A Framework for Effort Management in Software Projects

Topi Haapio
TietoEnator Telecom & Media
P.O. Box 1779, FI-70601 Kuopio, Finland
topi.haapio@tietoenator.com

Abstract

Objective. The objective of this paper is to provide a framework for effort management in software projects to increase effort estimation accuracy.

Method. We applied a multimethodological approach employing a case study and constructive research.

Results. Based on the case study on four earlier proposed frameworks related to effort management and three popular process maturity assessment models, we constructed a framework to manage effort in a software project in a proactive manner, yet fulfilling the requirements of the assessment models.

Conclusions. A project's software engineering process involves various functions which have an effect on the success of effort estimation. Two approaches, proactive and reactive, can be taken to manage effort in a software project. In this paper, a case study is conducted regarding both approaches. Based on the case study, we provide a new framework for effort management in software projects to increase effort estimation accuracy.

Keywords: effort management, framework, maturity model, project management, SPI

1. INTRODUCTION

It is crucial for both software suppliers and customers that the software project effort is estimated as accurately as possible to ensure adherence to projects budget and schedules, and the success of resource allocation [20]. Several formal models, methods and tools have been proposed for software project effort and cost estimation, although the informal expert judgment technique has remained as the most commonly employed estimation technique in practice [11]. However, despite the employed approach, the estimates have in many cases been unacceptably inaccurate. The reasons for inaccuracy include inadequate effort data collection of previous projects [5] and the light consideration on different activities involved with the project. The focus in effort estimation has been on software construction and project management, whilst neglecting other support activities related to the software projects [3][4][5].

In other words, the success of the effort estimation in terms of accuracy may depend not only on effort estimation but also on effort management, i.e. a high estimation error can be a result of insufficient control in a project [2]. Effort management is an organized process to estimate, collect, monitor and control, and analyze effort related to software projects and their activities. Software engineering literature (e.g. [15][16][20]) does focus on project management and effort, but the emphasis is on effort estimation and effort-related planning (e.g. scheduling) rather than on a total management of effort. Moreover, the effort management related functions are presented in the literature as parts of project management instead of presenting effort management as a coherent and an individual area of management. We believe a focus on effort management is advantageous, and a broader perspective on the topic is needed. In this paper, we propose a framework for effort management in software projects to be applied by project managers and effort analysts to increase control over the effort related to the software projects.

Effort can be managed by taking either a proactive or a reactive approach. The proactive approach attempts to consider the issues and act in advance (*ex ante*). Different frameworks to manage effort represent the proactive approach. Earlier proposed frameworks related to effort estimation or effort management include [1][3][21][22]. The reactive approach considers the issues afterwards (*ex post*). The process maturity assessment models represent the reactive approach. The most widely known and employed capability maturity models include ISO/IEC 15504 [8][9], CMM [18], and CMMI [19]. The capability maturity models are used to improve the related processes, and they set different effort-related requirements on software projects. Hence, effort management can also be the focus of the software process improvement (SPI) activity. SPI and the capability maturity model employment are under enormous interest of research, and numerous studies have been conducted applying the above mentioned models in the software industry. However, studies describing improvement activity regarding software project effort management context are not common.

This paper is structured as follows. Section 2 describes the research methodology. In Section 3, a case study is conducted on the existing approaches on effort management. Based on the analysis of the case study, a new framework is outlined in Section 4. The analysis in Section 5 is followed by a discussion in Section 6. A brief conclusion and suggestions for future work are given in Section 7.

2. RESEARCH METHODOLOGY

In this study, we applied a multimethodological research approach [14] employing two research methodologies influencing each other: case study [23] and constructive research [10]. Constructive research [7][10], also referred to as design science [6][13] or systems development [14], consists of two basic activities: building products and evaluating them [10]. Constructive research includes the development of new ideas and concepts, and construction of conceptual frameworks, new methods, or models [14], which result in research products: constructs, models, method, and instantiations [10][13]. In this research we construct an effort management framework, which represents a model. The research artifact is constructed on the basis of a case study. Case studies can also be used for evaluation [12]. The evaluation of the built product is based on user value or utility [10][13]. The evaluation process attempts to diminish the concern of unpractical research products [14] by applying suitable metrics and comparing the performance of artifacts for specific tasks [10][13]. The evaluation criteria of innovation for a model are applied in this research: framework's fidelity with a real world phenomenon, completeness, level of detail, robustness, internal consistency, and form and content. In other words, the research results need to be pragmatic in order to be utilized in the software industry, and they must provide significant improvement [10][13].

