify automated data transfer functions sent to other applications or device. These BFC types are messages to other applications, batch records to other applications and signals to devices or other applications.

Interface services from other applications is a procedure which specifies automatized data transfer functions, data groups received from other applications or devices. Here inbound interface functions are divided into three

BFC types which are messages from other applications, batch records from other applications and signals from devices or other applications.

Data storage services are concerned as functional services: collected or related data which the user requires the software to provide from data storages. These are usually termed as entity types, data groups, data classes or object of interest. FSM divide data storage services into two BFC types which are entities or classes and other record types.

Manipulation services, specifically algorithmic and manipulation services are user-defined data manipulation routines which are not included in the normal functionality of any other BFC type. FSM method divide these into six BFC types which are secure routines, calculation routines, simulation routines, formation routines, database cleaning routines and other manipulation routines.

The measurement process of FiSMA FSM consists of two main parts: measuring the end-user interface services and measuring indirect services. The measurement process is presented in (FIGURE 9, p.51). The process counts the size of each 28 BFC units. All of the boxes present in FIGURE 10 follow a similar procedure:

1. How many of this type of BFC's does the piece of software have?
2. What are they? They need to be identified.
3. What are they like? The number of details for each BFC has to be identified.

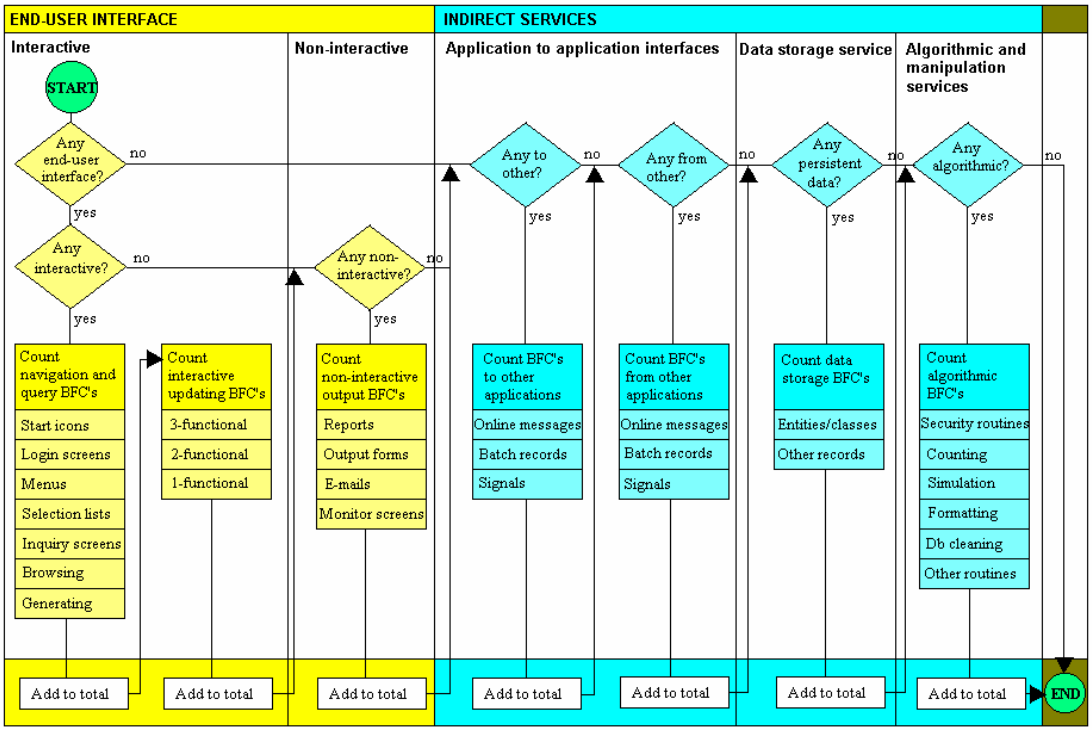

FIGURE 10 FiSMA FSM Process (Forselius, 2004, p. 10)

Each BFC type has specific counting rules, however, they are not defined in this thesis. The functional size of a piece of software is calculated by calculating the functional sizes of the seven classes identified in the previous paragraphs.

Some FiSMA methods have evolved to a position of theories. For example, FiSMA FSM 1.1 is no longer in a position of an artifact. Instead, it has achieved a position of a theory by practical utilization of both of the loops of behavioral science and design science shown in (FIGURE 1, p. 17).

3.6.2 Experience ND21 Situation Analysis Model

Experience ND21 Situation Analysis Model is a process which helps to estimate new development and enhancement projects. The model is classified into four categories: project, process, product and people. These categories consist of 21 productivity factors. These factors are rated using five alternative values, which are:

"++" = Excellent situation, circumstances much better than in average case

"+" = Good situation, circumstances better than in average case

"+/-" = Normal situation from the productivity point of view

"-" = Bad situation, circumstances worse than in average case

"—" = Very bad situation, circumstances much worse than in average case

Each factor is weighted on the basis of experience data. The four categories are presented later in this section.

Project Factors

1. *Involvement of the customer representatives*. This part measures how actively the customer can be interpreted as an end-user, purchaser or a defined market segment, or participates in the development work.
2. *Performance and availability of the development environment*. This phase involves the performance level evaluation of the tools, hardware, physical environment, network and equipment. The shortcomings or fulfilment of all the work related needs of these resources should be under review here.
3. *Availability of IT stuff*. This considers the availability of software personnel during the project, problems in personnel availability or readiness to possible overloads. The software personnel quality and qualifications should also be considered.
4. *Number of different stakeholders*. This relates to the total number of involved organisation units and parallel dependent projects.
5. *Pressure on schedule*. Here the pressure of schedule, goal schedule is compared to the first estimate of analogy.

Process factors

1. *Impact of standards*. Here the quality of the existing standards and procedures applied in the project, their flexibility and familiarity for the team are under evaluation.

2. *Impact of methods*. The exploited methods quality and use in the project. Methods in programming, testing, documenting and project management are evaluated. This appraises whether methods are used and whether support for them is well organized.

3. *Impact of tools*. This contains the use and quality of the tools available for the project. If the project uses a minimal set of basic tools such as editors, compilers and simple debugging tools or in the other end integrated CASE- environments which covers the whole lifecycle.

4. *Level of change management*. This part evaluates the requirements stability and predictability, and also maturity of the change management processes. The contract for the project is evaluated in terms of stabilizing the requirements and specifications. New requirements, functions and features expand the project.

5. *Maturity of software development process*. This evaluates the stability and conformance of the software development processes. Also tool support and quality models during the lifecycle are under evaluation.

Product factors

1. *Functionality requirements of software*. This concerns the software compatibility with the end-users, complexity of the requirements and level of integration. In a case where the system is developed for complex application area, multi-tier system for various, multi-cultural users, the evaluation factors drops. On the other side is an easy application area with a small application developed for a small group of users.

2. *Reliability requirements of software*. This part measures the maturity, fault tolerance and recoverability. In the case where operation faults may endanger human lives or cause great economic or environmental losses, the applications must recover from faults. In the case where the system is a large, integrated real-time system or has extremely high maintenance costs, the product factors drop.

