ISBN 3-901351-84-1
Verlag der Technischen Universität Graz
www.FTI.TUGraz.at/Verlag

EuroSPI’2003

EuroSPI’2003

European Software Process Improvement

10.-12.12.2003, University of Music and Dramatic Arts, Graz, Austria

http://www.eurospi.net

EuroSPI 2003 Proceedings

Partnership

ASQ, http://www.asq.org
ASQF, http://www.asqf.de
DELTA, http://www.delta.dk
ISCN, http://www.iscn.com
QINETIQ, http://www.qinetiq.com
SINTEF, http://www.sintef.no
STTF, http://www.sttf.fi

Supporters

APS, http://www.aps.tu-graz.ac.at
KUG, http://www.kug.ac.at

EuroSPI 2003
Proceedings

Proceedings

The papers in this book comprise the proceedings of the EuroSPI 2003 conference. They reflect the authors' opinions and, in the interests of timely dissemination, are published as presented and without change.

Their inclusion in this publication does not necessarily constitute endorsement by EuroSPI and the publisher.

EuroSPI

EuroSPI is a partnership of large Scandinavian research companies and experience networks (SINTEF, DELTA,STTF), QinetiQ as Europe's largest research center, the ASQF as a large German quality association, the American Society for Quality, and ISCN as the co-ordinating partner.

EuroSPI conferences present and discuss practical results from improvement projects in industry, focussing on the benefits gained and the criteria for success. Leading European industry are contributing to and participating in this event. This year's event is the 10th of a series of conferences to which countries across Europe and from the rest of the world contributed their lessons learned and shared their knowledge to reach the next higher level of software management professionalism.

Publisher

ISBN 3-901351-84-1

Verlag der Technischen Universität Graz

www.FTI.TUGraz.at/Verlag

Programme Committee

Micheal Mac an Airchinnigh, Trinity College & ISCN, IRL

Gualtiero Bazzana, Onion Technologies, It

Miklos Biro, Budapest University of Economic Sciences and Public Administration, HU

Mads Christiansen, Delta, DK

Howard Duncan, Dublin City University, IRL

Taz Daughtrey, James Madison University, USA and Editor, SOFTWARE QUALITY

PROFESSIONAL (American Society for Quality). USA

John Elliott, DERA, UK

Tim Hind, AXA-Sunlife UK

Bernd Hindel, ASQF, D

Theresa Hunt, Programs Chair of the ASQ Software Division, USA

Ioana Ene, Onion Technologies, It

Jorn Johansen, Delta, DK

Carsten Jorgensen, Delta, DK

Karl Heinz Kautz, Copenhagen Business School, DK

Pasi Kuvaja, University of Oulu, Finland

Patricia McQuaid, Americas Program Committee Chair for the Second (2000) and Third (2005) World Congresses on Software Quality, USA

Nils Brede Moe, Sintef, No

Messnarz Richard, ISCN IRL & ISCN A

Risto Nevalainen, STTF, Fin

Carsten Nøkleby, Grundfos A/S

Poul Grav Petersen, Novo A/S

Dag Sjoberg, University of Oslo, No

Tor Stalhane, Norwegian University of Science and Technology, No

David Teague, British Telecom, UK

Timo Varkoi, Pori School of Technology, Fin

Yingxu Wang, University of Calgary, Canada

EuroSPI Board Members

ASQ, http://www.asq.org

ASQF, http://www.asqf.de

DELTA, http://www.delta.dk

ISCN, http://www.iscn.com

QINETIQ, http://www.qinetiq.com

SINTEF, http://www.sintef.no

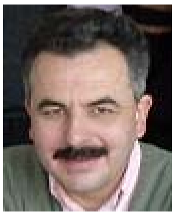
STTF, http://www.sttf.fi

Editors of the Proceedings

Dr Richard Messnarz, ISCN LTD

Katja Jaritz, ISCN GesmbH

Welcome Address by Dr Richard Messnarz



EuroSPI is a partnership of large Scandinavian research companies and experience networks (SINTEF, DELTA,STTF), QinetiQ as one of Europe's largest research centers, the ASQF as a large German quality association, the American Society for Quality, and ISCN as the coordinating partner. As the chairman of EuroSPI I welcome you to the EuroSPI 2003 conference and hope that you will benefit from it by exchanging experiences which are fruitful for your own organisation.

EuroSPI conferences present and discuss practical results from improvement projects in industry, focussing on the benefits gained and the criteria for success. Leading European industry are contributing to and participating in this event. This year's event is the 10th of a series of conferences to which countries across Europe and from the rest of the world contributed their lessons learned and shared their knowledge to reach the next higher level of software management professionalism.

Welcome Address by Mag. Andrea Fenz

APS is the National Contact Point for the Information Society Technologies (IST) program of the European Union. APS is also the regional Innovation Relay Centre (IRC) network partner for Southern Austria and enables transnational cooperation and knowledge transfer between Styria (region of Austria with Graz as its capital) and other European regions. The foundation of ISCN (responsible for the management of the EuroSPI partnership) was one of the most successful EU mobility and expert transfer projects by APS in 1993. Mag. Andrea Fenz, managing director of APS, welcomes all attendees and wishes a successful conference and fruitful technology transfer opportunities.

Contents

Session I – SPI and Engineering

Session II – SPI and Assessment Experiences

Session III – SPI and eWork and eSCM

Session IV – SPI and Risk Management

Session V – SPI and Analysis

Session VI – SPI and Improvement Implementation

Session VII – SPI and Analysis/Measurement

Session VIII – SPI and New Management Paradigms

Session IX – SPI and Cultural Factors

Session X – SPI and Improvement Implementation

A Situational Method for Requirements Engineering in E-business Projects

Ir. Alexander Smeitz, Ir. Erwin Folmer

Abstract

Situational method engineering allows methods to be flexible, adaptable, and practical. It is based on the assumption that a method can be created for each specific situation. For this, a situation characterisation is performed, and based on this characterisation method fragments are selected. A complete set of method fragments forms the situational method. This paper presents a situational method tailored for requirements engineering in e-business projects, as e-business differs from traditional software engineering, which influences the requirements engineering process. A so-called selection matrix is introduced that facilitates the construction of the method.

Keywords

Requirements Engineering, Software Engineering, E-Business, Software Process Improvements, Method.

1 Introduction

Research studies support the fact that requirements play a major role in project failure [1]. Over the last few years, the problems that arise with requirements have been increasingly recognised and requirements engineering has become an important phase in the system and software development process. However, most requirements engineering methods are focused on requirements engineering for traditional system and software development processes, i.e. development for a single customer or for customers with similar interests. System development for e-business is relatively new and, so it seems, more complex than traditional system development. Dealing with one customer is difficult; dealing with two, three or even more customers is more difficult, especially when interests are different and keeping customers aligned with each other is of key importance. The same is true for requirements engineering in this e-business context. This could become a cause for the failure of many e-business projects. The current offering of requirements engineering methods did not fulfil the need for TNO E-Business in e-business projects.

Therefore, TNO E-Business started research on the development of a new or adapted requirements engineering method suited for e-business projects. Flexibility and adaptability were stated as important requisites of the new method. Projects undertaken by TNO E-business in the recent past proved that the new method should also be highly practical. Situational method engineering addresses these aspects. It has the philosophy that information system development projects should strive for controlled flexibility, being the balance between rigid general-purpose methods and ad-hoc, flexible development [2].

2 Characteristics of e-business

The development of e-business information systems has become a real challenge compared with traditional system development. This is equally true for the requirements engineering part of system development. Certain aspects can be defined that are typical for, or especially important for, e-business projects. Extensive literature search indicated that there are five main aspects for e-business, best described in [3].

1. Greater number and diversity of stakeholders*.* E-business typically involves multiple organisations or departments; it concerns coupling and integration issues. This implies a greater number of stakeholders. The various organisations have evolved and developed in a different way, are structured differently, and have other core businesses. This means they have diverse interests, expectations and needs regarding the system(s) to be developed. This heterogeneity also implies that trust is a major issue; business partners need to be able to trust each other, as well as the systems and the infrastructure exploited for establishing communication. This results in all sorts of potential problems: conflicts, politics and misunderstandings, to name a few.

2. High level of innovation*.* E-business projects involve the innovative use of certain existing aspects of system development. The aspects themselves are not necessarily innovative; it is the combination of aspects which are used that is innovative. This means that the level of innovation is relatively high.

3. High degree of dependability. It is imperative to the (commercial) success of an e-business project that the systems are dependable. Dependability involves security (i.e. authentication, confidentiality, non-repudiation, and integrity), reliability (i.e. continuity of service), availability (i.e. readiness for usage), and safety (i.e. non-occurrence of catastrophic events that threaten human life, health, and the environment). A dependable system contributes toward raising and assuring trust in business relationships and services.

4. Changed environment. Due to societal influences and the possibilities of ICT, businesses have less control over their environment that has changed dramatically. As the barriers between traditional business columns have dissolved, businesses are no longer likely to have total control over the systems and networks upon which their e-business applications depend. Relationships with stakeholders outside the company have perhaps become more important than they ever were before.

5. Limited time-to-market. Ever since the early 1990s, Internet and web technology have been developing rapidly. It is unclear when or whether this will come to an end, but at this point in time it is considered as an aspect of e-business [3, 4, 5]. As with traditional system development, it is important to shorten the time-to-market, because of the stiff competition between companies. In e-business, however, this is even more important, not in the least as the result may be superseded when the project ends.

These five aspects of e-business have a great impact on the requirements engineering process, and therefore played an important part in the development of the situational method.

3 Situational Method Engineering

Various field studies show that situation-dependency is particularly important during the first stages of IS development, including requirements engineering [2]. In fact, many development projects fail, because requirements analysis is not performed in conjunction with an assessment of the project situation or the project goals. As a result, a situational method, which is an information systems engineering method tailored and tuned to a particular situation [6], is deemed needed.

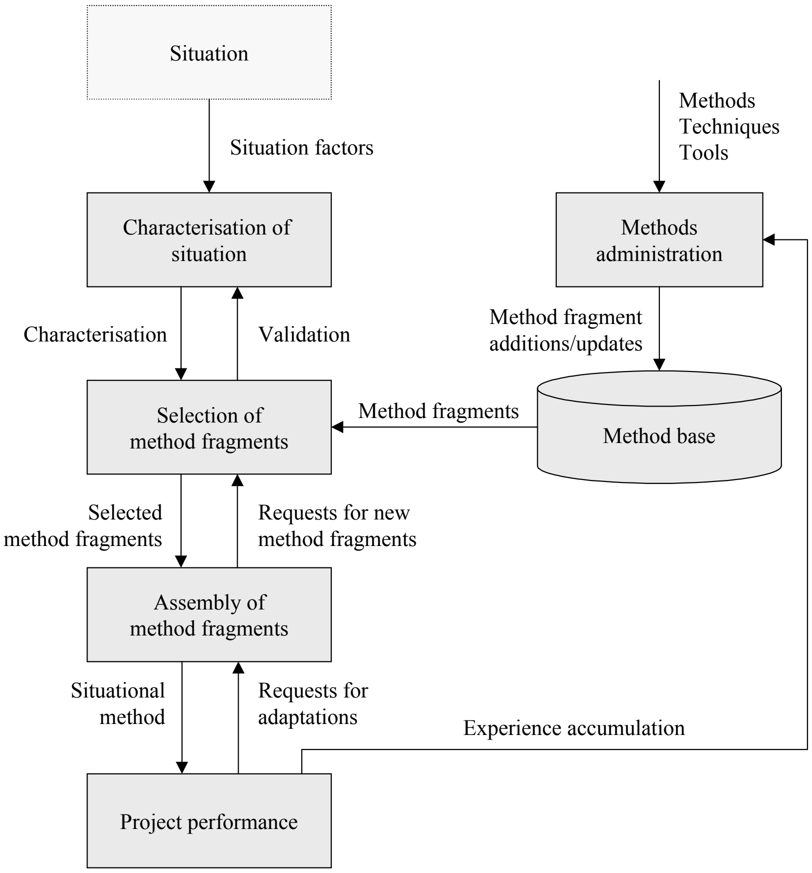


Figure 1 – The process of situational method engineering

To engineer such a situational method, several steps need to be carried out. Firstly, a characterisation of the situation is made in which the method will be applied. This characterisation is input to a selection process, where existing techniques and parts of existing requirements engineering methods are selected. Such parts, called method fragments, are stored in a method base. A method fragment is defined as a description of an information systems engineering method, or any coherent part thereof, whereas a method base is a structured collection, which stores method fragments using a conceptual model containing all types of method fragments, their relationships, properties, and constraints [6]. The unrelated method fragments are then assembled into a situational method, using assembly guidelines to ensure internal consistency and completeness. Figure 1 depicts the generic process of configuring a situational method.

This theory is a way to arrange the information system development process in its entirety, and needs to be modified to be useful in the requirements engineering process. The result of this modification is described in the next paragraph.