3. CASE STUDY

The case study is on four effort management related frameworks and three popular process maturity assessment models. The frameworks represent a proactive approach on the subject and the assessment models represent a reactive approach. The cases (frameworks and assessment models) were chosen to represent typical approaches, and based on their popularity (assessment models). By choosing the first releases of CMM and ISO/IEC 15504 and the latest release of CMMI we strive to examine the effort management related trajectory of the competing models, since CMMI has evolved from CMM and has been influenced by ISO/IEC 15504.

3.1. PROACTIVE APPROACH (1): EFFORT MANAGEMENT RELATED FRAMEWORKS

Case 1.1: Fairley, 1992. Although Fairley concentrates on presenting different software estimation techniques [1], he also proposes a framework for software estimation by describing different factors to be considered in preparing a software estimate. These factors can be categorized into product objectives, corporate assets, and project constraints, which continue to be valid factors. The employed software estimation technique should correspond with these factors to be effective. Hence, Fairley's framework is a cost estimation paradigm modeling the relationships between

these factors. Fairley considers effort management related issues through the various estimation techniques presented. Besides the most considered effort-related function, effort estimation, mentioned functions include effort collecting and analysis, and calibration of the estimation method. Additional considerations include effort repository and effort monitoring. Fairley considers work breakdown as an alternative approach to derive effort.

Case 1.2: Vesterinen, 1998. Vesterinen's paper on a framework and process for effort estimation [22] consists of a process description and a framework for effort estimation. Vesterinen's comprehensive description on effort estimation process includes, besides making the estimates at certain estimation points, storing the estimates, and reporting and analyzing the effort data. Vesterinen's perspective on managing effort is heavily one of the estimation of effort and in improving its process. In fact, the proposed framework is based on SEI's presentation on establishing a software measurement process [17], and consists of three main phases of planning, implementing, and improving. In planning, the scope objectives, issues and measures are identified, and procedures for measures, collection, analysis and feedback are defined. In implementing, effort data is collected and analyzed. In improving, the process is evolved further. Although the framework focuses on effort estimation, it includes several effort management related functions in order to manage the estimation process. However, the described functions are more of process functions to improve estimations rather than effort management functions. Vesterinen urges to question the effort-related process and its functions, and, when necessary, improve them.

Case 1.3: Tsoi, 1999. Tsoi proposed a framework, Software Development Management (SDM), in late-90's [21]. SDM, which acknowledges effort management, consists of two stages: an acquisition phase and an operation phase. The acquisition phase deals with effort planning related functions. These functions involve the estimation of development effort, establishing effort management policy, establishing effort repository, and identifying the critical development factors. The operation phase includes all functions during the actual software development phase. The effort repository is employed during the operation phase. The major functions include reviewing monitoring and evaluation metrics, monitoring of development effort, and evaluation of the collected information. Besides the various effort-related functions considered, resource allocation is also included in the SDM framework. However, as the framework covers the pre-project and project phases, the post-project functions, such as effort estimation method calibration, have been neglected. Moreover, SDM does not consider WBS or other activity-related functions.

Case 1.4: Haapio, 2006. We proposed a framework for improving software project effort management. The framework can be used for pinpointing areas needing the most attention in improving effort management, for which it provides a matrix. The framework identifies elements related to effort management and five phases (three project phases and a phase preceding and proceeding the project), and lists different effort-related and activity-related functions for each phase. A theoretical background is given for the key effort management functions and elements. Moreover, the framework acknowledges the significance of software project activities in effort management.