3. *Usability requirements of software*. This involves understand ability and easiness to learn the user interface and the logic of the work-flow.

4. *Efficiency requirements of software*. This part covers effective use of resources and appropriate performance. Here the requirements for databases, data records, simultaneous end-users and complexity of data requests are evaluated.

5. *Maintainability requirements of software*. Here the stability environment is evaluated. How frequent the changes are for the software and how long the lifetime is. Laws and standards and business rules are evaluated.

6. *Portability requirements of software*. These evaluate the adaptability and install ability to different environments, possible different organizational environments, and openness of architecture, various platforms and structural components.

People Factors

1. *Analysis skills of stuff*. This takes into consideration whether the project personnel has had similar projects and evaluates the skills and experience of the personnel in requirements analysis.

2. *Application knowledge of staff*. This analyzes both the supplier and customer knowledge of the application domain in the kick-off moment.

3. *Tool skills of stuff*. This evaluates both the suppliers and customers' average experience on development and documentation tools at the kick-off moment. All tools must be known.

4. *Experience of project management*. This evaluates the project manager's and other key personnel's previous experience of similar projects.

5. *Team skills of project team*. This evaluates the project team working capabilities as a whole, project and collaborations skills. Do they follow the best project practices?

3.6.3 Experience MT22 Situation Analysis Model

The previous section discussed Experience ND21 Situation Analysis Model. It helps to estimate new development and enhancement projects. Experience MT 22 Situation Analysis Model estimates annual maintenance and modification projects. The model is composed of 22 standard productivity factors, and is classified into 6 organization factors, 5 process factors, 6 product factors and 5 people factors. However, this thesis looks only into new development and enhancement projects of PEMT. Factors that are related to maintenance and modification projects are out of the scope of this thesis. MT22 Situation Analysis Model is not discussed more detailed in this thesis.

Meta-Requirement: PEMT artifacts shall support new development, enhancement, maintenance and modification projects

3.6.4 FiSMA Reuse Measurement Method, FiSMA RMM version 2002

FiSMA RMM is a process which is built to the basis of FSM and a model of software project deliverables. RMM measures internal and/or external reuse. Both internal and external reuse can exist in one project. The model of software deliverables address those project deliverables, which are directly related to

each functional user requirement. These deliverables are program code, test cases, software documentation (for maintainers), and user documentation (for users of the software). Each of these represents certain percentage of the total effort needed in the implementation of the functions. The default weights of the deliverable types are:

- Program code 40%
- Test cases 30%
- Sw documentation 20%
- User documentation 10%

FiSMA FSM and a complete list of base functional components work as an input for FiSMA RMM. The impact of the reuse may be negative or positive. If a deliverable of any function must be developed to be reusable, the impact to effort will be the weight of the deliverable multiplied by the size of the function (Forselius, 1999).

3.6.5 Delivery Rate

Software project delivery rate can be indicated as work hours divided by FPs. The base delivery rate can be sought from Experience® database.

An analogy should be selected for benchmarking. The analogies vary from project type, business domain, software production environment type and the prevailing implementation.

3.6.6 Software Project Risk Estimation

By definition, risks are not very probable factors, but if they do occur, they may have a great impact to the effort and cost of the project (ISBSG, 2005, p.14). Software project risk management can be divided into risk identification, influence analysis, risk priority, risk management planning, risk reduction, and risk tracking. Software project risk estimation is a process that estimates risks in

terms of uncertainty, impact and defeat costs. Experience Top21-risk model divides risks into 21 questions, which are the most typical ones that needs to be analyzed. Risk estimation is included into Experience database. Risk estimation is not considered into the ISDT for PEMT because is not a measurable calculation process. However, it needs to be discussed when talking about PEMT. Risk is added to Project Planning phase as an estimate made by actors. Risk aids project planning, it is nice to know or additional information to the project plan. When risk is added to the project estimate, the result is a guess of the project length, not a formal, calculated number. If the risks are especially high, software vendors may not want to do the reject at all.

3.7 FiSMA Calculation Process

PEMT support software size calculation. According to OTA (2005) total work effort estimate can be calculated by multiplying FSM, delivery rate, ND21, RMM. In addition common sense assessment and risk factors can be taken into consideration. However, risk and common sense assessment are out of the scope of PEMT because they are not formally counted. They are guessed.

As mentioned earlier in (SECTION 3.3 p, 39), PEMT should support the automated calculation of the estimation process.

3.8 Processes Required by PEMT

In addition, design theory requires the fulfillment of the following meta-requirements to meet PEM-process requirements. These factors work as inputs for the PEMT process. They are required in order to make the PEM process exact. While PEM can be executed without exact input information, the estimates may not be exact.

PEMT requires exact definition of the functional requirements, reusability requirements and non-functional requirements. They should be exact because PEM is derived from the software requirements. They work as input

information for PEMT as shown in (FIGURE 9, p. 51). The changes in reusability requirements, recourses and non-functional requirements need to be updated throughout the project life-cycle in order to keep the estimates up to date.

In addition, information about changes and project deliverables needs to be updated in PEMT. The changes may change the target delivery rate. As a consequence, in may need to be recalculated.

Meta-Requirement: PEMT artifacts shall require

- **effective requirements process throughout the project life-cycle (specifically during initial scoping and analysis and whenever new change requirements or constraints require change management and impact analysis) to yield high quality specifications for calculating functional size of the software**
- **evaluation of reusability requirements**
- **evaluation of resources and non-functional requirements**
- **information about changes**
- **information about project deliverables**
- **identification in target delivery rate**

3.9 Conclusion

This chapter identified why PEM is needed and how PEMT can support the PEM process. ISDT meta-requirements were defined from the bases of FiSMA measurement methods and processes. Partial meta-design was briefly outlined from the bases of the actors defined in (FIGURE 9, p. 51). The estimating process consists of FS calculation, situation multiplier, reuse multiplier and delivery rate determination. These phases support the PEM and should be used in PEMT.

4 BENCHMARKING

The (SECTION 3.6.5, p.61) discussed delivery rate determination. The quality of data was also under consideration earlier. It is further discussed in this section. However, it is discussed more detailed here. This section takes benchmarking into consideration. New possibilities of benchmarking data are further discussed.

As mentioned in the introduction section, experience databases can be used for two purposes: delivery rate determination and process improvement. In addition, Experience® database can have two dimensions. One that is a combination all projects and collection of all the customer organizations, the other that is customer organization specific.

Benchmarking is a macro-estimation process where one project or a group of similar projects are compared against the development project. Experience database is a project performance and functionality database that records a grate amount of project history data that can be benchmarked with current software development projects. It stores 700 projects data database which can be divided used and compared according to different search criteria, for example, according to project or business sector. (OTA, 2005)

It is important to not to relay on one method of estimation. It is advisable to use different techniques to provide checks for the preferred technique. While the concentration of this thesis is in macro-estimation using history based databases, it is important to make reference to the essential micro-estimation technique, commonly referred to as a work breakdown structure. (ISBSG, 2005, p.50)

Form these bases, PEMT should support experience database and different techniques to provide more accurate and reliable benchmarking results.