4 The situational method

This paragraph describes the situational method for requirements engineering in e-business projects which we suggest.

4.1 Situation factors

The characterisation of the situation in which the method will be applied (the project characterisation) is performed with the use of situation factors, also known as contingency factors. Situation factors are circumstances of the project influencing in some way the selection or construction of an approach to system development [7].

Using an extensive enumeration in [8], a set of 19 situation factors has been selected for this initial research. Because the method will be a requirements engineering method specific for e-business projects, the emphasis in the selection procedure has been put on situation factors typical of this context. This is achieved by deriving the situation factors from the aspects that distinguishes e-business projects from other projects, which were discussed in paragraph 2.

A distinction is made between positive situation factors, and negative situation factors. Positive situation factors contribute to the success of the project when valued high; negative situation factors contribute to the success of the project when valued low.

4.2 Method fragments

Once the project characterisation is finished, the method fragments need to be selected, and assembled into a new situational method. The method fragments used in this model are process fragments, which are descriptions of activities to be carried out within a method [6]. In requirements engineering, we define four activities: data collection, data analysis, requirements generation, and requirements validation. However, these activities are too generic to be included as method fragments. The techniques that are used in these activities however, are not generic. Several properties of these techniques provide means to discriminate, which allows them to be method fragments.

4.3 The selection process

The central idea of the selection process is that for each value of each situation factor a method fragment may be applicable. For instance, when degree of resistance is valued high, workshops can be used. These heuristics are incorporated in a selection matrix (Figure 2), which is used to facilitate the method fragment selection. Here, for each method fragment, a value in the upper left corner of a matrix cell denotes the applicability of the method fragment for the corresponding value of the situation factor. The steps in the selection process are:

1. Characterising the project. The characterisation of the project is done by valuing the 19 situation factors. The possible values are *high, medium,* and *low*.

2. Determining the influence of situation factors. Situation factors may have the same value, but a different degree of influence on the project. To facilitate this, a second property of situation factors is included: *influence*. The possible values of influence are also *high*, *medium*, and *low*.

3. Weighting of the method fragments. When the values of the method fragment and situation factor correspond (and thus the method fragment is applicable), the method fragment receives a weighting score. This enables a better ranking of method fragments. The weighting is performed using a simple weighting scheme. The values of the situation factor and its influence are combined to yield a numeric-weighting factor, which is assigned to the method fragment. This process is repeated for all situation factors.

4. Calculating the total of the weighting factors. When the weighting process is completed for all situation factors, the weighting factors are added up to yield a total score for the method fragment. This is done for all method fragments, which creates a ranking based on applicability and importance.

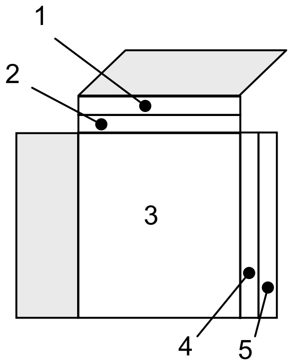
5. Selecting the method fragments. For each phase, the total scores for the method fragments are compared. The method fragment with the highest relative score is the most suitable fragment in the current situation, for the concerning phase. If this method fragment cannot be applied for some reason or the other, the next fragment in ranking can be selected, and so forth.

6. Constructing the method. The method is constructed by placing the method fragments, selected for each phase, in the right order. Miscellaneous method fragments are optional. The (combination of) method fragments are checked for relevancy before completing the selection process.

After these steps, the situational requirements engineering method can be used for the specific project situation. An example of how to use the selection matrix is given in Figure 2.

		Clarity project goal (+)	Degree of standardisation (+)	Project conformity (+)	Technical possibilities (+)	Contractual agreements (-)	Degree of innovation (-)	Heterogeneity of stakeholders (-)	System complexity (-)	Capability project team (+)	Management commitment (+)	Strategic importance (+)	Stability environment (+)	Conflict potential (-)	Degree of resistance (-)	Geographical distribution (-)	Organisational impact (-)	Scarcity resources (-)	Scarcity time (-)	Scarcity time (stakeholders) (-)		
	Value:																					
	Influence:																				Total	Rank
DC1.	Brainstorming	L						H		M				H				H	H	M		
DC2.	Literature study				L					L	L	L						H	L	H		
DC3.	Workshops							H		M		H		H	H		H	M	M	L		
DC4.	Contextual inquiry							H		H					H			M	M	H		
DC5.	Interviews									M						H		H	M	M		
DC6.	Context-free questions	L		L				H		M					H			H	H	H		
DC7.	Ethnographic study									H					H			L	L	M		
DC8.	Questionnaires	H	H							M						H		H	H	M		
DC9.	Drawing rich pictures								H	L			L					H	H	H		
DC10.	Videorecording									L								M	L	H		
DC11.	Use cases								H	M								H	M	H		
DC12.	Protocol analysis									L							H	H	M	M		
DC13.	Storyboarding									M					H			H	H	H		
DC14.	Role playing									L	L							H	M	H		
DC15.	Evaluation of existing systems		H		L		H		H	M								M	M	H		
DC16.	Change suggestions and problem reports									M					H			H	M	H		
DC17.	Focus groups	L		L				H		M				H				M	M	L		
DC18.	Studies of descriptive documents									M					H			H	M	H		
DC19.	Derivation from business requirements					H				L						H		H	H	H		
DC20.	Prototyping								H	H		H						L	L	M		
DC21.	Observations									L					H			H	M	H		
DA1.	Usability context analysis							H		H					H			M	L	L		
DA2.	Task analysis									M								H	M	H		
DA3.	Functionality matrix							H		L								H	H	H		
DA4.	Scenario modelling									M					L			M	M	H		
DA5.	Problem frames								H	H								M	L	H		
DA6.	Repertory grids								H	H								L	M	H		
DA7.	Group comparisons							H		H				H				L	M	L		
RG1.	Function heuristics					H			H	M								H	H	M		
RG2.	Heuristics for idea reduction							H		L				H				H	H	H		
RG3.	Heuristics for uncovering ambiguity	L		L				H		L								H	H	H		
RG4.	Affinity analysis									L								H	H	H		
RG5.	Redefinition techniques	L				H		H		L								H	H	H		
RG6.	Requirements priorization									L								H	H	H		
RG7.	Quality function deployment							H		M								H	M	M		
RG8.	Concept mapping								H	M								H	M	M		
RG9.	Formal methods								H	H		H					H	M	L	M		
RV1.	Validation checklists									L								H	H	H		
RV2.	Validate against standards		H							L								H	H	H		
RV3.	Review requirements							H		M								H	M	M		
RV4.	Test cases						H			M								M	M	H		
RV5.	Prototyping								H	H		H						L	L	M		
RV6.	Draft user manuals									M					H			M	L	H		
MS1.	Feasibility study				L					M		L						M	L	M		

Figure 2 – The selection matrix



Example, using the first situation factor *clarity project goal*.
(the numbers correspond with the numbers in the figure on the left)

1. The value of *clarity project goal* is *high*. Fill in ‘H’.
2. The influence of the situation factor is *medium.* Fill in ‘M’.
3. Only the value of the method fragment *questionnaires* corresponds with the value of the situation factor (*high*). Calculate the weighting factor for value *high* and influence *medium*. Fill in ‘2’.
4. Add up all weighting factors in the row of *questionnaires* and fill in the total (for example ‘26’).
5. Compare the totals and determine the rank of *questionnaires* based on the highest number. Fill in the rank (for example ‘3’).

5 Conclusions & Outlook

We believe that the aspects, that are typical for e-business, have a great impact on requirements engineering. Therefore, these aspects are incorporated in the situational method described in this paper. This provides a requirements engineering method tailored for e-business projects. Or, to be more specific, a method tailored for *every* single e-business project that is performed. The situational method consists of situation factors and method fragments, which makes the method flexible and adaptable. The flexibility is provided by offering several alternatives for techniques in each phase of requirements engineering. The adaptability is provided by offering the possibility to add method fragments to the method base, which is used to store them. Another important requisite of the new method is its ease of use. This is provided by the development of the selection matrix, which can easily be filled in. Finally, the situational method provides quality, by facilitating the selection of techniques that are best for a specific situation.

Although the situational method solves some of the weaknesses of other methods, it still has some shortcomings. It can only assist in finding necessary method fragments, and does not guarantee completeness. Furthermore, the method is complex, because it needs to be assembled based on factors which may not be clear in the beginning of the project. When the method is extended, the number of combinations and rules increase, and thus also the complexity. And, possible computerised support leads to a new dimension of complexity. Finally, the valuing of the situation factors and their influence, and the assessment of the heuristics that support the selection of method fragment, are based on a certain amount of subjectivity, which can only be reduced to some extent.

Validation of the situational method is a complicated matter. Using the method in an actual project is practically the only option. For a more theoretical validation, a case study could be used. If these ways of validating are not possible, a fully theoretical way is a final alternative. For this, a list of elements the method should contain, could be checked.

This paper describes the results of an initial research about this subject. In future projects, the situational method will be extended and improved. Further research includes the following: the advice *not* to select certain method fragments based on certain values of situation factors; the advice to select certain method fragments based on *two* (or even more) situation factors instead of one; the advice to select method fragments that fit together; the advice *not* to select method fragments that exclude each other; incorporating user experience or preferences in the selection process; a decision tool for valuing the situation factors; the integration of Analytic Hierarchy Process (AHP) in the selection process, to provide means for a more thoughtful selection of method fragments; integrating the situational method in existing approaches, like architecture styles; and increasing the flexibility of the method when project members are confronted with changing circumstances.

6 Literature

[1] The Standish Group, *Chaos*, Standish Group Report, 1995.

[2] F. Harmsen, I. Lubbers, and G. Wijers, “Success-driven selection of fragments for situational methods: the S3 model”, in Pohl, K., & Peters, P. (eds.). *Proceedings of the second international workshop on requirements engineering: foundations of software quality,* Aachener Beitrage zur Informatik, 1995, Band 13, pp. 104-115.

[3] S. Jones, M. Wilikens, P. Morris, and M. Masera, “Trust requirement in e-business”, *Communications of the Association for Computing Machinery*, 2000, 43(12), 80-87.

[4] A. Finkelstein, A. Savigni, “A framework for requirements engineering for context-aware services”, *Proceedings of 1st International Workshop From Software Requirements to Architectures (STRAW 01),* Toronto, Canada, 2001.

[5] J. Gordijn, H. Akkermans, and H. van Vliet, “Value based requirements creation for electronic commerce applications”, *Proceedings of the 33rd Hawaii International Conference On System Sciences*, Hawaii, USA, 2000.

[6] Harmsen, A.F., *Situational method engineering,* Doctoral dissertation, University of Twente, Enschede, 1997.

[7] Slooten, K. van, *Situated methods for systems development,* Doctoral dissertation, University of Twente, Enschede, 1995.

[8] Hoef, R. van de, F. Harmsen, and G. Wijers, “Situatie, scenario & succes”, *Memoranda Informatica*, Internal technical report, University of Twente, Enschede, 1995.

7 Author CVs

Ir. Alexander Smeitz

Alexander Smeitz studied Business Information Technology at the University of Twente. Since 2003 he is working for TNO, the largest independent research institute in the Netherlands.

Alexander can be contacted at:

TNO-FEL
Oude Waalsdorperweg 63
2509 JG, The Hague
The Netherlands

smeitz@fel.tno.nl

Ir. Erwin Folmer

Erwin Folmer studied Technical Business Administration at the University of Twente, with ICT as specialization. His former employers are Baan Development and KPN Research. Since 2001 he is researcher & consultant for TNO E-Business.

Erwin can be contacted at:

TNO E-Business
Colosseum 27
7521 PV Enschede
The Netherlands

folmer@fel.tno.

Simulation of Requirements Phase Improvement for Telecom Systems using System Dynamics

Naeem Akhtar, Laique Ahmad, S.Tauseef Ur Rehman, Sikandar Hayat Khiyal
Department of Computer Science, International Islamic University, Islamabad, Pakistan

Zaigham Mahmood
Derbyshire Business School, University of Derby, DE22 1GB, UK
Department of Computer Science, International Islamic University, Islamabad, Pakistan

Abstract

This paper presents a study with the intent to examine the opportunities provided by creating and using simulation models of software development processes. A model of one software development project was created through means of system dynamics, with data collected from documents, interviews and observations. The model was run using a commercially available simulation tool. The simulation runs indicate that increasing the effort spent on the requirements phase, to a certain extent, will decrease the lead-time and increase the quality in similar projects. The simulation model visualizes relationships between the software processes, and can be used by project managers when planning future projects. The study indicates that this type of simulation is a feasible way of modelling the process dynamically although the study calls for further investigations as to how project and process managers can benefit most from using system dynamics simulations.

Keywords

System Dynamics, Simulation, Requirements Analysis, Case Study, Software Process Improvement, Lead Time, Quality Factors

1 Introduction

This study was performed in cooperation with the telecommunication wing of Nippon Electric Corporation, Japan and is based on a development project carried out in 2001. This study is part of a much bigger project aimed at improving the system reliability issues in Telecommunication.