3.2. REACTIVE APPROACH (2): PROCESS MATURITY ASSESSMENT MODELS

Case 2.1: CMM, 1993. The Capability Maturity Model (CMM) [18] presentation is rather general and concise. The effort management related issues are considered first and foremost on CMM's maturity level 2 ('repeatable') in the software project planning key process area in which goal 1 requires software estimates to be documented for use in planning and tracking the software project, goal 2 requires software project activities and commitments to be planned and documented, and goal 3 requires affected groups and individuals to agree with their commitments related to the software project. The CMM model does not specify individual effort management related issues, or how to carry out the maturity requirements in practice. However, the descriptions refer to effort (cost), schedule, software activities, and resources. On the higher maturity levels, the requirements relate to effort management but from the perspective of the key process area.

Case 2.2: ISO/IEC 15504, 1993. The ISO/IEC 15504 assessment model [8][9], formerly known as SPICE, provides an extensive and detailed representation on the assessment of effort management related issues. However, effort is not isolated to an individual management area. Instead, effort is included into the project process category ('PRO') as one project resource parameter to be estimated together with size, cost, schedule, and resources [8]. The effort-related issues are considered throughout the project process category. The assessment documentation provides usable checklists to manage effort [9]. Therefore, the ISO/IEC 15504 model can be employed as a proactive tool in project management, effort management included. Again, effort-related issues are not assembled together but are pointed out to be considered in several areas. The issues include for example schedule, WBS, estimates, plan, project plans, progress status record and report, analysis results, measures, and project measures.

Case 2.3: CMMI, 2006. Elements of effort management are required by CMMI. Mostly, the effort management relates to the project management process area in CMMI. Like CMM, CMMI requires an organization's measurement repository, which is used to collect and make available measurement data on processes, e.g. effort and cost estimates, and the realized actual effort and costs, to analyze the measurement data. Moreover, the staged representation requires work to be arranged as work elements and their relationship to each other and to the end product, i.e. into a work breakdown structure to estimate the scope of the project, and to plan the project resources and to manage configurations (at the lowest level indicating maturity, 2) [19].

4. EFFORT MANAGEMENT FRAMEWORK

Based on the analysis of the case study, we propose a new framework for effort management which enables to manage software project effort in a proactive manner, yet fulfilling the requirements of the assessment models. The framework is largely based on our earlier proposed framework for improving effort management [3] in software projects, but is modified to assist managing the effort instead of improving the effort management.

4.1. FRAMEWORK ELEMENTS

The effort management framework includes several elements in relationships with each other. These elements include:

- Framework perspectives, which include effort management lifecycle, effort-related functions and effort distribution on particular activities or activity sets. These perspectives are described in Sect. 4.2.
- Effort management lifecycle, which include a prior-project phase, three phases of the project's lifecycle (pre-project, project, and post-project phase), and a phase after the project's existence. The lifecycle is described in Sect. 4.2.1.
- Software project activities. A software project consists of activities, which can be arranged into a work breakdown structure (WBS). WBS is a particular tree-hierarchy in which project activities are categorized [16]. A generic organization-proprietary WBS as a part of project-specific WBS enables comparison between projects within a company.
- Activity sets. A software project activity set is a specific set of project activities. For example, the activity set of testing can include unit testing, integration testing, system testing, and customer acceptance testing [5].
- Software project effort. Effort of a software project can be generally defined as the number of person days (or hours or months) consumed by the project.
- Registration entities. The software project effort is registered on the work time registry system's registration entities by the project group.
- Effort repository. Effort repository contains analyzed effort data, which is suitably organized. Effort repository is utilized in effort management functions, e.g. effort estimation and effort estimation method calibration. Preferably, the work time registry system and the effort repository are linked with each other with consistent activities, which enables an automatic interface between these two systems for an efficient and easy employment of software project activity information.

- Effort management functions. These functions include both effort-related (e.g. effort estimation) and activity-related (e.g. WBS establishment) functions. The functions are described in Sect. 4.2.2.
- Effort analyst. Effort analyst is an expert, e.g. a project manager, whose task is to manage project effort throughout the project [5], e.g. analyze the project effort after the project has been completed. The effort analyst also designs the appropriate WBS and the registration entities in the work time registry system, and operates with the effort repository.