Meta-Requirement: PEMT artifacts shall support benchmarking with different databases

PEMT should support top-management decision making through benchmarking, as shown in (FIGURE 5 p 35). The data values can be benchmarked in order to find out, whether the performance of the software development has changed.

Meta-Requirement: PEMT artifacts shall support top-management decision making, new goal setting and project steering through

- **data benchmarking**
  - **deriving product and process development goals from the business goals**

FiSMA Situation analysis was described in (SECTION 3.5.2, p.56). Situation analysis supports also process improvement. The benchmarking process may identify, which are the factors, that the organization does well and what are the weaknesses of the organization. PEMT should support this process by using situation analysis and by analyzing the results that can be derived from them.

Meta-Requirement: PEMT artifacts shall support software process improvement and organizational improvement through

- **during the development project**
  - **progress controlling and change management**
    - **continuous analyzing and development of project, process, product and people situational factors**
- **benchmarking**
  - **comparing current situation to past performance**
  - **process development**
    - **IT tools and software engineering environments, and development methodology improvement**
    - **development processes improvement**
    - **people development, co-workers, their competencies and responsibilities**

4.1 Quality Concept and Data Quality

According to ISBSG (2005, p. 7) "Any technique is only as good as the data and information on which it is based. You cannot expect any technique to compensate for lack of definition, understanding, or agreement of scope of the software job to be done." When benchmarking experience databases, data quality must be taken into consideration. Even if the measurement and estimation is done accurately and thorough, the benchmarking results might fail due to bad data quality.

Internal data quality is the capability of a set of static attributes of data that satisfy stated and implied needs when the data are used under specified conditions. In practice this is the quality of data gathered to Experience database. External data quality is the capability of data that enables the behavior of a system to satisfy stated and implied needs when the system is used under specified conditions.

Data quality in use can be defined according to Forselius (2005) as the "capability of the data to enable specific users to achieve specific goals with timeless, amount of information, relevancy, credibility and understandability in specific contexts of use".

"Errors can arise from several sources: inaccuracies that are inevitable when an estimate is produced from incomplete information; inaccuracies that might be caused by using a poor technique to do the estimate; and "scope creep" when the final system contains functionality that was not part of the initial specification." ISBSG (2005, p. 24)

Analogy based estimation differs from the comparison based estimation. "Analogy operates with one, or perhaps two past projects selected on the basis of their close similarity to the proposed project. Comparing a planned project to a past project is commonly used in an informal way when "guesstimating",

consequently it is a familiar technique to the practitioner." ISBSG (2005, p. 47) Term guesstimating is used as project planning where the guessed element, risk is added to the project estimate.

The quality of data in FiSMA is managed by a centralized unit which is demonstrated in (FIGURE 6, p.36). As defined earlier, PEMT should support data collection and data distribution by experience database. The data is collected from measurement network. PEMT should also support data quality assurance.

The data quality is an important factor in PEM and data collection. PEMT should support the quality management process.

Meta-Requirement: PEMT artifacts shall support data quality assurance

4.2 Benchmarking Data

According to Forselius & Maxwell (200, p.81-82) the most significant productivity variance in Experience® database is derived by company variables. The variances are presented in FIGURE 11. As consequence, when benchmarking, companies should establish their own software-metrics databases. In addition, the project estimates should be benchmarked against other projects in Experience® database. The next most significant benchmarking classification variable was the customer company's business sector when compared among the banking, insurance, manufacturing, wholesale-and-retail, and public administration.

Application programming language, CASE tools, and software centralization did not seem to be that significant. However, jointly they make ND21 a significant part of estimation process.

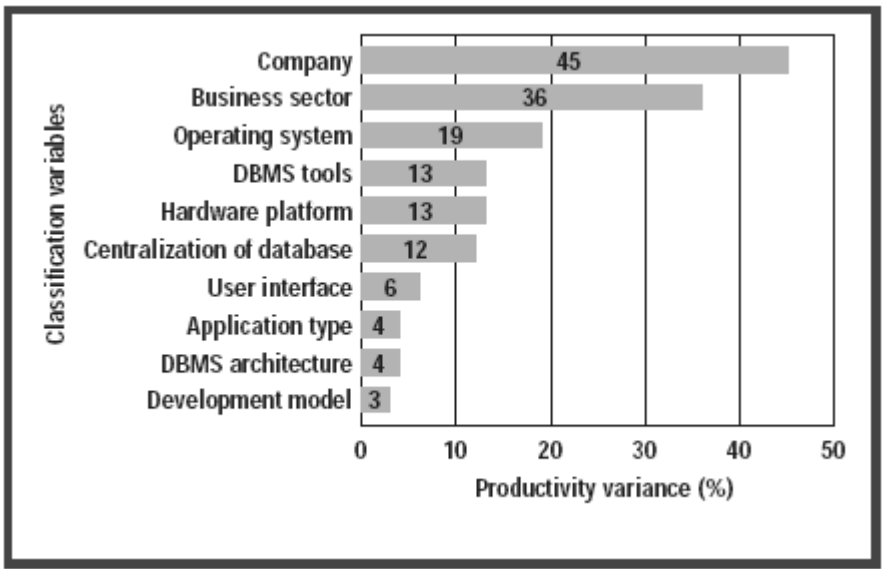

FIGURE 11 Experience® database variance of accounted classification variable (Forselius, Maxwell, 2000, p. 82)

FIGURE 12 presents the main productivity (delivery rate) of each industry sector. The manufacturing sector has the highest productivity rate, the banking and insurance sectors the lowest. It makes grate difference, how benchmarking is made. The development environment competitiveness, in-house production, quality requirements, requirements volatility, and the core operations dependencies make also variance to the measured productivity. It is critically important to recognize the key productivity factors, which influences the benchmarking results. This domain specific, organizational specific and project specific knowledge needs to be managed.

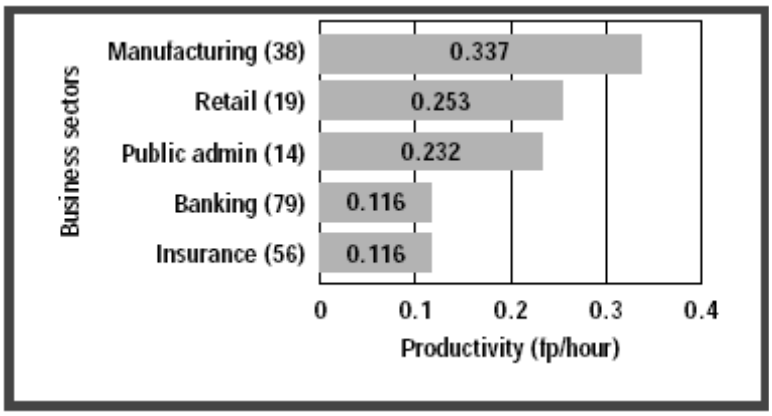

FIGURE 12 Experience® database productivity by business sector (Forselius, Maxwell, 2000, p. 82)

As a conclusion, benchmarking results differs in productivity between business sectors. Due to that reason, PEMT should support data classification and business sector selection when the delivery rate is determined and when benchmarking.