As a step in the constantly ongoing work with quality improvements at NEC, this study was carried out to examine and show if simulation can be used for visualizing how different factors affect the lead-time and product quality. One of the most important factors that affect the lead-time of the projects and the product quality is the allocation of human resources to the different process phases. Thus, the focus of this study is on resource allocation.

Simulation is commonly used in many research fields, such as engineering, social sciences and economics. It is a general research methodology that may be applied in many different areas. Software process modeling and improvement is, of course, no exception and simulation has started to gain a foothold. For example, in [1] a high-maturity organization is simulated with system dynamics models, and in [2] a requirements management process is simulated with a discrete event simulation model. In [3] an overview of simulation approaches is given.

There are several advantages of building and simulating models of software processes eg new knowledge can be gained that can help to improve current processes. Simulation can also be used for training and to enforce motivation for changes.

The objectives of the study presented here are two fold: firstly, to investigate if it is possible to develop a simulation model that can be used to visualize the behavior of selected parts of a software process, and, secondly, to evaluate the usefulness of this type of models in this area. The objective of the model is to identify relationships between processes and mechanisms within a project. The study is focused on the tendencies of the simulation results and not the quantitative aspects. A complete quantitative analysis is currently underway at International Islamic University, Islamabad, Pakistan. Its results will be reported in a future publication.

The outline of the paper is as follows: In Section 2, the method used in the study is described. Section 3 describes the execution of the simulation model. Sections 4 and 5 present, discuss and summarize the results obtained.

2 The Method

This project was designed as a case study. Case studies are most suitable when data is collected for a specific purpose and when a sub goal of the study is to establish relationships between different attributes. A main activity in case studies is observational efforts.

With support from existing results in the literature [4,5], our research was conducted as three consecutive steps: problem definition, simulation planning and simulation operation. This methodology is based on the process-chain concept, but due to a lack of enough appropriate and reliable data, the process in practice went into an interactive pattern. In the first phase, i.e. problem definition, the problem was mapped. Then through deeper definition and delimitation, an agreement was reached as to the objectives of the study.

The main activity in the second phase of the study, simulation planning, was to identify factors influencing the product quality. For this, we wished to test the idea of using simulation models. This seemed natural as most of the ideas for the quality factors were picked up from within the organization eg through interviewing the project team and through available documentation. To create a broader perspective, results and ideas were also taken from the available literature on software processes and related aspects. Influence diagrams were built which included different quality factors and their relationship to each other, but the primary focus was on their effects on lead-time and product quality.

The third phase, operating the simulation model, started with translating a small component of the theoretical model into a simulation tool. Initial test cases showed that the simulation tool worked reasonably properly. More features were added from the theoretical model into the simulation tool and more test runs were performed. The verification and validation of the model was made stepwise through the input of the whole model into the simulation tool, and the comparisons carried out using documentation and by discussions with the lead engineers at NEC, Japan.

3 Development of the simulation model

In simulation modeling, there are two main strategies: continuous and discrete. The continuous simulation technique is based on system dynamics [6] and is mostly used to model the project environment. This is useful when controlling systems containing dynamic variables that change over time.

The continuous model represents the interactions between key project factors as a set of differential equations, where time is increased in steps. In the standard system dynamics tools, interconnected differential equations are built up graphically. A system of interconnected tanks filled with fluid is often used as a metaphor. Between these tanks or levels there are pipes or flows through which the variables under study are transported. Valves that can be controlled by virtually any other variable in the model limit the flows. This mechanism together with the level-and-flow mechanism can be used to create feedback loops. This layout makes it possible to study continuous changes in process variables such as productivity and quality over the course of one or several projects. It is however more problematic to model discrete events such as deadlines and milestones within a project [7,8].

In the discrete model, time advances when a discrete event occurs. Discrete event modeling is preferred when modeling queuing networks. In this context, in its simplest form, one queue receives time-stamped events. The event with the lowest time-stamp is selected for execution. The time-stamp indicates the current system time. When an event occurs, an associated action takes place, which usually involves placing a new event in the queue.

Since time is always advanced to the next event, it is difficult to integrate continually changing variables. This might result in instability in any continuous feedback loops [7,8].

To satisfy the aims of this study, which is to visualize process mechanisms, continuous modelling was used. This was chosen in order to include systems thinking [9] and also because it is often better than the discrete event modelling at showing qualitative relationships.

3.1 Problem definition

The current study is based on a process that is similar to the waterfall model [10]. The whole process is shown in Figure 1, but the simulation model was focused on the requirements and the test phases.

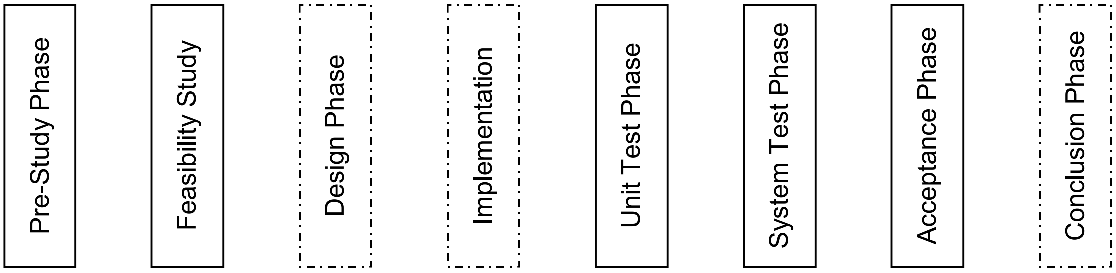


Figure 1. Process description

The other phases, shown as boxes with broken lines in Figure 1, were excluded to get a less complex model. The requirements phase includes the pre-study and the feasibility study stages. The test phases involve the unit, system and acceptance testing. All these types of tests are included, since the data available did not separate between test types and they also overlapped in terms of time.

3.2 Simulation planning

This step included identifying factors that affect the quality of the developed software and the lead-time of the project. This was made through interviews with the project team and based on information in the available literature [11,12].

The relevant project team consisted of seventeen persons, categorized in three groups with whom discussions were held continually during the entire study. Among the factors discovered during interviews, only those considered relevant to software development processes were selected. The identified factors are listed in Table 1.

Table 1. Factors that affect quality and lead-time

Number of personnel in the project
Level of personnel education
Level of personnel experience
Level of personnel salary
Level of personnel turnover
Communication complexity
Geographical separation of the project
Software and hardware resources Environment, e.g. temperature, lighting, ergonomics
Amount of overtime and workload
Level of schedule pressure
Level of budget pressure
Amount of new market requirements
Amount of requirements changes
Level of inadequate requirements
Amount of review
Amount of rework
Level of structure in the project organization
Standards that will be adhered to e.g. ISO and IEEE
Amount of software functionality
Testing and correcting environment and tools
Productivity
Amount of program documentation
Level of reusable artifacts, e.g. code and documentation

Discussions with those concerned pointed out to the most important factors in respect to both quality and lead-time. The major factors considered to affect the quality and lead-time were chosen to be included in the influence diagrams [13]. Influence diagrams for the requirements and the test phases were built to show how the chosen factors affected the lead-time and the software quality. The importance of each factor for each phase was also considered together with the relationships between the factors. The influence diagram for the requirements phase is shown in Figure 2. The factors shown in the diagram are further explained below.

- Amount of functionality is the estimated software functionality to be developed.
- Amount of new market requirements is a measure of the change in market expectations.
- Amount of requirements changes is a measure of the changes made in the requirements specifications.
- Amount of review involves reviewing requirements specifications.
- Amount of rework is the effort spent on reworking both new and inadequate requirements.

- Communication complexity is an effect in large project groups where an increasing number of participants increase the number of communication paths.
- Level of inadequate requirements is a measure of the requirements specification quality.
- Level of personnel experience is a measure of knowledge of the current domain.
- Level of schedule pressure is the effect of the project falling behind the time schedule.
- Number of personnel is the number of persons working with requirements specifications in the project.
- Productivity is a measure of specifications produced per hour and person.
- Time in requirements phase is the lead-time required to produce the requirements specifications in this project.

It is beyond the scope of this paper to present all details of the simulation model. In this paper the simulation model and related models, such as the influence diagram, are presented in some detail for the requirements phase only and the influence diagram for the test phase are not presented. Main reason being that the requirements phase is by its nature more intuitive and easy to understand than the test phase. For a presentation of details of the complete simulation model with all related models, refer to [14].

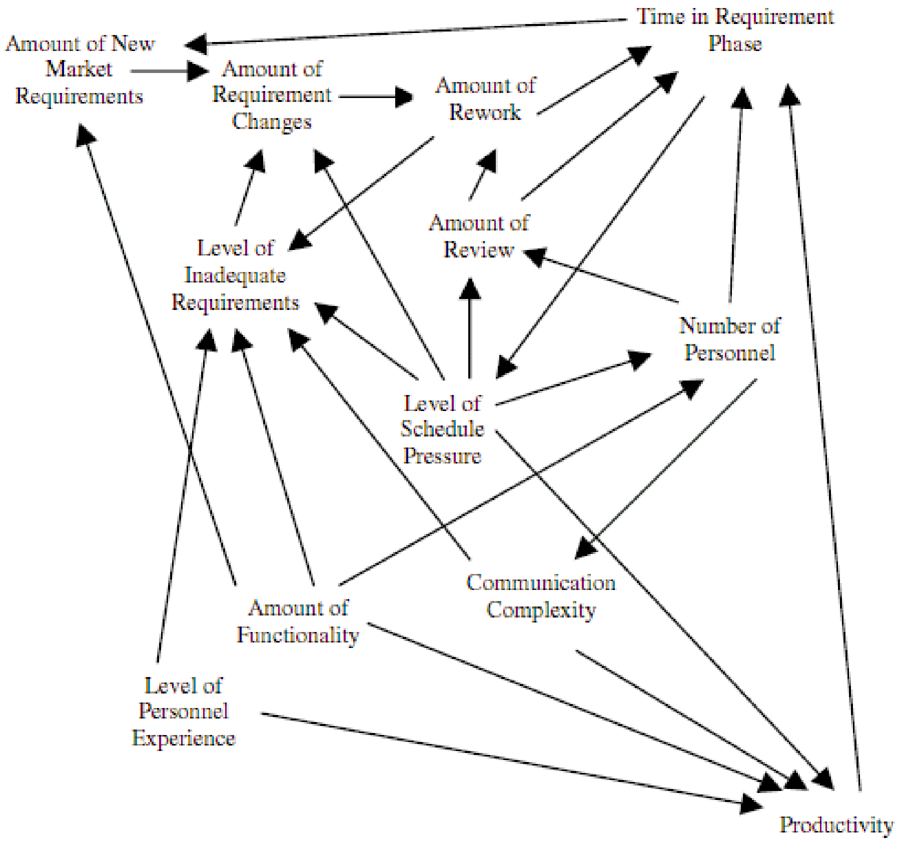

Time in Requirement Phase
Amount of New Market Requirements
Amount of Requirement Changes
Amount of Rework
Amount of Review
Level of Inadequate Requirements
Number of Personnel
Level of Schedule Pressure
Communication Complexity
Amount of Functionality
Level of Personnel Experience
Productivity

Figure 2. Influence diagram for the requirements phase

At the time when the influence diagrams were constructed, causal-loop diagrams were built to get a basic understanding of the feedback concepts. These diagrams are often used in system dynamics to illustrate cause and effect relationships [6]. When examining these relationships in isolation, they are usually easy to understand. However, when they are combined into long chains of cause and effect, they can become complex. The causal-loop diagrams increase the understanding of these complex relationships. Figure 3 illustrates how the schedule pressure affects the time spent in the requirements phase. An increased schedule pressure increases the error generation, due to a higher stress level. A high error density increases the amount of necessary rework and thereby increases the time in the requirements phase, which in turn increases the schedule pressure. At the same time, high schedule pressure increases the productivity because of its motivational role. Increased productivity decreases the time spent in the requirements phase, which in turn decreases the schedule pressure.

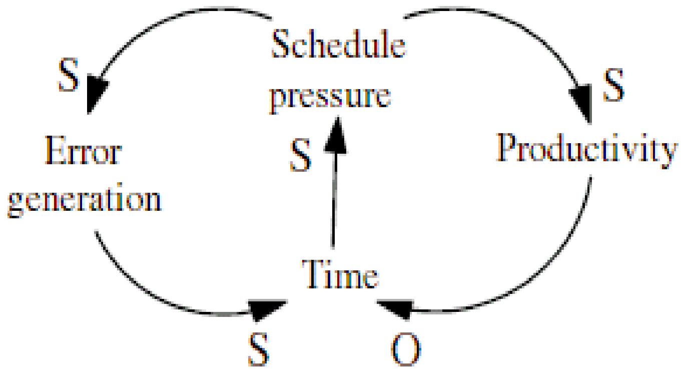


Figure 3: Causal Loop Diagram

Information about the relationships between the factors in the causal-loop diagram is shown by adding an “O” or an “S” to the arrows. An “O” implies a change in the opposite direction, while an “S” implies a change in the same direction.