4.2. EFFORT MANAGEMENT PERSPECTIVES

The framework includes three perspectives into effort management: Effort management lifecycle with different phases, necessary functions during those phases, and software project activities (Fig. 1). These perspectives influence on each other.

Figure 1: The three perspectives of the effort management framework

4.2.1. Effort Management Lifecycle

The effort management lifecycle can be divided into continuous five phases. The three middle phases (pre-project, project, and post-project) refer to software projects lifecycle (phases II-IV highlighted with gray in Fig. 1). However, the effort management lifecycle consists also of two phases preceding and following the software project's phases.

During the pre-project planning, the project is planned and set up. Project planning includes the project activity planning and effort estimation sub-functions. Effort is initially estimated with the method in use with project information supplied by the customer. At this point, the different activities concerning the project are also planned. These activities include the activities related to actual software construction, project management, and other project-related activities. The planned activities are created during the project setup into the work time registry system as registration entities for effort registrations during the project execution. These entities are organized into particular activity sets as a work breakdown structure.

During the actual project phase the project effort is collected, monitored, and re-estimated. Effort is collected during the project execution to monitor its realization. If effort realizes different than planned, the reasons are explored and necessary actions are taken. Furthermore, effort is collected for re-estimations with adjusting project-specific data, and for effort analysis after the project has been completed. The effort collection includes effort registration by the project group members on registration entities in the work time registry system. The new project activities and new sets of activities require an adoption period before effort can be registered on the new entities.

During project closure, the post-project phase, the delivered project is analyzed, and a project final report is drawn. As a sub-function, effort analysis is conducted to produce input for a thorough, usually qualitative, project post-mortem analysis. In effort analysis the realized effort of project activities are compared with the estimated, and the reasons for the accuracy or inaccuracy are analyzed and explained. Effort analysis produces effort information for improving and calibrating the estimation method, thus improving the software process. Moreover, if new significant activities are identified, they are recorded for activity planning of future projects.

Effort management, however, starts before a project exists and continues also afterwards. These effort management functions include the establishment of a general, project-independent WBS. Also, the effort estimation method (tool) for the software project is either acquired or a proprietary estimation method is constructed, which after the method is initially prepared for estimations, for example with effort or cost driver calibration. The method is adjusted after the project has been post-mortem analyzed. The analyzed effort data is stored in the effort repository. The effort repository requires preparation and adjusting actions during the first and last phase, respectively.

4.2.2. Effort Management Functions and the Role of Software Project Activities

Besides effort estimation, a software project involves various other effort-related functions throughout the project's lifecycle. Some effort-related functions exist also before and after the project. The effort-related functions have an influence on effort estimation. Moreover, the software project activities are closely related to effort, since the implementation of an activity requires certain amount of effort. Therefore, it is advantageous to consider also the activity-related functions within the effort management framework. The effort-related and activity-related functions are listed in Table 1 and specified below.

Table 1: The effort-related and activity-related functions of the effort management framework

Framework Phase	Effort-Related Functions	Activity-Related Functions
Prior to project's existence	Effort estimation method preparation, Effort repository preparation	WBS establishment
Project: Pre-project	Effort estimation	Registration entity creation
Project: Project	Effort collection, Effort monitoring, Effort re-estimation	New registration entity adoption, Effort registration on registration entities
Project: Post-project	Effort analysis	Software project activity analysis
After project's existence	Effort repository updating, Effort estimation method adjustment	WBS adjustment

Effort-Related Functions

- Effort estimation method preparation. Before effort can be estimated, a suitable method and application must be chosen, and calibrated with effort history data to enable analogies.
- Effort repository preparation. The effort repository contains effort data of the completed projects, and is populated with effort data resulting from effort analyses. The repository can be used for calibrating the employed effort estimation method's weights. The effort analyst

collects project activities and relating effort, and reorganizes the activities consistent into the repository as specific reasonable activity sets [5].