Meta-Requirement: PEMT artifacts shall support database classification and business sector selection in

- **delivery rate determination**
- **benchmarking**

Different companies have different productivity results. Therefore companies should collect and benchmark their own company specific data. Due to those prerequisites PEMT should support domain specific benchmarking, which is justified in FIGURE 12. To gain compatibility, top-management information and results reliability, PEMT should support organizational data benchmarking.

Meta-Requirement: PEMT artifacts shall support domain specific and organization specific data benchmarking

4.3 Possibilities in Benchmarking Data

By collecting project data, companies can learn from their own experience. On other words, by benchmarking data, companies can learn by other companies' experiences. (Forselius, 2005) When collecting data for estimating and project planning, companies' awareness increases. Situation analysis and delivery rate can give an enormous amount of information about the development factors. Furthermore company specific experience databases can be benchmarked. In other words, the company specific project measurement information can be benchmarked against previous software development projects. However, more can be achieved when company specific experience databases or single project information is benchmarked against other companies experience databases by selecting only specific group of companies to compare with.

As shown in (FIGURE 6, p. 36), STTF collects and manages Experience® database. This is to ensure the quality of the data and to ensure the anonymity of the projects and companies who provide their project information to Experience® database. It is important to ensure that the competing software companies do not get any specific companies project information out of Experience® database. However, project information of one customer organizations can be benchmarked to any specific and selected group of comparable projects from competing organizations, or to an average performance of comparable projects found from experience databases. Hereby any company can compare its processes, productivity, project size, risk and delivery rate factors against competitors.

A grate variance between companies' measurement results and productivity information can be found from experience databases. Based on the FiSMA measurement methods, the following information can be derived out of Experience® database by benchmarking.

Benchmarking may show what should be improved? By benchmarking productivity and efficiency factors against competitors or similar projects the results may show where the strengths and weaknesses are. Based on benchmarking results the improvement actions priority and amplitude can be identified. (Forselius, 2005)

The quality of processes and methods can be benchmarked from the bases of situation analysis. The quality of processes and methodologies can be benchmarked. This may show that the quality needs to be developed. (Forselius, 2005)

Based on situation analysis, the processes used in projects can be benchmarked. The impact of standards, methods and tools, level of change management and portability requirements of software can be compared. This may show that some other company would gain better results in a comparable project with less method and tool usage. The tools used can be compared from situation analysis. The performance can be calculated from FSM and delivery rate. In other words, the level of standards, methods or tools impact to the delivery rate can be concluded from benchmarking. This shows also the maturity of software development processes. The productivity of the projects can also be compared from FSM and delivery rate. Benchmarking can also show if we are producing products cost-effectively. This can be derived by benchmarking the same kind of projects with similar business sector and equivalent FS to delivery rate. (Forselius, 2005)

It can also be asked if somebody has gained better results from some other development environment. The delivery rate with specific situation analysis factors in any specific environment can be benchmarked against the selected group of projects. (Forselius, 2005)

The skills, knowledge, experience of project management and sustainability of the teams can be benchmarked from situation analysis people factors. They can

be compared to the delivery rate of the projects. The rate of reuse and its impact to total size of the project can be benchmarked. (Forselius, 2005)

The improved benchmarking possibilities will clarify to organizations, which are the best practices for their software project and how to build successful software projects. PEMT should support the different benchmarking possibilities.

Meta-Requirement: PEMT artifacts shall support different benchmarking alternatives

4.4 Conclusion

This section described the benchmarking process and how it should be performed in order to achieve process improvement and organizational learning. The quality of the data is one main element in benchmarking. If the data quality is poor, the estimates may not be exact. Variance can be found between different business sectors productivity. The possibilities of benchmarking data were discussed. A grate deal of benchmarking practices may be found.

5 DUALITY AND DESIGN PRODUCT HYPOTHESIS FOR PEMT

The partial meta-requirements for PEMT were defined in the previous chapters. This chapter creates the connection between the most relevant parts of DIS and PEMT and defines the design product effectiveness hypotheses of PEMT.

5.1 Dual Infromation Systems interconnection with PEMT

A partial link between the models can be identified. Integration of DIS and PEMT has never been done before in scientific research. When DIS is developed to gather very detailed information about actors working and learning, DIS is fully comparable with this thesis only in project benchmarking aspect. However, connections can be found between ISDT for PEMT and ISDT for DIS.

The business layer of DIS can be linked to the SECTION 3. The PEM processes and PEMT discusses about PEM organizational working and the organizational structures of PEMT when DIS discusses about the working and learning of the organization.

The project layer and KSS of DIS relates to SECTION 4, which discussed the benchmarking of PEMT. The process improvement and organizational learning of PEM benchmarking can be seen as the (re)desing & negotiating control of DIS defined in project layer.

The benchmarking of PEMT can also be seen as KSS in DIS. DIS defined that KSS stores both current and historical work process knowledge of the organization. Current project data and history project data is stored to experience database. KSS provides explicit work process knowledge from other units to project layer. This process can be related to benchmarking in PEMT.

The KSS of DIS relates also to the FiSMA network. The FiSMA network manages the experience database, improves the measurement methods and

metrics, collects experience data and shares the new database. It manages the collaboration of the different units of FiSMA.

Top management of DIS answers to the question "what ought to be" when top management of PEM sets new goals from the bases of past project performance. New goals are set from the bases of project results and reports.

The Projects, described in (FIGURE 5, p,35) represents the base production process of business layer of DIS. The project office, measurement network and top management are supportative processes of DIS.

The breakdown layer and the other functions of DIS are out of the scope of this thesis. The Breakdown layer is a subject to future research. The DIS users can be investigated with the guidance of DIS.

5.2 The Design Product Effectiveness Hypotheses

Design product effectiveness hypotheses of ISDT for PEMT are subdivided into FP, situation analysis, reuse analysis, delivery rate, benchmarking and PEMT identification.

Duality implies in PEMT that agents do not fully understand the components and productivity factors of benchmarking. The Dual PEMT (DPEMT) implies if the users do not fully understand the components of PEM and the calculation processes of PEMT.

The knowledge sharing server of DIS is used in this thesis in accordance with PEMT hypotheses to provide organizational learning and development. Experience database cannot be modified. As consequence, the organization and project team needs to modify their working practices and be able to reflect their working practices.

Function point

1. The FSM artifacts designed according to all the feasibility related hypotheses reinforce standardized, accurate, traceable, and transparent size measurement and functional software requirements.

2. When hypotheses 1 holds, it will improve elicitation and documentation of functional requirements.

3. When hypotheses 2 holds, it will improve the estimating, productivity analysis, project planning, tracking, and control.

4. When hypotheses 3 holds, it will improve accuracy of estimates.

5. When hypotheses 4 holds, it will enhance benchmarking and contracting.

6. When hypotheses 5 holds, it will enhance coordination and the number and severity of breakdowns and the time needed to recover from them.