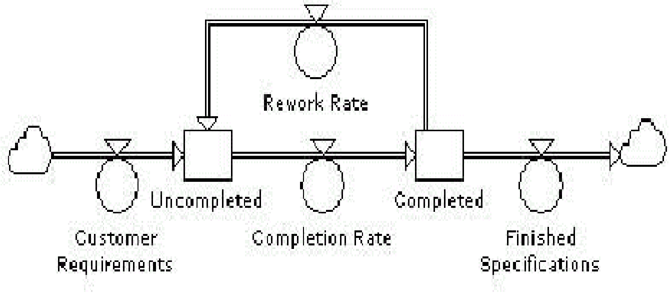
3.3 Simulation operation

The simulation model was built based on the knowledge gained from creating influence diagrams and causal-loop diagrams. The idea behind the model of the requirements phase is based on a flow of tasks, from customer requirements to finished specifications. In the requirements phase, there is a transformation from uncompleted to completed tasks by the production of specifications. A fraction of the specifications may not be acceptable and therefore there may be a need to rework the loop. Refer to Figure 4.

The test phase in the model is based on the same idea as the requirements phase and is built in a similar way. A flow of test cases is performed, a certain percentage of the functionality needs to be corrected and retested, and the rest is supposed to be acceptable.

This basic model was built using the SIMNEC simulation tool [15] and further developed with help from the factors in the influence diagrams. Factors from the influence diagrams were added to the model in order to affect the levels and flows. The causal-loop diagrams were also considered during the development, to ensure that the model was adapted to systems thinking.

To avoid formulating too complex a model, all the factors in the influence diagrams were not included. Some factors were included indirectly via the parameters in the model. These can be extracted from the parameters and it is therefore possible to control the user interface of the simulation tool eg the communication complexity that is included in the productivity. The construction was carried out incrementally, by adding a few factors at a time and then running the simulation. The values of the parameters were taken from project documentation except for *Amount of new market requirements*, the value for which was taken from [12]. This parameter was not available in project documentation but the value given in [12] is an average from several software projects and was considered to be valid for this project. Some values were estimated by iteration and verified by discussions with the personnel concerned at the organization.



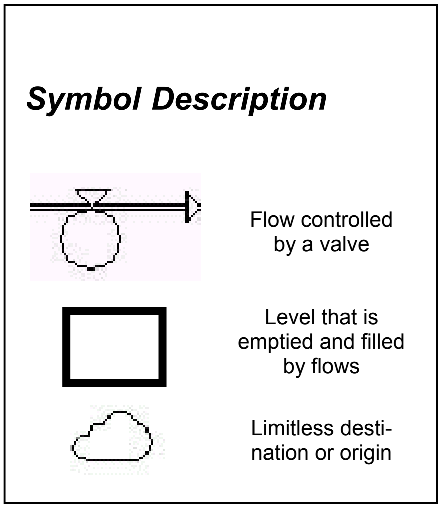

Symbol Description
Flow controlled by a valve
Level that is emptied and filled by flows
Limitless desti-nation or origin

Figure 4: Basic Model of Requirement Phase

The verification of the simulation model was carried out through checking that the amount of code used as an input to the model is the same as the amount of output code. The verification also included comparing the time taken in the simulation to the time given according to the project documentation to ensure that the estimations were correct.

The final model for the requirements phase is shown in Figure 5. The flows in Figure 4 form the basis of the final model, which is then further developed. To get a measure of the quality of the specifications, another flow was included, which counts the inadequate specifications. This measure affects the amount of defective code produced in the design and implementation phases which in turn affects the test phases. The design and implementation phases in the simulation model are modeled as a delay. An additional flow was added to the basic model to terminate this phase and start the next phases.

The rest of the additions to the basic model to produce Figure 5 can be grouped as follows, where each group originates from the influence diagram.

- The first group, `Lines of code` and `Functionality,` describes the functionality of the code to be developed. This controls the inflow to the phase.
- The second group, `Percentage, Effort` and `Duration,` allocates a percentage of the planned total effort to the requirements phase and is controlled from the user interface. This makes it possible to study how the amount of resources in the requirements phase affects the lead-time and the quality.
- The third group, `Productivity` and `Duration,` controls the completion rate of the specifications. The `Duration` also affects the amount of inadequate specifications because of the schedule pressure that might increase during the project's duration.
- The fourth group, `Amount of rework` and `Functionality`, decides the amount of the specifications that needs to be reworked after the reviews.

Note that some factors are part of more than one group. This is because some factors affect more than one factor.

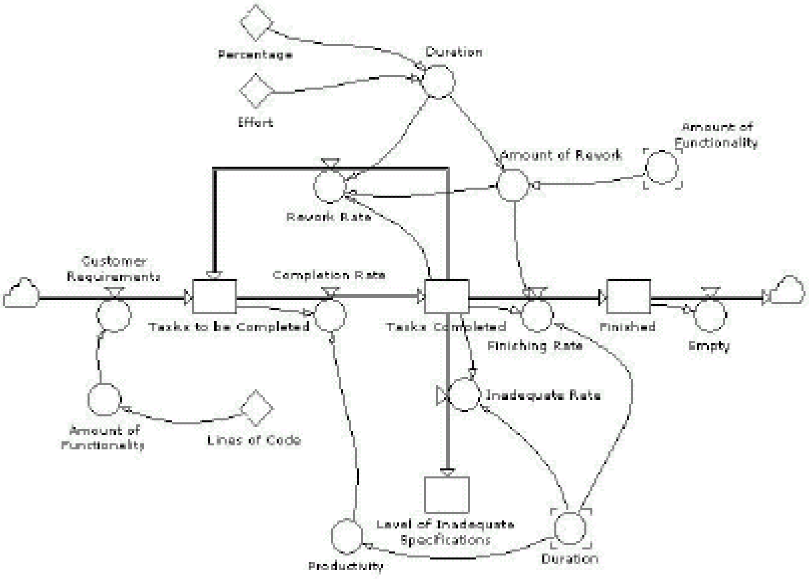


Figure 5: Simulation Model of Requirement Phase

4 Results from the simulation

The final model was run to show how a relocation of resources to different process phases affects the quality of the software products and the lead-time of the project. This model included both the requirements phase and the test phase. The model was run several times with different values of the percentage of the planned project effort spent on the requirements phase. Refer to Figure 6. The results are given in precise figures but since there are a number of uncertainties they should be broadly interpreted. The uncertainty arises because of the difficulty in measuring the values of the included factors. It is the trends in the results that are important and not the exact figures.

The simulation runs indicate that the effort spent on the requirements phase has a noticeable effect on the lead-time of the project. The decrease in days, when increasing the effort in the requirements phase, arises from the increased specification accuracy. A more accurate specification facilitates the implementation, decreases the error generation, and then results in a higher product quality from the start. This decreases the amount of necessary correction work and thereby shortens the time spent in the test phase. At a certain point, the total lead-time will start to increase again because the time in the test phase stops decreasing while the time in the requirements phase continues to increase. The time in the test phase stops decreasing because there is always a certain amount of functionality that needs to be tested at a predetermined productivity. Note that the number of days in Figure 6 is the total lead-time for the whole project.

In the same manner, the quality increases when increasing the effort in the requirements phase to a certain extent. The simulation runs indicate that the quality optimum appears in the same area as the lead-time optimum. The increase in quality originates from a higher specification accuracy, which is explained above. However, if too much effort is spent in the requirements phase, the quality will start to decrease again because there is less effort left for design, implementation and test tasks.

In order to verifiy our results, they were compared to results obtained by other researchers [12,16]. Their results points to the same magnitude of effort in the requirements phase for a successful project as the simulation results obtained through our study.

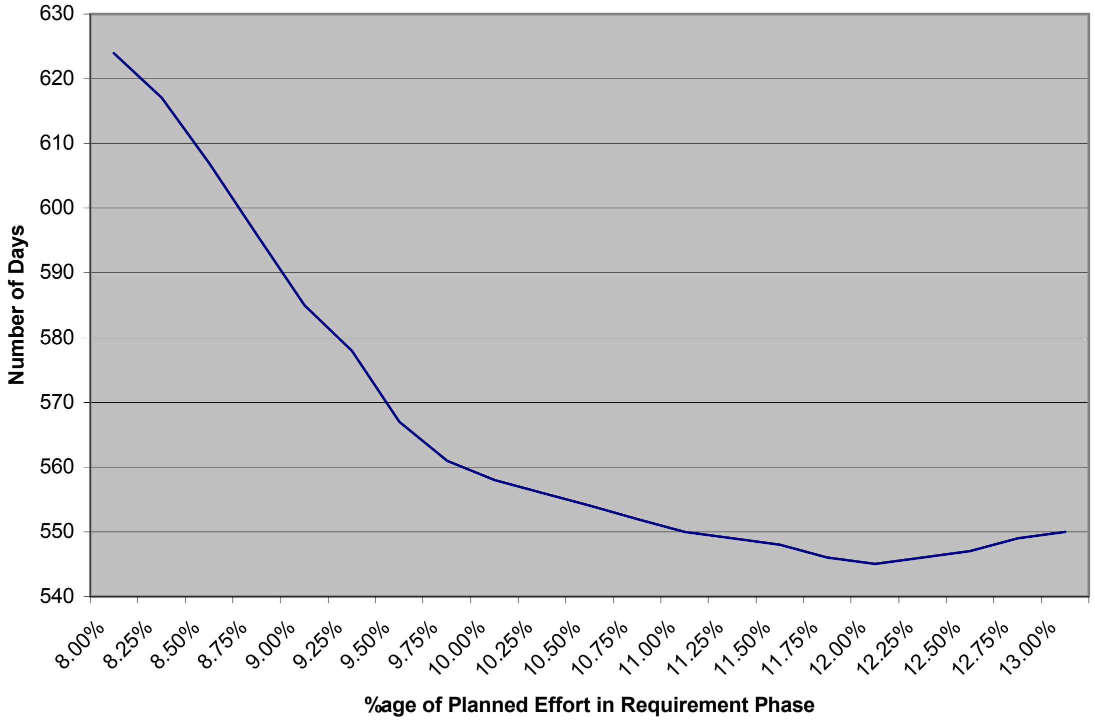


Figure 6: Lead-Time vs %percentage Planned Effort

To summarize, the simulations indicate that there is an optimum for both the quality and the lead-time. If the effort in the requirements phase is lower than the optimal value, increasing it towards the optimum will result in increased quality of the developed software and decreased lead-time.

5 Discussion

One output from the current study is a simulation model that visualizes different relationships in a software development process. A simulation of this kind can contribute to enhancing the systems thinking in an organization. Thereby, it is easier for the members of the organization to understand the relationships between the quality factors in the process.

The results from this kind of simulation cannot be interpreted precisely since there are a number of uncertainties (eg the included factors might not be the ones that affect the model the most, the assumed relationships between the factors might not be correct etc) and thus the values of the factors can be incorrectly estimated. However, it is the trends and the behavior apparent in the results that are important and by changing the parameters in the model it is possible to get a picture of how the process mechanisms interact. The results clearly show that there is an optimum for the effort spent in the requirements phase which can be intuitively expected for many projects.

A simulation of this kind can also be used to increase the motivation of the organization to work with quality issues and to increase the product quality early in the project. Another advantage is that the knowledge gained from such simulations can be used in the model building process. The procedure to build such models forces the participants to communicate their mental models with others and to create a common image of the organization’s direction.

Although, it is a simple model based on a single project which needs to be further developed, it can be used by project managers, by only changing the parameters. The structure of the model can also be easily changed by adding or deleting factors and adding or deleting relationships between factors.

Acknowledgement

The authors would like to thank T.Hashimoto and K.Tagaki at NEC Tokyo, Japan, for all their help with this study.

Literature

1 Burke, S., “Radical Improvements Require Radical Actions: Simulating a High-Maturity Software Organization”, Technical Report CMU/SEI-96-TR-024, Software Engineering Institute, Carnegie Mellon University, USA, 1996

2 Höst, M., Regnell, B., Natt och Dag, J., Nedstam, J, Nyberg, C., “Exploring Bottlenecks in Market-Driven Requirements Management Processes with Discrete Event Simulation”, Accepted for publication in Journal of Systems and Software, 2001

3 Kellner, M.I., Madachy, R.J., Raffo, D.M., “Software Process Simulation Modeling, Why? What? How?”, Journal of Systems and Software, Vol. 46, No. 2-3, pp. 91-105, 1999

4 Banks, J., Carson, J.S., Nelson B.L., *Discrete-Event System Simulation,* Prentice Hall, 1996

5 Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B.,Wesslén, A., *Experimentation in Software Engineering: An Introduction,* Kluwer Academic Publisher, 2000

6 Abdel-Hamid, T., Madnick, S.E., *Software Project Dynamics: An Integrated Approach,* Prentice Hall, 1991

7 Martin, R., Raffo, D., “A Model of the Software Development Process Using both Continuous and Discrete Models”, *Int. Journal of Software Process Improvement and Practice*, 5:2/3, June/Sept, pp. 147-157, 2000.