- Effort estimation. Effort is initially estimated for a project during the pre-project bidding and tender phase. Effort is estimated with the chosen method(s) and application(s).
- Effort collection. Effort is collected throughout the project duration. Effort data can be imported from the registration entities of the work time registry system into the effort repository. The collected effort data is used for monitoring, re-estimation, analysis, and calibration purposes.
- Effort monitoring. The realization of effort is monitored throughout the project duration. If effort realizes different than planned, the reasons are explored and suitable measures are taken.
- Effort re-estimation. Effort is repeatedly re-estimated during project whenever new project-related effort data is available or the project situation has changed. Therefore, effort estimation is not a single event.
- Effort analysis. Effort is analyzed after the project has been completed, during the post-project phase, as a part of post-mortem analysis. Effort analysis produces effort information for project reports. Moreover, based on the analysis, it can be determined whether a particular estimation model emphasizes its effort factors and drivers correctly [5].
- Effort repository updating. The effort repository is updated with essential effort information (identified effort data and new activities) as the project has ended and its effort has been analyzed [5].
- Effort estimation method adjustment. The employed effort estimation application is adjusted when analyzed effort information is available, for example by calibrating the application's weights used for effort derivation [5].

Activity-Related Functions

- WBS establishment. A WBS is established for each new software project. The established WBS is used for effort estimation and for creating the registration entities in the work time registry system. Each custom software development project requires an individual WBS. However, these individual activities concern mainly the software and its construction. Most activities related to project management and to the non-construction activities (general project activities) occur in most projects. Therefore, parts of the WBS are generic, and these project-independent parts are established before a project exists.
- Registration entity creation. The registration entities are created into the work time registry system much according to the established WBS. The project group members feed their work time (effort) information on the entities.
- New registration entity adoption. During project execution the project group registers effort on the activity entities that were created for the project in the work time registry system. The effort registration on old, familiar activity entities begins immediately, and effort is usually registered on the correct activity entity. The new project activities and new sets of activities, however, require an adoption period before effort can be registered on the new entities. At worst this period can last whole project execution which results in skewed effort data as effort is registered on wrong activities or is not registered at all. The correct registration of effort is essential for effort monitoring and effort re-estimations [4].
- Effort registration on registration entities. Effort is registered on the registration entities by project group members.
- Software project activity analysis. As a part of project's post-mortem analysis, the planned project activities are analyzed in relation to the realized activities whether some activity did not realize as planned in the project, or occurred in the project unpredictable. If new activities are identified, they are described in the final report with relating effort proportions, and, if found significant, complemented in the effort repository and in corresponding activity set for further utilization in WBS and effort estimations [5].
- WBS adjustment. After the project is completed and its activity analysis is performed, the generic part of the employed WBS is adjusted by adding new generic activities into the WBS and removing the out-dated activities.

5. ANALYSIS

This paper introduces a framework for effort management in software projects. Since a framework refers to a model artifact in constructive research, the criteria metrics for the evaluation of a model are applied: fidelity with a real world phenomenon, completeness, level of detail, robustness, and internal consistency [13], and form and content [10]. A model (here; the framework) is an abstraction of the real world, and completeness of the model in relation to the reality cannot be demanded [10]. Nevertheless, these criteria were the driving forces in constructing the proposed framework. Besides completeness and a suitable level of detail, robustness was a key premise for the framework. A starting point for our research was the earlier fragmented presentations of effort management. The criterion on internal consistency is a natural requirement from the research point of view (c.f. [10]). To support the criterion for form and content, special consideration was given on the presentation to support communication and diminish misunderstanding.

The constructed innovation must provide significant improvement [10][13]. The proposed effort management framework was constructed by developing an existing framework for improving effort management in software projects [3] further. Moreover, the framework is based on the case study analysis on three other frameworks ([1][21][22]) and three process maturity assessment models ([8][18][19]). A comparison between the framework proposed in this paper and the existing frameworks and capability maturity models in terms of the three perspectives on effort management implies that some major elements related to effort management seem to have been previously neglected (Table 2):

- A lifecycle related to effort management is identified in our framework, and two other frameworks ([3][21]). However, the SDM framework [21] is limited mostly to the pre-project and project phases whereas our framework includes phases before and after these two phases. The lifecycle the assessment models refer to is the process to be evaluated.
- All examined frameworks and assessment models identify several effort-related functions. The frameworks, however, concentrate mostly on the effort estimation activity whereas our framework identifies a larger range of effort-related functions concerning a software project.
- The role of software project activities is acknowledged in all but two frameworks ([21][22]). However, effort distribution is not included in the other frameworks as a key perspective into effort management whereas the activities and sets of activities are a key perspective for our framework. Only our framework and the framework it is based on [3] include activity-related functions. The assessment models identify activity-related functions, but only ISO/IEC 15504 [8] specifies them in more details.