7. When hypotheses 4 holds, it will improve measurement practices and the DPEMT by documenting project size information in the experience database.

Situation Analysis

1. The situation analysis artifacts designed according to all the feasibility related hypotheses reinforce standardized, measurable and transparent project estimation practices.

2. When hypotheses 1 holds, it will improve awareness of actors and improve documentation of project, process, product and people related factors.

3. When hypotheses 2 holds, it will increase motivation and sense of responsibility of actors.

4. When hypotheses 3 holds, it will enhance coordination and to reduce the number and severity of breakdowns and the time deeded to recover from them.

5. When hypotheses 4 holds, it will improve the quality of the organizational products. It will enhance project, process, product and people factors progression and development.

6. When hypotheses 5 holds, it will improve DPEMT further by better documenting project, process, product and people factors of the organization and by storing information about them in the experience database.

Reuse Analysis

1. The reuse analysis artifacts designed according to all the feasibility related hypotheses reinforce standardized, measurable, traceable, and transparent reuse analysis practices.

2. When hypotheses 1 holds, it will increase the awareness of the project team and the acquisition management about the benefits and drawback of software reuse.

3. When hypotheses 2 holds, it will help (1) acquisition management set the right level of reusability requirements for the deliverables of the project and (2) the project team leverage existing reusable components optimally.

4. When hypotheses 3 holds, it will increase delivery rate of the software.

5. When hypotheses 4 holds, it will reduce the total resources required.

Delivery rate

1. The delivery rate artifacts designed according to all feasibility related hypotheses reinforce standardized, measurable, traceable and, transparent project estimation practices.

2. When hypotheses 1 holds, it will improve actor/personnel resource estimates more accurate, and reduce the total resources required.

3. When hypotheses 2 holds, it will improve DPEMT further by better documenting the delivery rate and by storing delivery related information in the experience database.

Benchmarking

1. The benchmarking artifacts designed according to all feasibility related hypotheses reinforce standardized, measurable, traceable and transparent project data collection practices.

2. When hypotheses 1 holds, it will improve planning and documentation of projects.

3. When hypotheses 2 holds, it will improve project time and cost measurement and estimation.

4. When hypotheses 3 holds, it will increase control, awareness of actors and improve management of projects.

5. When hypotheses 5 holds, it will improve DPEMT further by better documenting project related information and storing them in the experience database.

DPEMT

1. The DPEMT artifacts designed according to all feasibility related hypotheses reinforce actors to use standardized and measurable size estimation and measurement practises.

2. When hypotheses 1 holds, it will improve project size measures by using function point analysis, improve situation analysis by using situation analysis, improve reuse measurement by using reuse measurement, improve delivery rate, and improve project data benchmarking.

3. When hypotheses 2 holds, it will increase awareness of the actors.

4. When hypotheses 3 holds, it will improve planning and documentation of the projects.

5. When hypotheses 4 holds, it will enhance coordination and reduce the number and severity of breakdowns and the time deeded to recover from them.

6. When hypotheses 5 holds, it will make project size estimation more accurate by improving project size measures, delivery rate, project development, shorten time-to-market and increase project related information in the experience database and KSS.

7. As a result, the PEMT will perceive the DPEMT more useful and use it more actively. This self-reinforcing cycle can continue from hypotheses 1.

Software project size is a broad concept, which is so far considered to consist of five steps: functional size, delivery rate, situation factors and reuse factors. However, common sense is still needed when estimating software project size. Management and measurement without expert knowledge is equally ineffectual, so we must be careful to avoid over-emphasizing the quantitative

aspects of Software Engineering Management (SWEBOK, 2004, p. 8-2). It is evaluated and added to the risk estimation phase.

5.3 Artifacts Derived From Information Systems Research Framework

This paragraph intends to extend the idea of PEMT by artifact evaluation which provides improved understanding of the problem according to Hevner et al. (2004) As a result of the ISDT, one artifact was created. However, ISRF divides artifacts to smaller peaces.

As mentioned in the Introduction section p, 16 (Hevner et al. 2004 quoting Denning 1997; Tsichritzis 1998) argue that artifacts seeks to create innovations that define the ideas, practices, technical capabilities, and products through which the analysis, design, implementation, management, and use of information systems can be effectively and efficiently accomplished. The ideas practices and technical capabilities are discussed more detailed.

The idea of PEMT:

The idea of PEMT is defined in meta-requirements, meta-design and effectiveness hypotheses.

Practices of PEMT:

The processes of PEMT must be defined, which are initiating, estimating, progress, change, closing. The roles of people must be defined in order to do situation analysis. The project requirements must be defined in to enable functional size measurement. The environmental contexts must be defined for situation analysis. All the PEMT stages must be carried through. A comparable project (analogy) must be found from experience database. Company's own or the shared experience database. Data must be gathered in order to gain company specific experience database. Top management must support the use of PEMT and data collection.

Technical capabilities of PEMT:

PEMT enables data collection and calculation, benchmarking data and expanded possibilities in benchmarking

On the other hand, "The crucial nature of design-science research in IS lies in the identification of as yet undeveloped capabilities needed to expand IS into new realms "not previously amenable to IT support" (Markus et al. 2002, p.180) Such a result is significant IS research only if there is a serious question about the ability to construct such an artifact, there is uncertainty about its ability to perform appropriately, and the automated task is important to the IS community (Hevner et al. 2004, p. 84). Benchmarking one company data against a selected group of companies' project information in order to know which parts of one company's projects are succeeded and which needs improvement.

According to Hevner et al. (2004, p. 82) "Design science research requires the creation of an innovative purposeful artifact". In artifact creation the problem domain is PEMT. "Because the artifact is purposeful, it must yield utility for specified problem" (Hevner et al. 2004, p. 82). Utility can be demonstrated by the benefits gained by using PEMT. However, the demonstration of the utility is a subject of future research.

According to Hevner et al. (2004, p. 78) "The artifacts are specified as constructs, models, methods, and instantiations" This point of view is discussed briefly in the TABLE 2.

Constructs	PEMT has an impact on the way in which information systems projects scope management is managed and executed. The way in which information systems are estimated, measured, monitored and benchmarked. PEMT facilitates project benchmarking.

Models	Software industry delivers products late, over budget, and of poor quality and with uncertain functionality. The problem is project estimation and measurement. FiSMA Measurement Metholologies, PEMT processes and benchmarking Experience Database describe the solution space. All together, they enhance and improve the project estimation and measurement.
Methods	Process descriptions as FIGURES. The instantiation of Best Practices Benchmarking Service of the ISDT for PEMT helps actors search better process solutions from Benchmarking Experience Database for improving project estimation and measurement.
Instantiations	New capabilities in benchmarking.

TABLE 2, Constructs, Models, Methods and Instantiations of PEMT

The artifact identification by Hevner et al. (2004) may broaden the understanding of artifacts presented by Walls et al. (1992, 2004). When ISDT gets deeper into the problem space by identifying the meta-requirements and meta-design, ISRF Design-Science is not that complete. However, ISRF Behavioral-Scinece may extend ISDT by evaluating the results.