8 Martin, R., Raffo, D., “Application of a Hybrid Process Simulation Model to a Software Development Project”, proceedings of *PROSIM 2000,* July 12-14, London, UK

9 Senge, P.M., *The fifth discipline,* Random House Business Books, 1990

10 Sommerville, I., *Software Engineering*, Addison-Wesley, 2001

11 Fenton, N.E., Pfleeger, S., *Software Metrics: A Rigorous & Practical Approach,* International Thomson Computer Press, 1996

12 Jones, T.C., *Estimating Software Cost,* McGraw-Hill, 1998

13 Rus, I., Collofello, J.S., “Assessing the Impact of Defect Reduction Practices on Quality, Cost and Schedule”, proceedings of *PROSIM 2000*, July 12-14, London, UK

14 Andersson, C., Karlsson, L., “A System Dynamics Simulation Study of a Software Development Process”, CODEN:LUTEDX(TETS-5419)/1-83/(2001)&local 3, Dept of Comm Systems, Lund Institute of Technology, 2001

15 NEC Corporation, Tokyo Japan

16 Stewart, R.D., Wyskida, R.M., Johannes, J.D., *Cost Estimator’s Reference Manual,* John Wiley, 1995

Author CV

Dr Zaigham Mahmood

Name:	Zaigham Mahmood
Email:	Z.Mahmood@Derby.ac.uk
Qualifications:	BSc, MSc, MSc, PhD, CEng, MBCS
Position:	Senior Lecturer, Computing – University of Derby, UK Since Jan 1991 Currently, Visiting Professor – Int Islamic University, Islamabad, Pakistan From Jan 2003 – June 2003
Research and Scolarly activity:	10 publications Numerous book reviews and peer reviews of conference papers Member Int. Advisory Committee – SQM, INSPIRE, NWTIT

A Comparison of Automated and Manual Functional Testing of a Web-Application

M.sc. Geir Kjetil Hanssen and Dr. Torgeir Dingsøyr,
SINTEF Telecom and Informatics, Norway

Abstract

In this paper, we compare two methods of doing functional testing of a web-application: automated testing using a test tool, and manual testing. We find that automated testing is surprisingly time consuming when there are large changes in the user interface of the web-application, and when the test scripts were recorded automatically. We found that manual testing was faster when the testers got to know the application and the test plan better.

Keywords

Testing, automated testing, functional testing, manual testing, goal question metric (GQM).

1 Introduction

Testing is one of the most important success factors in software development. Nevertheless testing is often more or less omitted because of time squeeze or lack of knowledge or focus on the importance of thorough testing. Developers and other stakeholders in the development process may also regard testing as boring and not as challenging and interesting as design and development activities. Testing is an activity that in most cases has to be performed repeatedly in a 'test-correct-retest'-loop. One idea to help this problem is to automate testing procedures by using a tool that performs the tedious tasks and also speeds up the test process. There are several comprehensive automated test tools on the marked today. Initially, automated testing seems like the perfect solution to the testing problem. However, this allegation is most often found in the tool manufacturer's sales material. Is automated testing the perfect solution? Will the use of such a tool eliminate the testing problems? In this paper we look at these questions and present an experiment where the use of a modern test-tool has been compared with a manual test process. Some of the answers we found confirm some of the assumptions of automated test tools, but we also made some other observations that should be of interest for others that consider taking such a tool into use.

We have narrowed the scope of testing to functional regression testing of web-applications[1]. We define functional testing as [2] ”(1) testing that ignores internal mechanisms in a system or component, and focuses solely on the output that is generated as a response to selected input and run conditions, (2) testing that is carried out to evaluate that a system or component act according to functional specifications". This type of testing is done by testing the system via the user interface, basically by submitting data or performing actions and then observe the response and check whether it is as expected or not according to a test plan.

A web application is per definition an application that interacts with the user via a web browser which is served by a web server that hosts the application. The presentation logic is partially located on the server and partially located on the browser. Usually all application and business logic is located on the server side. This architecture brings some extra architectural complexity to the application compared with a rich client application which usually holds both the presentation and application logic, possibly also the business logic. Functional testing of web applications, both manual and automated, must handle this extra complexity [3].

A software system is a product of a continuous process and changes all the time. It is therefore needed to continuously test the system as it evolves to ensure that the product matches the requirements. This repeated testing is called regression testing.

Testing is, or should be, an inevitable part of any software development project. It can be expensive and may conflict with the need of resources for other more 'productive' activities. It is therefore important to establish cost efficient testing routines. Basically, functional testing can be performed in two ways, either manually or automated which means that the manipulation of the system and observation of the results is done by a tool. It is likely to expect that automation of testing, especially of regression testing, will speed up the process, and thereby reduce the costs, compared with manual testing which requires a person to perform the tedious interactions and observations. However, it is not quite clear what other effects automated testing might have, either positive or negative. In which cases would automated testing be the better choice? In which cases should manual testing be preferred? To investigate this, we have conducted an experiment where we compare manual and automated testing in a real software company developing a real software solution.

We continue the article by first describing some of the basic terms. We then give a short description of the settings of the experiment by describing the system being tested and the company that develops and test it. We then present our research method and the results, which leads to a discussion and a conclusion.

1.1 Manual testing

A manual testing process requires a test plan. This plan defines a series of actions and responses. Ideally, the test plan should cover all functionality of the system being tested and also test each function with a wide variety of data to find possible errors when using border values. The test plan is produced manually and it is up to the plan writer to ensure that all, or at least, the most important/critical functionality is included in the test[4].

The test itself is straight forward, the tester performs each action and records the results in a test log. After all actions are performed the recorded results are verified against the expected responses defined in the test plan. This will tell if the test succeeded or possibly if it failed and which function that failed.

1.2 Automated testing

Automated testing requires the use of a tool that completely or partially replaces the human tester. The tool uses a suite of tests that are recorded or programmed by the tester, usually expressed in a script language. The tests are detailed instructions of actions such as submit of data, menu browsing and other ways of using the functionality of the system trough the user interface. Each action is paired with an expected or wanted result, for example which data that should be returned, which dialogue that should be displayed etc. With a complete suite of tests the tool can, automatically, perform the tests, log the results and produce a test report describing which tests that succeeded and which that failed and possible give some explanation that can be used to correct the error or flaw. The motivation for using such a tool may be to speed up the regression testing and to improve the quality of the test results, e.g. the number of real errors found.

There are several tools for test automation available to day. For the type of testing being discussed here, 10 is a rough estimate. These tools have more or less a common set of basic functions::

- Recording of tests. The user instructs the tool by performing a sequence of actions which the tool records, usually in some type of script language in such a way that the actions can be reproduced by the tool.
- Script editing. The recorded actions can be edited to fine tune or change the actions.
- Reporting. The tool produces a report that give details about the automated test run.
- Browser independence. The tool supports the use of the most common web browsers by accessing the system using the standard HTTP-protocol
- Change tolerance. The tools handles minor changes in the user interface, e.g. changed location of a login dialogue box.
- Test library. The tool holds a library of recorded test to be reused.

Beyond this, the different tools vary regarding functionality and costs.

In this experiment the test tool being used was Rational Robot from Rational Inc. [2] This tools has a common set of functionality with some other of the most used tools today. It is therefore likely to believe that the results from this investigation apply to other software companies using other, but similar, tools. This paper is not meant to focus specifically on Rational Robot, others have done such assessments comparing several tools [5] including Robot.

1.3 The context

The experiment was done at a small Norwegian software company, in this paper named as Acme Software. This was a part of an internal project focusing on improving the test process. The company wanted to try out a test tool to see if it could improve their test process, which traditionally had been performed manually.

This work started out by selecting the tool to test. Based on a requirements list a survey was performed to find the most appropriate tool. A number of 11 tools were identified. Technical specifications was collected and compared with the requirements. This part of the work concluded that Rational Robot seemed to be the most appropriate tool.

The system being used in the experiment is a company management system, named CMS. It handles project resource management, billing, salary and travel costs etc. CMS is a three layered web application, with a standard database, a web server holding most of the business and application logic and a presentation layer that are accessed using an ordinary web browser. The interface to the server side is kept inside a dynamic link library (DLL) that takes input from the browser and generates HTML-pages as a response (also called an ISAPI-application). There is no preconfigured web pages including scripts on the web server, but seen from a functional testing perspective this does not matter. The test tool interacts with the system being tested using standard HTTP request-response sequences.

2 Method

The starting point for this work was that the company wanted to improve their test process which up till this point was done manually and on a more or less ad-hoc basis. We started out the work by having a GQM-workshop [2] to define which research questions we needed to answer to be able to select the appropriate testing strategy for this company, that is 'manually or automated testing'. The GQM-workshop resulted in the following list of questions to be answered:

a) Is automated testing faster than manual testing?

b) Does automated testing reveal more errors/flaws than manual testing?

c) Is automated testing more interesting (for the testers) than manual testing?

d) How often must automated test be re-recorded?

e) What parts of the test plan is better done manually?

f) Does the automated test tool report incorrect errors?

Then a survey was done to select the apparently most appropriate test tool. This selection was based on a list of requirements that was defined by the company.

To compare the selected test-tool with manual testing we formed an experiment. The basic idea was to perform exactly the same test plan automatically and manually and then measure the time needed to run the test plan.

2.1 Experiment set-up

We carried out a quasi experiment on two versions of the CMS system, each with a manual and automated test, each done by two testers which give a total of eight tests.

A problem in doing repetitive tests of the same system is that the person who is doing the experiment will learn more about testing and about the system under test while doing the test. To make up for this problem, we decided to organise two parallel sets of tests, where one subject would perform manual testing and then automated testing, while the other would first do automated testing, then manual testing on the same version of the CMS system.

We then got a list of experiments as follows:

Step	System	Activity
1	-	Test tool course for A and B
2	CMS August 2001	Manual test for subject A
3	CMS August 2001	Automated test for subject B
4	CMS August 2001	Automated test for subject A
5	CMS August 2001	Manual test for subject B
6	Today's CMS	Manual test for subject A
7	Today's CMS	Automated test for subject B
8	Today's CMS	Automated test for subject A
9	Today's CMS	Manual test for subject B

Table 1 - experiments

Before conducting the experiment, we interviewed the subjects A and B to find if their knowledge of the CMS system differed. We conducted a course in the test tool to make subjects equally skilled in using the system.

The two versions of the system we used for testing were:

- An old version of CMS, from August 2001, named CMS1
- Today's version of CMS (2002), named CMS2

We used the same database for each of the two versions.

For each experiment, we had one observer watching the subjects performing the functional test, either using the tool or doing it manually. The observers asked the subject to "think aloud", and:

- Registered what the subject was saying for each test-case in the test-plan.
- Registered the time usage for each test-case in the test-plan.

2.2 The test plan

The test plan being used for both the manual and automated tests includes 73 actions to be performed, addressing the most important features of the system. Examples: "Change start date to 2002.01.13 and verify that the systems displays an error message" and "Choose week 44 and register 8 hours on Monday on the 'Prestudy'-project, verify that 'planned hours' equals 50 and 'rest' equals 34".

The manual test means that the tester goes trough the list, performing the actions and logging the results, step by step. The automated tests mean that the tester records a script for each of the actions and uses the test tool to describe the expected result from each action. The script is then executed by the test tool which compares the actual results with the expected results and reports any variance.

3 Results and discussion

The results from the eight tests are presented in the table below:

	CMS version	*CMS1*						*CMS2*					
	Type of test	Manual		Automated				Manual		Automated			
	Tester	A	B	A		B		A	B	A		B	
	Record/play			record	play	record	play			edit+ rec.	play	edit+ rec.	play
	Testplan, tasks												
1	Log in	1	2	9	2		1	1	1	3	34	0	29
2	Customer register	3	4	7	1		1	6	4	14	43	33	18
3	Project register - main project	5	5	9	1		14	4	3	4	5	16	10
4	Project register - sub project	6	6	12	1		2	2	2	6	16	14	3
5	Project memebership	10	5	8	10		1	2	5	6	8	18	3
6	Time registration	4	4	10	1		2	6	2	8	3	7	6
7	Travel bill - main	11	8	19	18		52	8	5	11	5	22	13
8	Travel bill - specifications	7	9	16	137		7	8	6	10	12	36	11
9	Travel bill - last part	6	4	6	4		3	3	2	3	3	2	7
10	Prosject status	1	1	3	1		1	1	1	1	4	2	2
Total time		*54*	*48*	*99*	*176*	*114*	*84*	*41*	*31*	*66*	*133*	*150*	*102*

Table 2 - recorded data (time)

(Note: The first test performed was CMS1, automated, tester B. For this test only the total time spent was recorded. After this it became clear that it was needed to record times for each task in the test plan. This explains the missing data.)