The process maturity assessment models do not provide direct tools to improve effort management rather than requirements for the software engineering process to consider effort management, i.e. the models tell 'what' to do (or should be done) rather than 'how' it should be done. The assessment models are focused on the improvement of the technical processes of software development rather than project management, in which effort management is commonly included in the assessment models.

Our framework aims to improve the accuracy of effort estimations. Therefore, the effort-related functions which are influenced by effort but which do not (significantly) influence effort are delimited from the framework. An example of such a function is resource allocation, which has an influence on the effort estimation, since resources are not identical, i.e. other members are more experienced than others and thus require less effort to complete a task. However, we consider this influence to be rather insignificant, since project groups are usually heterogenic, i.e. the group consists of both more- and less-experienced members.

The effort management framework has been employed in software development projects at TietoEnator Telecom & Media, the largest business area within TietoEnator Corporation. Not only has the framework increased the project managers' knowledge on effort management related issues, but also improved the adherence to the effort management related process and functions. According to the framework, we have distinguished five phases and applied the effort-related and activity-related functions related to each phase. For effort management purposes and to

Table 2: A comparison on three effort management perspectives

	Identifies Effort Management Lifecycle Phases (#)	Specifies Effort Management Functions: Effort- and Activity-Related	Acknowledges Software Project Activities
Effort Management Framework	5	Both	Yes
Fairley, 1992 [1]	None	Effort-related	Yes
Haapio, 2006 [3]	5	Both	Yes
Tsoi, 1999 [21]	2 (emphasizes forepart)	Effort-related	No
Vesterinen, 1998 [22]	3 (process improvement related)	Effort-related (process improvement related)	No
CMM, 1993 [18]	None	Both (identifies, not specifies)	Yes (does not specify)
ISO/IEC 15504, 1993 [8]	None	Both	Yes
CMMI, 2006 [19]	None	Both (identifies, not specifies)	Yes (does not specify)

increase the project comparability we have divided effort in three major project activity sets: software construction, project management, and non-construction activities [4], and established a WBS with a generic part concerning the non-construction activities to be employed with software projects.

6. DISCUSSION

This study complements the existing research on software process improvement in particular by focusing on a special case of effort management improvement in software projects. Although there has been a vast interest in the concepts of SPI and effort in software projects, effort management frameworks are in a short supply. A rigorous view on the software project effort in the form of effort management framework combines the traditionally separate elements of managing the project effort to a coherent presentation where the elements influence each other. We believe that the proposed framework can be employed within the software industry. The framework is independent from the organization and is relatively general and likely applicable in software development by project managers and effort analysts. The practitioners can apply the results of this research to derive own adapted version of the effort management framework with a suitable effort distribution categorization. A general breakdown of the activities is advantageous for comparing the projects to each other.

7. CONCLUSIONS AND FUTURE WORK

This paper proposed a framework for effort management in software projects. The effort management framework assists in a more complete management of effort in a software project compared to other frameworks we have analyzed in this paper's case study. The different effort-related functions, which can be found fragmented in the software engineering literature and in the capability maturity models, are assembled as a coherent presentation in the framework. In addition, this research supplements into the research of effort distribution by considering not just construction or project management but also the rest of the software project activities which are frequently ignored both in research and the effort management in software industry. Further, a follow-up case study on projects employing the proposed framework is endeavored to concretize the benefits.