5.4 Conclusion

This chapter discussed the connection between the most relevant parts of DIS and PEMT. The design product aspect of PEMT were divided into function point, situation analysis, reuse analysis, delivery rate, benchmarking and DPEMT. The ISDT for PEMT was discussed in order to broaden the understanding of the differences and similarities of the ISDT and ISRF.

6 DISCUSSION

The goal of this thesis was to build a partial ISDT for the class of PEMT. ISDT partial meta-requirements and partial design product effectiveness hypothesis were built. Further benchmarking possibilities were discussed.

This thesis provided value for both scientists and practitioners. The results of this thesis are remarkable for practitioners because most organizations have failed to institutionalize the estimation and measurement processes and there has been lack of external pressure from software businesses leverage PEM. Furthermore, the results of this thesis are remarkable for researchers because only little research could be found on software PEMT and how to use PEMT effectively. To our knowledge, holistic information systems design theory building for PEMT has never been presented in a theoretical research. An ISDT for PEMT has never been published earlier in scientific research. In addition, the link between ISRF, ISDT within compatibility with DIS. DIS does not fully fit to ISDT. However some linking points between ISDT and DIS were presented. In addition, DIS was used to fulfill ISDT for PEMT by guiding the design product effectiveness hypotheses. Utilization of ISRF positioned this thesis in IS research area and fulfilled the ISDT for PEMT. Further the definition of ISRF and DIS gives good future research possibilities from the bases of this thesis. The theories were linked within the practical FiSMA Scope Management Concept. The scope and the theories used in this thesis are exceptionally broad. From the bases of these factors, the results of this thesis are eminent.

The wide knowledge area of PEM were analyzed from the bases of ISBSG (2005), PMBOK (2004), SWEBOK (2004), OTA (2005), FiSMA (2004), (2002), (2000) and Forselius (2005). The PEM created in this thesis were defined from the bases of FiSMA Concept of Organizational Learning, FiSMA actors, PEMT effort estimation, effort estimating for software development, PEMT processes, FiSMA Effort Estimation Process, FiSMA measurement methods, benchmarking

and the importance of data quality. In addition, possibilities in benchmarking data were discussed. The PEM defined in this thesis consists of functional size measurement, delivery rate analysis, situation analysis, and reuse rate measurement. From the bases of FiSMA, all these components of PEMT need to be fulfilled in order to make exact estimates. This thesis defines PEMT to consist of five primary steps: (1) project size calculation, (2) software reuse impact to the effort, (3) delivery rate determination (4) situation analysis, and (5) benchmarking data. These steps were discussed more detailed. All these steps are needed in IS supported PEM. The findings were defined from the bases of the literature review.

By default, ISDT gives additions to ISRF knowledge base. When ISDT creates one artifact, ISRF behavioral science and design science are cyclic models, i.e. they are repetitive. On the other hand, ISDT is more complete and more precise compared to ISRF design science. In order to develop a complete theory for PEMT, both ISDT and ISRF are needed. Evaluated researches results may promote the PEM research by giving guidelines how information systems can best promote PEM. All these theories are based on design theory which goal is to describe, how a design process can be carried out in a way that is both effective and feasible. This thesis claims that if PEMT is constructed following the ISDT for PEMT, the quality of the product increases.

The partial meta-requirements, a part of meta-design and partial design product hypothesis were defined. Further benchmarking possibilities were discussed based on Forselius, Maxwell, (2000) and FiSMA methods.

The knowledge area of PEM and the ISDT for PEMT required an in depth analysis of the literature. Due to that reason, the ISDT for PEMT defined in this thesis needs future research. On the other hand thesis raises a grate deal of future research possibilities in the research area of PEMT. The partial meta-requirements and design product hypothesis need to be validated by behavioral

science research. Meta-design may also need to be defined from the bases of FiSMA. This thesis took only FiSMA methods under research. Other PEMTs can also be evaluated by ISDT for PEMT. Hereby, when many PEMTs are investigated, the PEM research area may evolve and mature. If the results are validated, the findings may either strengthen ISDT for PEMT made from the bases of FiSMA or disprove it. If the theory is disproved, it needs further development. The ISDT for PEMT may evolve over time. The connection between ISDT for PEMT and ISDT for DIS can be further analyzed into more depth level. DIS mode may be further developed to meet the ISDT for PEMT. The breakdown layer of DIS may also be a subject to future research.

The validation of ISDT for PEMT is out of the scope of this thesis. The long term goal of PEMT research may be to further develop PEMT research by deepening the ISDT for PEMT by defining the meta-design, by analysing the link between ISDT and DIS and by further developing DIS model to better meet the needs of ISDT for PEMT. The partial ISDT for PEMT presented in this thesis needs validation. ISDT hypothesis validation is also needed. This thesis introduced the research area of PEMT and suggested theories that may be used when PEMT's are taken under research.

The results of this thesis are valuable because the main components of PEM were defined, and the practices and capabilities of PEMT were introduced. The scientific evidence of the findings of ISDT for PEMT may promote the research area. This is a subject to future research.

The research area of PEM is broad. Due to that reason, this thesis created an ISDT for PEMT based only on FiSMA methods. Partial meta-requirements and design product effectiveness hypothesis were created from the bases of best knowledge available at the particular time the thesis was created. Some parts of meta-design were outlined. However, the findings were not validated. DIS model was processed only partly.

7 SUMMARY

The primary goal of this thesis was to build a partial ISDT for PEMT. The PEM processes were identified and PEMT was defined. Partial meta-requirements and design product effectiveness hypotheses were derived from the bases of the literature review. Further benchmarking possibilities were discussed. This thesis outlined and evaluated PEMT scientifically in order to improve both the scientific and practical understanding of the usefulness and functionality of PEMT.

This thesis outlined how ISDT, ISRF and DIS are interrelated to PEMT and how the research area should be investigated. The interrelations between ISRF, ISDT and DIS were identified. This has never been done before in scientific research. The basic artifacts of the PEMTs were defined by meta-requirements, and design product effectiveness hypotheses. Further benchmarking possibilities were discussed. One goal of this thesis was to provide guidance to both scientists and practitioners. This thesis outlined how ISDT, ISRF and DIS are interrelated to PEMT and how the research area should be investigated. The interrelations between ISRF, ISDT and DIS were identified. This has never been done before in scientific research.

The ISDT for PEMT built in this thesis demanded an in depth analysis of the project scope management knowledge area, an in depth analysis of literature. Several PEMTs are available in the markets. However, this thesis took only FiSMA methods under research. A comprehensive analysis of FiSMA models was done. This thesis defined PEM to consist of functional size measurement, delivery rate analysis, situation analysis, and reuse rate measurement. PEMT was defined to consist of five primary steps: (1) project size calculation, (2) software reuse impact to the effort, (3) delivery rate determination (4) situation analysis, and (5) benchmarking data. The ISDT meta-requirements, a part of

meta-design and design product effectiveness hypotheses were defined to support the literature review.