Based on these figures we can draw some conclusions:

- **Test recording using the tool was more time consuming than manual testing**

It was more time consuming to execute the test plan with the test tool in record modus (as opposed to just run the plan manually). The average for the two versions of the system and the two testers the manual test took 44,9% of the time it took to record the same test plan using the test tool. The reason for this is that the tester had to handle the test tool and that he had to spend some time to handle errors in the generated test script. The tester should ideally not notice that the tool registers the script. It also took some time to verify that the generated test script worked correctly. This could be a novice phenomenon. It is also reasonable to believe there will be fewer errors in the test scripts as the tester gets used to the test tool and its weaknesses and strengths.

- **Manual testing of version 2 of the system was less time consuming than the testing of version 1**

The figures show that the total tome for the two testers doing the manual tests of version 1 took 102 minutes while the total time for version 2 took 72 minutes. Manual test of version 2 took consequently 70,6 % of the time it took to test version 1 manually. By doing the tasks in the test plan once both testers get known to the system being tester and no less also got known to the test plan. This resulted in the reduction of time sent on the manual tests of version 2.

- **Playback of the recorded scripts was time consuming**

For version 1, tester 1 and 2 spent respectively 176 and 84 minutes just to run the generated test script. For version 2 the testers spent respectively 133 and 102 minutes. Ideally, this should take much less time than these figures shows. When the scripts were played the testers experienced that the generated scripts failed, which cost quite a lot of time for fault correction. The playback was interrupted repeatedly and the testers spent time to correct the problem and get started over again. One example is that the test tool did not handle the change of forms (in the web browser) smoothly, therefore the tester had to manually by-pass this problem, which was time consuming. This tells us that it is important (at least for this tool) to the system being tested must have exactly the same state as when the script was recorded. In this case this means that the database must be identical, it must hold exactly the same customer data, project data etc.

- **Straight reuse of the test script was problematic**

Initially, the idea was that a record test script (or set of scripts) should be replayed on version 2 of the system and thereby save time as opposed to repeated manual testing. The figures show that the tester nevertheless had to spend much time on re-recording/editing of the scripts. The reason for this is that there had been done quite a lot of changes in the system. Consequently, the scripts recorded for version 1 was not completely reusable for version 2. Both changes in GUI and changes in functionality affect the reusability of test scripts; extensive changes in the system require extensive changes in the test scripts. If the extent of changes is extensive enough it will be more effective to just rerecord the complete test script instead of spending time correcting the script that was generated using a previous version of the system being tested. In this experiment we did not collect enough data to estimate a break-even value where script reuse is not cost effective.

Beyond these key findings we have some additional experiences that are worth mentioning:

- For the tool used in this experiment, it was practical to record the test plan into a set of scripts and to give the scripts meaningful names. This makes it easier to re-record or edit parts of the test plan and leave the partial scripts that are ok as they are.
- If the test tool are to be used during the development extension of a system it is practical to also continuously develop the test plan to keep it up to date with the system it is meant to test.

- When doing repeated tests (regression testing) the use of a tool for automated tests will reduce the total time spent on testing. However, it takes some repetitions to achieve this time profit. The following figure illustrates a possible scenario:

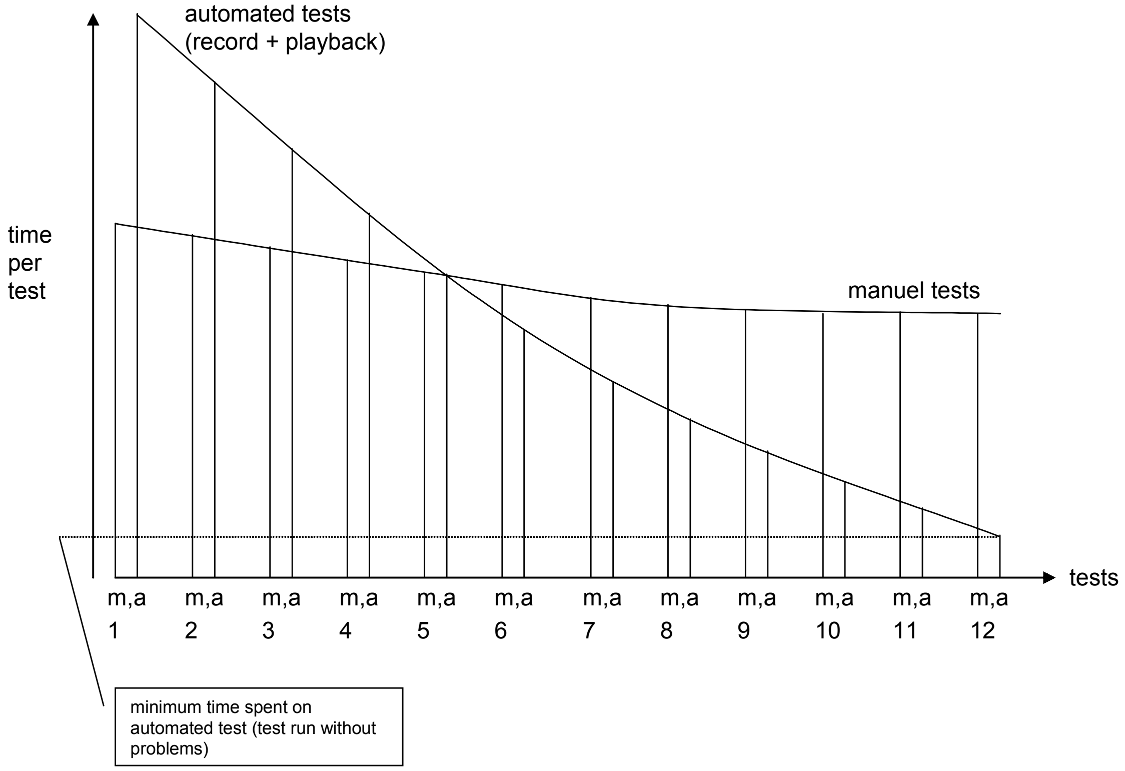


Figure 1 - a possible scenario

In this experiment we did not collect data to calculate the actual figure; the figure here presents the idea. The time used on repeated manual tests will decrease, but will reach a minimum that is higher than the minimum for automated tests. The time spent on automated tests will probably decrease rapidly for each iteration when errors/problems with the test scripts are fixed. After some iterations we will find a break even value. If the tool is not used frequently enough it will be more profitable to do manual testing. In addition, it also takes some time to get used to the tool initially.

4 Conclusion

After having performed an experiment with automated and manual test of two versions of a web-application, we found the following answers to the questions we presented earlier:

- Is automated testing faster than manual testing?

We found that the automated test tool was considerably slower for the two versions of the web application.

- Does automated testing reveal more errors/flaws than manual testing?

We found the same number of errors with manual and automated testing.

- Is automated testing more interesting (for the testers) than manual testing?

The testers appreciated working with the tool compared to doing the manual tests, but the script-editing with the test tool was rather tedious.

- How often must automated test be re-recorded?

We had to re-record test scripts very often due to changes in the user interface of the web application.

- What parts of the test plan is better done manually?

In this case, we found that all parts of the test plan were done more efficiently manually.

- Does the automated test tool report incorrect errors?

We found no incorrect errors in the error logs from the test tool.

5 Literature

1. Nguyen, H.Q., Testing Applications on the Web: Test Planning for Internet-Based Systems. 1 ed. 2000: John Wiley & Sons.
2. IEEE, IEEE Standards Volume One: Customer and Terminology Standards. 1999: IEEE.
3. Autotester, Product Overview. 2002. p. 4.
4. Koomen, T.a.P., Martin, Test Process Improvement. 1999: Addison Wesley. 218.
5. Yphise, Software Assessment Report: Functional test automatation tools - executive volume. 2002: Paris, France.

6 Author CVs

Geir Kjetil Hanssen

Geir Kjetil Hanssen (born 1969) is a Research Scientist in SINTEF Telecom and Informatics. He is a M.Sc. (Cand.Scient) from the Norwegian University of Science and Technology (1996). He works within the computer science area covering various aspects of information technology. System architecture is one of his focus areas, and he has experience in systems design and systems engineering methodology and processes. Other focus areas are transport telematics and process improvement where he assists several Norwegian software companies in improving their internal software development processes. He is working with messaging systems using XML based technology and a system for generating XML messages based on graphical message models (UML). He has experience from the IT industry, both as a systems engineer and developer and as a project leader.

Torgeir Dingsøyr

is a research scientist at the Department of Computer Science at SINTEF Trondheim, Norway. He wrote his doctoral thesis on “Knowledge Management in Medium-Sized Software Consulting Companies” at the Department of Computer and Information Science, Norwegian University of Science and Technology. He has published papers on knowledge management in software engineering, case-based reasoning and on software engineering education. In 2001, he spent half a year as a guest researcher at the Fraunhofer Institute for Experimental Software Engineering in Kaiserslautern, Germany.

Implementation of CMM Requirements Management Key Process Area using a Goal/Question/Metrics based Technique

Zaigham Mahmood
Derbyshire Business School, University of Derby, DE22 1GB, UK
Department of Computer Science, International Islamic University, Islamabad, Pakistan

Shoaib Adnan, Tauseef Ur Rehman
Department of Computer Science, International Islamic University, Islamabad, Pakistan

Abstract

This paper aims to present a framework for software measurements for the implementation of the goals of the Requirements Management Key Process Area (KPA) of the Capability Maturity Model (CMM), and provides practical guidance to implement the Requirements Management KPA. The CMM, developed by the Software Engineering Institute (SEI) is not well supported by software measurement and it is somewhat complex. An application of the Goal/Question/Metric (GQM) paradigm to the Requirements Management KPA is also presented. Hopefully, the metrics obtained, as a result of the current study, will help small-to-medium sized enterprises to satisfy the goals of the Requirements Management KPA.

Keywords

Capability Maturity Model, CMM, Requirements Management, Key Process Area, Goal/Question/Metric, GQM, Software Process Improvement

1 Introduction

Software processes are considered to be the main area for quality improvement. There are two main streams within Software Process Improvement (SPI) [1]. One is based on the assessment of organizations' capability, e.g. Capability Maturity Model for Software (SW-CMM) [2], Software Process Improvement Capability dEtermination (SPICE) [3], BOOTSTRAP [4], and the ISO9000 family. The other is based on measurements of software practices within an organization, e.g. Goal/Question/Metric (GQM) [5], Quality Improvement Paradigm (QIP) [6], and Application of Metrics in Industry (AMI) [7]. These approaches complement each other, however, they are seldom applied together [1] eg the SW-CMM developed by the Software Engineering Institute is intended to help software organizations to improve the maturity of their software processes, but is weakly supported by a measurement-based approach. In the present study, we propose a set of measures for the Requirements Management Key Process Area (KPA) of the SW-CMM. The aim is to combine the assessment and measurement based methodologies as mentioned above. The underlying assumption being that it is easier and less expensive to focus on the measurement and improvement activities of a specific process area rather than measure and improve all the process areas at once. The Requirements Management KPA has been chosen because it is important to control the continuing definition of requirements as they change throughout the software development life cycle.

Initially, our approach will be implemented for a small-scale academic project at the International Islamic University, Islamabad, Pakistan and the measures vigorously tested at Ms. Myson Engineering Systems, a medium sized company in Islamabad, Pakistan. Hopefully, the results of our study will help to demonstrate that a joint approach is more complete than an assessment or measurement based approach and that the measures obtained can be used in quantifying the amount of changes to requirements and to predict the cost of such changes.

In the present work, we apply the GQM paradigm, analyze and implement the Requirements Management KPA of the SW-CMM and its key practices [8].

We first present a comprehensive set of software measures for the implementation of the goals of the Requirements Management KPA within the SW-CMM. Then we detail a simple method for improving the Requirements Management activity. Finally, we present a practical approach with general guidance for small-to-medium sized enterprises (SMEs) trying to fulfill the goals of the KPA.

2 The Requirements Management KPA of the CMM

As defined by the SEI Technical Report [2], the CMM is composed of 5 distinct levels: *Initial, Repeatable, Defined, Managed, Optimizing* where each level, except the *initial*, has several Key Process Areas (KPA) eg one level 2 KPA is "Requirements Management". According to [8], its purpose is "to establish a common understanding between the customer and the software project of the customer's requirements that will be addressed by the software project". This suggests that the requirements of a software project should be complete, clearly documented, unambiguous, controlled, etc., in order to design a software product, which satisfies the customer's needs. Often, requirements change through out the software development life cycle but the control of the change requests is usually poor. The activity of "Requirements Management" is focused on the control of the requirements gathering, establishing an agreement between the customer and the software team on the requirements, checking, reviewing and managing the changes on requirements. The objective is to ensure that the final product meets the customer requirements.

3 The GQM Paradigm

The Goal/Question/Metric (GQM) paradigm is a method for helping an organization to focus the measurement program on their goals. It suggests that an organization should have specific goals in mind before data are collected [5]. GQM does not specify concrete goals - it provides a structure for defining goals and refining them into a set of quantifiable questions. The questions imply a specific set of metrics and data to be collected in order to achieve these goals.