Bibliography

- [1] Fairley R. (1992) Recent Advances in Software Estimation Techniques. *Proceedings of 14th International Conference on Software Engineering*, Melbourne, Australia, 11-15 May, pp. 382–391, ACM Press.
- [2] Grimstad S. and Jørgensen M. (2006) A Framework for the Analysis of Software Cost Estimation Accuracy. *Proceedings of the ISESE'06*, Rio de Janeiro, Brazil, 21-22 September, pp. 58–65, IEEE.
- [3] Haapio T. (2006) A Framework for Improving Effort Management in Software Projects. *Proceedings of the European Systems & Software Process Improvement and Innovation (EuroSPI2006)*, Joensuu, Finland, 11-13 October, International Proceedings Series 6, pp. 10.1–10, University of Joensuu.
- [4] Haapio T. and Ahonen J. (2006) A Case Study on the Success of Introducing General Non-construction Activities for Project Management and Planning Improvement. *Proceedings of the 7th International Conference on Product-Focused Software Process Improvement (PROFES 2006)*, Amsterdam, The Netherlands, 12-14 June, LNCS 4034, pp. 151–165, Springer-Verlag.
- [5] Haapio T. and Eerola A. (2006) Post-Project Effort Analysis Method. *Proceedings of the 1st Iberic Conference on Information Systems and Technologies (CISTI)*, Ofir, Portugal, 21-23 June, Vol. I (Organizational Models and Information Systems), pp. 3–19, Microsoft.
- [6] Hevner A., March S., Park J. and Ram S. (2004) Design Science in Information Systems Research. *MIS Quarterly*, **28**, 75–105.
- [7] Iivari J. (1991) A Paradigmatic Analysis of Contemporary Schools of IS Development. *European Journal of Information Systems*, **1**, 249–272.
- [8] ISO/IEC TR2 15504 (1993) Software Process Assessment : Part 2 : A model for process management, Version 1.00. ISO, Geneva, Switzerland.
- [9] ISO/IEC TR2 15504 (1993) Software Process Assessment : Part 5 : Construction, selection and use of assessment instruments and tools, Version 1.00. ISO, Geneva, Switzerland.
- [10] Järvinen P. (2001) *On Research Methods*. Opinpajan kirja, Tampere, Finland.
- [11] Jørgensen M. (2004) A review of studies on expert estimation of software development effort. *Journal of Systems and Software*, **70**, 37–60.
- [12] Kitchenham B., Pickard L. and Pfleeger S. (1995) Case Studies for Method and Tool Evaluation. *IEEE Software*, **12**, 52–62.
- [13] March S. and Smith G. (1995) Design and Natural Science Research on Information Technology. *Decision Support Systems*, **15**, 251–266.
- [14] Nunamaker J., Chen M. and Purdin T. (1991) Systems Development in Information Systems Research. *Journal of Management Information Systems*, **7**, 89–106.
- [15] Pressman R. (2005) *Software Engineering: A Practitioners Approach*. 6th edn. McGraw-Hill, New York.
- [16] Royce W. (1998) *Software Project Management: A Unified Framework*. Addison-Wesley, Reading, MA.
- [17] SEI-93-TR-016 (1993) Establishing a Software Measurement Process. Software Engineering Institute, Carnegie Mellon University, Pittsburgh.
- [18] CMU/SEI-93-TR-024 (1993) Capability Maturity Model for Software, Version 1.1. Software Engineering Institute, Carnegie Mellon University, Pittsburgh.
- [19] CMU/SEI-2006-TR-008 (2006) CMMI for Development, Version 1.2, CMMI-DEV, V1.2. Software Engineering Institute, Carnegie Mellon University, Pittsburgh.
- [20] Sommerville I. (2001) *Software Engineering*. 6th ed., Pearson Education, Harlow, UK.
- [21] Tsoi H. (1999) A Framework for Management Software project Development. *Proceedings of the 1999 ACM symposium on Applied computing (SAC'99)*, San Antonio, Texas, USA, 28 February-2 March, pp. 593–597, ACM Press.
- [22] Vesterinen P. (1998) A framework and process for effort estimation. *Proceedings of the 9th European Software Control and Metrics Conference*, Rome, Italy, 27-29 May. In: Vesterinen P. (2000). On effort estimation in software projects. Ph.D. Thesis. Acta Universitatis Lappeenrantaensis 96. Lappeenranta University of Technology, Lappeenranta, Finland.
- [23] Yin R. (1994) *Case Study Research: Design and Methods*. 2nd ed., Sage Publications, Thousand Oaks.