Benchmarking in PEM needs to be based on standardized project history database which enables comparison against other group of projects with similar characteristics. Benchmarking possibilities were discussed from the bases of FiSMA methods.

This thesis took the product aspects for the class of PEMT. Partial ISDT for PEMT was introduced by defining partial meta-requirements a part of meta-design and by design product effectiveness hypotheses. The results were derived from the literature review. The partial meta-requirements and design product effectiveness hypothesis were not validated.

The scope of this thesis was challenging because of the wide knowledge area of PEMT and the number and difficulty of the theories used. Partial ISDT for PEMT creation needs in depth exploration and definitions of PEMT processes. By building a partial ISDT for PEMT, this thesis raises a grate deal of future research possibilities.

REFERENCES

Albrecht A. J. “Measuring Application Development Productivity”, Proceeding of the Joint SHARE, GUIDE, and IBM Application Developments Symposium, p. 83-92, 1979

Basili V. “The Experimental Paradigm in Software Engineering”, In D. Rombach, V. Basili, R. Sleby, editors, Experimental Software Engineering Issues: Critical Assessment and Future Directives, p. 3-12, 1992

Band W. “Benchmark Your Performance for Continuous Improvement” Sales & Marketing Management in Canada, 1990, Vol. 31, No. 5, p. 36-38

Boehm B. W. “Software Engineering Economics”, Prentice-Hall, Englewood Cliffs, New Jersey: Prentice Hall PTR, 1981.

Ciborra, C.U., Lanzara, G.F., “Formative Contexts and Information Technology” Accounting, Management & Information Technologies, Vol. 4, No. 2, 1994, p. 61-86

Churchman, C., ”The Design of Inquiring Systems, Basic Books, New Your, 1971

Denning P. J. (1997) “A New Social Contract for Research” Communications of the ACM (40:2), February 1997, p. 132-134

FiSMA 2004. FiSMA Functional Size Measurement Method version 1.1 [online]. FiSMA Association [viitattu 29.8.2005]. Saatavilla www-osoitteessa <http://www.fisma.fi/pdf/FiSMA_FSM_Method_11.pdf>.

FiSMA 2001. Experience ND21 Situation Analysis Method [online]. FiSMA Association [viitattu 29.8.2005]. Saatavilla www-osoitteessa <http://www.fisma.fi/pdf/FiSMA_Situation_Analysis_Method_Nd21>.

FiSMA 2002. FiSMA Reuse Measurement Method, FiSMA RMM version 2002 [online]. FiSMA Association [viitattu 29.8.2005]. Saatavilla www-osoitteessa <http://www.fisma.fi/pdf/FiSMA_Reuse_Analysis_Method_10.pdf>.

Forselius, 2005, discussion 12.9.2005

Forselius, 2006, discussion 4.1.2006

Hall H., Paradice D., Courtney J. " Building a Theoretical Foundation for a Learning-Oriented Management System" Journal of Information Technology Theory and Application, Vol. 5, No 2, 2003 p. 63-85

Hevner A. R., March S. T., Park J., Sudha R. "Design Science in Information Systems Research" MIS Quarterly, Vol. 28, No. 1, March 2004, p. 75 – 105

ISBSG, 2004, "Practical Project Estimation 2nd Edition", ISBN 0957720114, International Software Benchmarking Standards Group Limited

Jones T.C. "Estimating Software Costs", New York: McGrawHill, 1998

Kasper, G. M. "A Theory of Decision Support System Design for User Calibration" Information Systems Research, Vol. 7, No. 2, 1996, p. 215-232.

Kemerer C.F. "An Empirical Validation of Software Cost Estimation Models", Communications of the ACM, 1987, Vol. 30, No. 5, p. 417-429.

Kraul, R., Streeter, L. "Coordination in Software Development" Communications of the ACM, Vol. 38, No. 3, 1995, p. 69-81

Kitchenham B. "Empirical Studies of Assumptions That Underline Software Cost-Estimation Models", Information and Software Technology, 1992, Vol. 34, No. 4, p. 211-218

Käkölä T., Dual Information Systems in Hyperknowledge Organizations. Ph.D. Dissertation. TUCS Dissertation, No. 2. Turku Centre for Computer Science. University of Turku, Department of Computer Science, 1996

Käkölä T., Taalas A. "Developing a Design Theory for Dual Change Management Information Systems", Manuscript accepted with revisions to Journal of Management Information Systems, 2005

Käkölä T., Koota K., Redesigning computer-Supported Work Processes with Dual Information System. Journal of Management Information Systems., Vol 16, No. 1. 1999, p. 87-119

Liimatainen J., "Building an Information System Design Theory for an Integrated Requirement Management and Release Management System" University of Jyväskylä, Department of Computer Science and Information Systems, 2005

Marciniak, J., 1994. Encyclopedia of Software Engineering, 518-524. New York, NY: John Wiley & Sons, 1994.

March S. T., Smith G. "Design and Natural Science Research on Information Technology" Decision Support Systems, December 1995, Vol. 15, No. 4, p. 251-266

Markus, M. L., Majchrzak, A. & Gasser, L. (2002) A Design Theory for Systems that Support Emergent Knowledge Processes. *MIS Quarterly,* 26**,** 179-212

Maxwell K., Forselius P. "Benchmarking Software-Development Productivity - Applied Research Results" IEEE Software, 2000, Vol. 17, No.1 p. 80-88

Miller J. "Measuring Projegress Through Benchmarking" CMA Magazine, 1992, Vol. 66, No. 5, p. 37

Nunamaker J., Chen M., Purdin T. D. M. "Systems Development in Information Systems Research" Journal of Management Information Systems, Vol. 7, No. 3, 1991, p. 89-106

Orlikowski, W. "The Duality of Technology: Rethinking the Concept of Technology in Organizations." Organization Science, Vol. 3, No. 3, 1992, p. 398-427

OTA ohjelmistoprojektin työmäärän arviointi, 2005, Software Technology Transfer Finland

PMBOK Guide, 2004, ISBN 1-930699-45-X, PMI Institute

PMBOK Guide, 1996, ISBN 1-880410-13-3, PMI Institute

Putnam L. H., Myers W. "Measures for Excellence, Reliable Software on Time, Within Budget", Yordon Press, Englewood Cliffs N.J., 1992

Premraj R., Twala B., Mair C., Forselius P. "Productivity of Software Projects by Business Sector: An Empirical Analysis of Trends", Bournemouth University, Empirical Software Engineering Research Group, 2004

Silver M. S., Markus M. L., Beath, C. M. "The information Technology Interaction Model: A Foundation for the MBA Core Course" MIS Quarterly, September 1995, Vol. 19, No. 3, p. 361-390

Simon H. A., "The Sciences of the Artificial" (3rd ed.), Cambridge, MA: MIT Press, 1996

Sharman P. "Benchmarking: Opportunity for Accountants" CMA Magazine, 1992, Vol. 66, No 6, p. 16-18

Shepetuk A. “Is Your Project Development Process A Tortoise or A Hare?” Management Review, 1991, Vol. 80, No. 3, p. 25-27

Sherer, S. “Using Risk Analysis to Manage Software Maintenance” Journal of Software Maintenance, 1997, Vol. 9, No. 6, p. 345-364.