The GQM paradigm consists of three steps:

- Specify a set of goals based on the needs of the organization and its projects. Determine what should be improved or learned. The process of goal definition is supported by templates (refer to figure 1 [5]) and the goals are defined in terms of purpose, perspective, and environment.
- Generate a set of quantifiable questions by translating business goals into operational statements. Basili and Rombach [5] provide guidelines to classify questions as product-related or process-related.
- Define a set of metrics that provides quantitative information to answer the quantifiable questions. Generally, each metric can provide information to answer several questions and sometimes a combination of metrics is needed to provide the answers to questions.

Purpose:

Analyse some

(objects: processes, products, other experience models, …)

for the purpose of

(why: characterization, evaluation, prediction, motivation, improvement, …)

Perspective:

with respect to

(focus: cost, correctness, defect removal, changes, reliability, user friendliness, ...)

from the point of view of

(who: user, customer, manager, developer, corporation, ...)

Environment:

in the following context

(problem factors, people factors, resource factors, process factors, ...)

Figure 1: Goal Template

Once the steps are identified, data are collected and interpreted to produce answers to the quantifiable questions to fulfill the goals of the organization [1,5].

4 Application of the GQM to CMM

4.1 Step 1: Specifying goals

The CMM and the GQM can be easily intertwined. The CMM defines one or more goals for each KPA as shown in figure 2. These goals can be used at the first step of the GQM process. There are two distinct goals for the Requirements Management KPA [8] which states the following:

> *"System requirements allocated to software are controlled to establish a baseline for software engineering and management use"*
>
> *"Software plans, products and activities are kept consistent with the system requirements allocated to software"*

The first goal focuses on the control of requirements to set up a baseline that serves as a standard by which things are measured or compared. If the requirements are not controlled, there will be no clear picture of the final product. The main focus of the second goal is the consistency between the requirements and the software product developed as a result of these requirements. The goals presented above can be redefined by applying the goal template in figure 1 as follows:

Goal 1: To *Analyze* the system requirements allocated to software *for the purpose of* establishing a baseline *with respect to* the control of the requirements *from the point of view of* academy and the software manager*, in the context of* the company where the Requirements Management is implemented.

Goal 2: To *Analyze* software plans, work products and activities *for the purpose of* consistency with the system requirements allocated to software *from the point of view* of academy and the software manager, *in the context of* the company where the Requirements Management is implemented.

4.2 Step 2: Generating questions

The second step in the GQM process requires the generation of a set of quantifiable questions. For the purposes of the present study, the questions have been produced by applying the guidelines for process related questions [5], analyzing the goals of the Requirements Management KPA and its Key Practices [8] word by word and by referring to research papers. For some questions a rationale will also be given to better understand the questions.

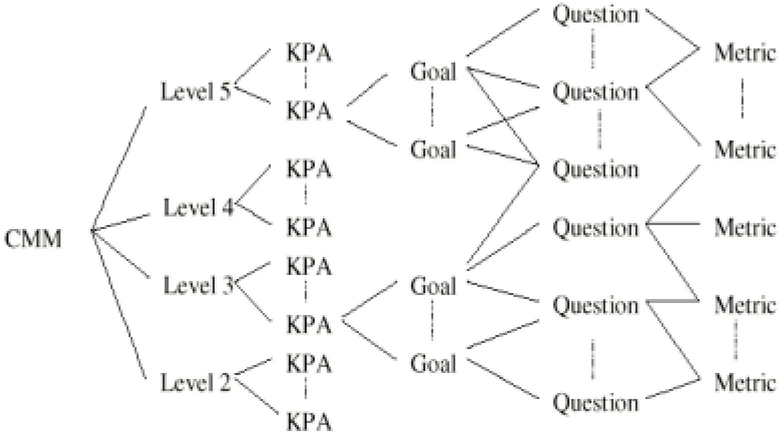


Figure 2: Relationship between CMM and GQM

Questions for the First Goal of the Requirements Management KPA

Analysis of the first goal suggests two fundamental questions: how can the requirements be controlled? And why should we control them? We know that it is not possible to specify at the beginning exactly what the customer wants. Neither is it possible to dictate the frequency or desirability of changes. The changes can come at the worst moment and impede our ability to complete a project within the allocated resources of budget and time. The one possibility is to control the continuing definition of requirements as they change throughout the development life cycle [9].

Any information on requirements can help to establish control. It is especially important to know the starting and the final sets of requirements. To increase the control of the requirements, their status (eg *new, analyzed, approved, documented, rejected, incorporated into the baseline, designed, implemented* or *tested*, etc) as well as their stability could be investigated (refer to questions 1 and 2 in table 1). Requirements stability is concerned with the changes made in requirements, therefore a set of questions (eg questions 3-9 in table 1) about requirements changes can be defined to refine the question 2. The level of requirements stability can also be measured by having information about the size of the requirements and by identifying problematic requirements (refer to questions 10-12 in Table 1).

Once there is control over the requirements, a baseline can be established. Thus, some questions about how the requirements are documented, and how many of them are included in the baseline, are defined (refer to questions 13-15 in table 1).

Table 1: Questions and measures for the 1st goal of the Requirements Management KPA

Sr.No.	Questions	Measures
1	What is current status of each requirement?	Status of each requirement
2	What is the level of the stability of the requirements?	# initial requirements # final requirements # changes per requirement
3	Why are the requirements changed?	# initial requirements # final requirements # changes per requirement # test cases per requirement Type of change to requirements Reason of change to requirements Major source of request for a change to requirements Phase where change was requested
4	What is the cost of changing the requirements?	Cost of change to requirements Size of a change to requirements
5	Is the number of changes to requirements manageable?	Total # Requirements # changes to requirements proposed

		# changes to requirements open # changes to requirements approved # changes to requirements incorporated into base line # changes to requirements rejected The computer software configuration item(s) (CSCI) affected by a change to requirements Major source of request for a change to requirements Requirement type for each change to requirements # requirements affected by a change
6	Does the number of changes to requirements decrease with time?	# changes to requirements per unit of time
7	How are affected groups and individuals informed about the changes?	Notification of Changes (NOC) shall be documented and distributed as a key communication document # affected groups and individuals informed about NOC
8	How many other requirements are affected by a require ment change?	# requirements affected by a change
9	In what way are the other requirements affected by a requirement change?	Type of change to requirements Reason of change to requirements Phase where change was requested
10	Is the size of the requirements manageable?	Size of requirements
11	How many incomplete, inconsistent and Missing allocated requirements are identified?	# incomplete requirements # inconsistent requirements # missing requirements
12	Does the number of “To Be Done” (TBD) decrease with time?	# TBDs in requirements specifications # TBDs per unit of time
13	How are the requirements defined and documented?	Kind of documentation
14	Are the requirements scheduled for implementation into a particular release actually addressed as planned?	# requirements scheduled for each software build or release
15	How many requirements are included in the baseline?	# baselined requirements phase when requirements are baselined

Questions for the Second Goal of the Requirements Management KPA

The purpose of the second goal is mainly to maintain consistency between the requirements and the software project, therefore it is suggested that traceability among the software documents is kept. Traceability between requirements and the software project facilitates the analysis of the effects of a software change and reduces the effort to locate the causes of a product failure. Tracking the requirements and changes made to requirements can help to maintain traceability among the requirement documents (refer to questions 1-7 in Table 2).

Changes to requirements also affect the status of associated documents. Therefore it is suggested that a status check (eg identified, evaluated, assessed, documented, planned, communicated to affected groups and individual, and tracked to completion etc) of other documents (eg software plans), work products, and activities is also kept.

Table 2: Questions and measures for the 2nd goal of the Requirements Management KPA

Sr.No	Questions	Measures
1	Does the software product satisfy the requirements?	• # initial requirements • # final requirements • # test cases per requirement • Type of change to requirements
2	What is the impact of the changes to requirements on the software project?	• Effort expended on Requirements Management activity • Time spent in upgrading • # documents affected by a change
3	What is the status of the changes to software plans, work products, and activities?	• Status of software plans, work products, and activities
4	Are the requirements scheduled for implementation into a particular release actually addressed as planned?	• # requirements scheduled for each software build or release
5	How are the requirements defined and documented?	• Kind of documentation
6	Does the number of TBDs prevent satisfactory completion of the product?	• # TBDs in requirements specifications
7	Are all development work products consistent with the requirements	• # inconsistencies

4.3 Step 3: Defining goals

The third step of the GQM process is to define a set of metrics that provides quantitative information necessary to answer the questions. For the purposes of the present study, the measures are presented in tables 1 and 2. There are overlaps among the questions for the two goals and the measures. This is acceptable as the same measures can be used to give information to answer different questions. Some of the measures are suggested by SW-CMM (eg status of allocated requirements, change activity, cumulative number of changes to allocated requirements etc) - whereas several of the other measures are those recommended by [10].

Once the three steps of the GQM paradigm are defined, an organization needs to determine ranges for "good data" (for instance a company could accept no more then three change requests per week). The data needs to be collected and compared to the "good data", and eventually improvement actions are taken. A comparison of the data for the actual project with the data collected for previous projects will provide a baseline for the requirements and give meaning to the measures.

5. Testing the measures in a company

At the time of writing, a framework has been developed and we are just beginning to test the measures at Ms. Myson Engineering Systems, a medium-sized company in Islamabad, Pakistan, which has used SW-CMM and successfully improved their software process to CMM level 3. The ongoing project is an expansion of a project for Ufone (a cellular telephone service provider).

The tests are being conducted following the guidelines suggested in [11]. The results are not yet available, as only the first step of the experiment process (definition of the experiment) has been completed:

Object: Requirements Management activity at Ericsson Erisoft AB in Umeå,
Sweden

Purpose: Evaluate the impact of the measures to improve the Requirements
Management activity

Quality focus: Control of the requirements

Perspective: Academy

Context: Medium-sized company

Results of the experiment will be made available in a later publication.

As we are beginning to test the measures, we are also in the process of planning the study (which is the second step of the experiment process). Here we are also focusing on the company's organizational structure, and the software processes used in the company. The processes currently used are a modified version of PRINCE2 as the management process and RUP (Rational Unified Process) [12] as the software development process. The authors also intend to map the terminology used by the people in the company to the CMM terminology. Some data from studying one particular increment has been compiled. A preliminary conclusion of this study is that the domain analysis is not done very deeply and the requirements are added and changed during the software life cycle.

6. Concluding Remarks and Future Directions

An application of the GQM to the Requirements Management KPA has been reported.

The set of questions and measures presented should be tailored to the particular organization. All CMM level 1 companies interested in improving the Requirements Management activity are suggested to select an appropriate subset of these measures as the starting point. For instance, a level 1 organi zation would probably have poorly defined requirements. The visibility of the process is very low at this level, and it is difficult to measure the process. Therefore, it is suggested that they count the number of requirements and changes to those requirements to establish a baseline. Other level 1 companies (eg those in [13]) need to document the requirements before starting to measure. A company like Ms Myson Eng. in Islamabad, Pakistan (who has improved the process through the CMM in the past), can collect all the data regarding the change activity. This is possible because information about change requests is stored in reports and database.

The measures produced, as a result of the study, will provide the organization with improved visibility and better insight into the Requirements Management activity, thus improving the software process a small step towards the goal of having a repeatable process. The measures will also be used in quantifying the amount of changes to requirements and to predict the cost for such changes, thus helping to control requirements and changes to the requirements. If the process is repeatable, more information on requirements can be collected such as the type of each requirement (eg database requirement, interface requirement, performance requirement, etc.) and change requests to each type. In general, the metrics collection will vary with the maturity of the process.

In the present study, we have combined the assessment and measurement based methodologies by analyzing the Requirements Management KPA of the SW-CMM and its key practices [8], and applying the GQM paradigm. We believe the results obtained will be relevant to areas such as Requirements Management, Software Measurement, Software Process Improvement, Software Quality, etc.

The current project is continuing. The measures obtained, will be used for the elicitation of requirements information in the aforementioned company. For each measure proposed, improvement actions will be suggested to help the management team. After these actions are taken, an enterprise can follow the approach described in AMI [7], which suggests a four step process (analyze, act, metricate, and improve) to continue with the improvement of the KPA The measurements will be automated as far as possible. Finally, the application of the GQM to all the KPAs of the CMM levels will be considered and reported in future publications.

Literature

[1] van Solingen, R., and Berghout, E., "The Goal/Question/Metric Method - A practical Guide for Quality Improvement of Software development", McGraw Hill, London, 1999.

[2] Paulk, M.C., Curtis, B. et al, "Capability Maturity Model for Software Version 1.1", Software Engineering Inst Tech Report, CMU/SEI-93-TR-24, ESC-TR-93-177, Pittsburgh, PA, 1993.

[3] El Eman, K., Drouin, J.N., and Melo, W., "SPICE The Theory and Practice of Software Process Improvement and Capability Determination", IEEE Computer Society Press, Los Alamitos, California, 1997.