Spendolini M.J. “The Benchmarking Book” Amerkican Management Association, New York: 1992

Stein E.W., Zwass V. ”Actualizing Organizational Memory within Information Systems” Information Systems Research, Vol. 6, No. 2, 1995, p. 85-117

Stutzke R. D. “Estimating Software-Intensive Systems Projects, Products, and Processes”, Paikka: Addison Wesley Professional, 2005

SWEBOK, 2004, ISBN 0-7695-2330-7, IEEE

Tsichritzis, D. “The Dynamics of Innovation in Beyond Calculation: The Next Fifty Years of Computing” Copernicus Books, New York, 1998, p. 259-265

van Aken J. E. ”Management Research Based on the Paradigm of the Design Sciences: The Quest for Field-Tested and Grounded Technological Rules” Journal of Management Studies, 41:2, March 2004, p. 219 – 246

Walls J. G., Widmeyer G. R., El Sawy O. “Building an Information System Design Theory for Vigilant EIS” Information Systems Research, Vol. 3, No. 1, 1992, p. 36 – 59

Walls J. G., Widmayrs G. R., El Sawy O. “Assessing Information System Design Theory in Perspective: How Useful was our 1992 Initial Rendition?” Journal of Information Technology Theory and Application, Vol. 6, No. 2, 2004, p. 44 – 58

Wieczorek I. “Improved Software Cost Estimation - A Robust and Interpretable Modelling Method and a Comprehensive Empirical Investigation”, Stuttgart: Fraunhofer IRB Verlang, 2001.

APPENDIX 1: CONCEPTS USED

PMBOK (2004) divides project management into nine sections: Project integration management, project scope management, project time management, project cost management, project quality management, project human resource management, project communication management, project risk management and project procurement management. These project management knowledge areas overlap.

Project management: "The application of *knowledge*, *skills*, *tools*, and *techniques** to meet the project *requirements*." (PMBOK, 2004, p. 368)

Estimate "An assessment of the likely quantitative result. Usually applied to project costs and durations and should always include some indication of accuracy (e.g., ± x percent). Usually used with a modifier (e.g., preliminary, conceptual, feasibility). Some application areas have specific modifiers that imply particular accuracy ranges (e.g., order-of-magnitude estimate, budget estimate, and definitive estimate in engineering and construction projects)." (PMBOK, 1996, p. 163)

7.1 Requirements in software estimation

Requirements: "Software project requirements can be divided into three subclasses: functional requirements, non-functional requirements and technical requirements" (ISBSG, 2005, p.4). On the other hand requirements are defined in PMBOK (2004, p.371) as: "A condition or capability that must be met or possessed by a *system*, *product*, *service*, *result*, or *component* to satisfy a *contract*, *standard*, *specification*, or other formally imposed *documents*. Requirements include the quantified and documented needs, wants, and expectations of the *sponsor*, *customer*, and other *stakeholders*."

Functional (user) requirements: "represent what user functions will be included in the software. Functional requirements are the business processes performed

by or supported by the software, (e.g. record and store ambient temperature) and include what functions the software must do. These requirements are part of the user(s)' responsibility to define. Functional requirements are the 'floor plan' for software. Functional size represents the size of the functional user requirements." (ISBSG 2004, p.4)

Nonfunctional requirements: "Nonfunctional requirements define the solution which functional requirements have identified. They are also known as constraints of quality requirements. They can be classified according to performance requirements, maintainability requirements, safety requirements, reliability requirements etc. "(SWEBOK, 2004, p. 2-2)

7.2 Project scope management

Scope planning "a scope management plan that documents how the project scope will be defined, verified, controlled, and how the work breakdown structure (WBS) will be created and defined." (PMBOK 2004, p.103)

Project scope management plan: "The *document* that describes how the *project scope* will be defined, developed, and verified and how the *work breakdown structure* will be created and defined, and that provides guidelines on how the *project scope* will be managed and controlled by the *project management team*. It is contained in or in a subsidiary plan of the *project management plan*. The project scope management plan can be informal and broadly framed, or formal and highly detailed, based on the needs of the *project*." (PMBOK 2004, p. 370)

Scope definition: "The *process* of developing a detailed *project scope statement* as the basis for future project decisions." (PMBOK 2004, p.375)

Project scope statement: "The narrative description of the *project scope*, including major *deliverables*, project *objectives*, project *assumptions*, project *constraints*, and a *statement of work*, that provides a documented basis for making future project decisions and for confirming or developing a common understanding of *project*

scope among the *stakeholders*. The definition of the *project scope* – what needs to be accomplished." (PMBOK 2004, p.370)

Effort: "The number of labour units required to complete a *schedule activity* or *work breakdown structure component*. Usually expressed as staff hours, staff days, or staff weeks. Contract with duration." (PMBOK 2004, p.360)

Stakeholders: "Person and *organizations* such as *customers*, *sponsors*, *performing organizations* and the public, that are actively involved in the *project*, or whose interests may be positively or negatively affected by execution or completion of the project. They may also exert influence over the project and its *deliverables*." (PMBOK 2004, p. 376)

Work breakdown structure: "A *deliverable*-oriented hierarchical *decomposition* of the *work* to be *executed* by the *project team* to accomplish the project *objectives* and create the required deliverables. It organizes and defines the total *scope* of the *project*. Each descending level represents an increasingly detailed definition of the *project work*. The WBS is decomposed into *work packages*. The deliverable orientation of the hierarchy includes both internal and external deliverables." (PMBOK 2004, p. 379)

Scope verification: "The *process* of *controlling* changes to the *project scope*." (PMBOK 2004, p.375)

Scope control: "The *process* of *controlling* changes to the *project scope*." while scope creep is defined as "Adding features and functionality (*project scope*) without addressing the effects on time, *costs*, and *resources*, or without *customer* approval." (PMBOK, 2004, p.375)

7.3 Project Quality Management

Project quality management: "Project Quality Management processes include all the activities of the performing organization that determine quality policies,

objectives, and responsibilities so that the project will satisfy the needs for which it was undertaken" (PMBOK, 2004, p.179)

7.4 Function point, functional size and functional size measurement

Functional size is defined by ISO/IEC/JTC1/SC7 Standard #14143-1 as "A size of the software derived by quantifying the Functional User Requirements." ISO defines FSM as "The process of measuring Functional Size". According to ISBSG (2004, p.91) Functional size represents the size of the subset of user requirements known as the Functional User Requirements (i.e., what functions the software must support), which excludes Quality and Technical Requirements.

7.5 Other inevitable definitions

ISBSG (2004, p. 95) defines project delivery rate as measures that rate at which a project delivers software functionality to the end user as a factor of the effort required to do so. It is defined as Project Work Effort, (measured in hours) over Functional Size of the delivered software, (measured in function points). It is expressed as Hours per Functional Size Unit.