[4] Kuvaja P. and Bicego, A., "BOOTSTRAP - Europe's Assessment Method", IEEE Software, pp. 83-85, May 1993.

[5] Basili, V.R. and Rombach, H.D., "The TAME project: Towards improvement-oriented software environments", IEEE Trans. Software Engineering 14(6), 1988.

[6] Basili, V.R., Caldiera, C., and Rombach, H.D., "Experience Factory", Encyclopædia of Software Engineering Volume 1, John Wiley & Sons, 1994.

[7] Pulford, K., Kunntzmann-Combelles, A., and Shirlaw, S., "The Ami Handbook: A Quantitative Approach to Software Management", Addison Wesley Longman UK, 1996.

[8] Paulk, M. C., Weber, C.V. et al, "Key Practices of the Capability Maturity Model Version 1.1", Software Engineering Institute Technical Report, CMU/SEI-93-TR-25 ESC-TR-93-178, Pittsburgh, PA, 1993.

[9] Reifer, D.J., "Requirements Management: The search for Nirvana", IEEE Software, pp. 45-47, May/June 2000.

[10] Hammer, T.F., Huffman, L.L., "Doing Requirements Right the First Time", CrossTalk , pp. 20-25, Dec 1998.

[11] Wohlin, C., Runeson, P. et al, "Experimentation in Software Engineering An Introduction", Kluwer Academic Publishers, Boston/Dordrecht/London, 2000.

[12] Kruchten, P., "The Rational Unified Process- An Introduction", Addison Wesley Longman Ltd, Reading Massachusetts, May 1999.

[13] Cline, A., "Overcoming Cultural barriers to the adoption of Object technology", in OOPSLA´96 conference proceeding, Atlanta, Oct. 1997.

Author CV

Dr Zaigham Mahmood

Name: Zaigham Mahmood

Email: Z.Mahmood@Derby.ac.uk

Qualifications: BSc, MSc, MSc, PhD, CEng, MBCS

Position: Senior Lecturer, Computing – University of Derby, UK
Since Jan 1991
Currently, Visiting Professor – Int Islamic University, Islamabad, Pakistan
From Jan 2003 – June 2003

Research and Scolarly activity:
10 publications
Numerous book reviews and peer reviews of conference papers
Member Int. Advisory Committee – SQM, INSPIRE, NWTIT

Did you already know?

- How good is an organization at identifying its own strengths and weaknesses?

Jørn Johansen DELTA
Kai Ormstrup Jensen DELTA

Abstract

The BOOTSTRAP assessment method, which DELTA uses, includes collection of the interviewees' perceptions of what the strong and weak processes in their own development process are.

They fill out ten sticky notes each with their spontaneous perception of the process maturity situation - five with the names of the weakest processes and five with the names of the strongest processes. These sticky notes are used to control the interview flow – such as subject and time used per interviewee.

DELTA has performed nearly 60 assessments. The data used for this analysis comes from the 50 assessments performed using BOOTSTRAP 2.3 and is based on 8678 stickers.

The purpose of this paper is to give the main conclusions of this work, and specifically answer the following questions:

- Is an organization able to identify strengths and weaknesses by asking "itself" – or formulated more precisely:
  - Is it possible to identify a correlation between the employees' perception of the strength and weaknesses in the organization and the maturity of the processes?
  - Is it possible to identify a correlation between the employees' perception of the strength and weaknesses in the organization and the recommendations for improvement identified in the assessment?
- Is there any difference between the way management and the project members perceive the situation?

The answers to these questions are: It depends on the process area and on how involved the employees are in the particular process. The article goes into details with the data analysis and gives the argumentations for this statement.

This work is performed in the project Talent@IT, which is a large 30 person year Danish SPI research project, which will run for the next 3 years. The project is funded in part by the Danish government - the Ministry of Science, Technology and Innovation.

Keywords

CMM, BOOTSTRAP, Process Improvement, Organizational Knowledge Improvement.

1 Introduction

This work is a part of the project Talent@IT, for which the goal is to develop a model and a corresponding method to assess an organization's capability in relation to process improvement and innovation. Talent@IT is a so called centre contract – a co-operation between the IT-University at Copenhagen, PBS, Danske Bank, ATP-Huset, SimCorp, and DELTA – with support from the Ministry of Science, Technology and Innovation. In this work several different types of data collection and analysis are evaluated. This paper is an example of this work. This paper is the first presentation of results from this analysis. Further analysis will hopefully bring new results.

The assessments are performed mainly in organisations in Denmark but also organisations in other parts of Europe and North America have been assessed. The assessed organisations are from many different organisational units. Approximately half of the assessed organisations are producing administrative software, the other half are producing embedded software.

We would like to thank Anne Mette Jonassen Hass for her indispensable database support and Robert Olesen for use of his statistical competence to verify the significance of the results. Without this support, this paper had ended up in data chaos and without statistical solidity. The authors also wish to thank Anne Mette Jonassen Hass, Robert Olesen and Ole Andersen for valuable reviews, comments and discussions.

2 Introduction to the BOOTSTRAP Model and Method

The basis for this work is generated during use of BOOTSTRAP version 2.3. Because of this, a brief introduction is necessary.

BOOTSTRAP version 2.3 consists of an assessment method and a model for software development activities. The model is shown in Figure 1 below. The BOOTSTRAP method involves a preliminary meeting, an assessment week, and delivery of an assessment report.

The aim of the preliminary meeting is to give assessors an overview of the organization and to ensure management commitment to the assessment and the SPI activities that will follow. During the assessment week, the assessment team—consisting of at least two certified assessors—spends three to four days interviewing key people in the organization. Although we call it a "week," the actual length of this assessment period varies depending on how many projects we're assessing. On the average, we assess management and four projects, but we might assess more (or fewer) depending on the nature of the organisation's typical projects.

The assessment week starts with an opening meeting that offers assessment participants a general introduction to the BOOTSTRAP model and method. The goal of the meeting is to give all participants the same starting point. We then begin the interviews. We base our detailed assessments on two questionnaires: one for the Software Producing Unit, or SPU (typically the organization's management); and one for the project teams. The first part of a questionnaire asks general questions and the second part covers the Bootstrap model and features about 200 detailed questions.

The assessment week concludes with a meeting in which we present the preliminary results of the assessment. We then further analyse the results and deliver a final report.

The BOOTSTRAP method includes an algorithm for generating detailed maturity profiles from the completed questionnaires. The result is a set of profiles that rates the maturity on a scale from 1 to 5, with a precision of 0.25 for each of the process areas defined in the model. We produce a set of profiles for the SPU and a set for each of the assessed projects. The SPU set of profiles shows management's support for the projects that they are ultimately responsible for. The second sets of profiles show the practices in the projects. Using these maturity profiles, we can identify strengths and weaknesses for the overall organization, as well as the areas that most urgently need improvement. Our suggestions for improvement areas are also based on additional information such as benchmarking against similar organisations, and the nature of the target organisation, its products, and the market. The final assessment report contains a list of the top four to seven improvement areas and a preliminary plan for implementing improvements in the organisation.

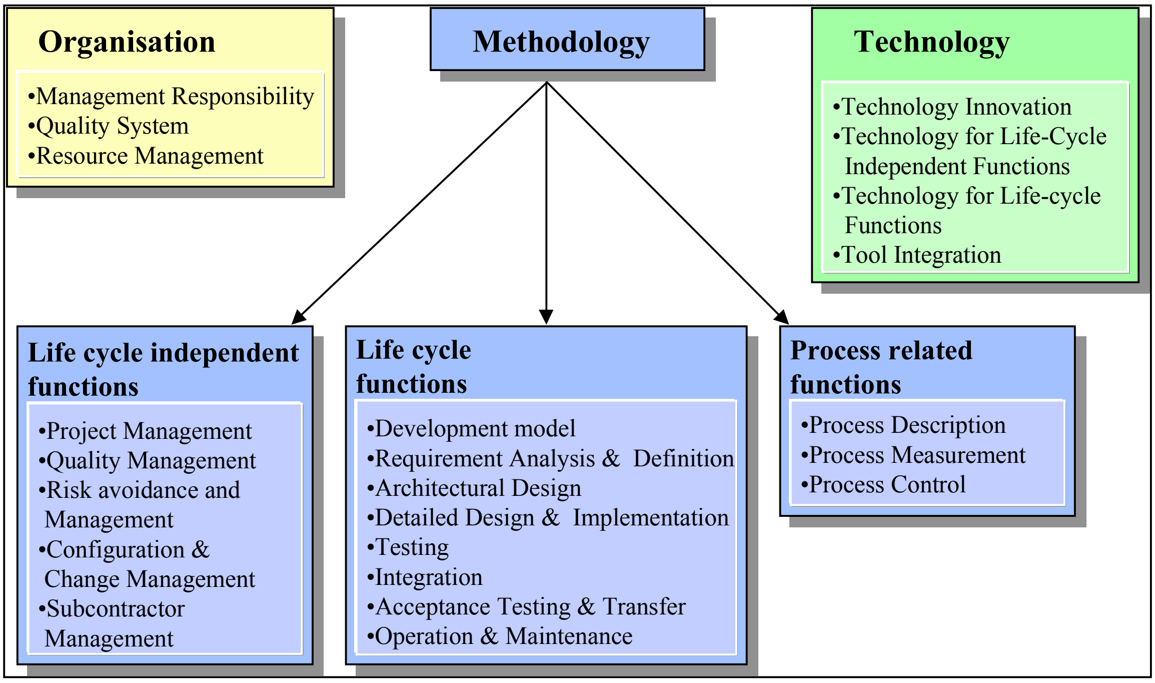

Organisation
•Management Responsibility
•Quality System
•Resource Management
Methodology
Technology
•Technology Innovation
•Technology for Life-Cycle Independent Functions
•Technology for Life-cycle Functions
•Tool Integration
Life cycle independent functions
•Project Management
•Quality Management
•Risk avoidance and Management
•Configuration & Change Management
•Subcontractor Management
Life cycle functions
•Development model
•Requirement Analysis & Definition
•Architectural Design
•Detailed Design & Implementation
•Testing
•Integration
•Acceptance Testing & Transfer
•Operation & Maintenance
Process related functions
•Process Description
•Process Measurement
•Process Control

Figure 1: The BOOTSTRAP model. The categories shown in this figure are presented in the following figures with abbreviated names. Life cycle independent functions are referred as "Support", Life cycle functions are referred as "Engineering" and Process related functions as "Process".

2.1 The interview Form

The interview form is important in relation to this paper. It includes gathering of the interviewees' spontaneous perception of the process maturity situation – and this is the basis of the analysis performed in this paper.

During the opening meeting, the assessment participants are provided with the detailed information about the processes in the maturity model, such as substance and requirements in relation to each maturity level.

An interview starts with a short presentation of each person and some questions from the general questionnaire. After this, each person gets ten sticky notes, which they use as follows:

- They write their name on each sticky note
- On five of the sticky notes, they write a minus (the minus stickers), and on the other five sticky notes they write a plus (the plus stickers)
- On each of the five minus stickers they write the name of one of the five processes that they feel are the weakest processes or perhaps even missing processes – seen from their perspective
- On each of the five plus stickers they write the name of one of the five processes that they feel are the strongest processes – seen from their perspective

During this work the participants are not allowed to discuss the processes. The only help is a short description of the model and processes, including keywords and definitions – and of course the information given at the opening meeting. This exercise takes approximately 10 minutes.

The sticky notes are grouped and placed at a whiteboard. This gives the participants in the interview (including the assessors) a first visual impression of the employees' own feel for the situation (i.e. which processes have most attention in positive and negative direction).

From the whiteboard the interviewing assessor now picks the sticky notes pertaining to one process and address these. This technique makes it possible to control the interview (bring all the interviewees in play, and divide their speaking time equally). It also gives the participants an overview of how the assessment proceeds and keeps the relation to the maturity model.

3 Data Background

Three types of data are collected and analysed:

- The interviewees' perception of the processes provided via the stickers.
- The assessment results presented as maturity profiles and one overall maturity level rating.
- The process improvement recommendations given by the assessors.

In the following sections this data background is presented.

3.1 Stickers

As mentioned above the BOOTSTRAP assessment method, which DELTA uses, includes collection of the interviewees' perceptions of which the strong and weak processes in their own development process are – using the five plus stickers and the five minus stickers.

The stickers were produced immediately after the presentation of the processes defined in the BOOTSTRAP 2.3 model. Because of this, each sticker refers directly to a BOOTSTRAP process and therefore it is interesting to see if there is a correlation between the interviewees' perceptions and the maturity as assessed by the BOOTSTRAP assessors.

The stickers from the assessments are the ones that are used for the analysis in this paper. The analysis is based on stickers from 1061 people that participated in 241 group interviews. 50 of these interviews were with management groups, called SPUs, in as many organisations. In these 50 organisations 191 group interviews were performed with representatives from individual projects in the organisations producing software products.

At the 241 interviews the 1061 people have produced 8678 stickers identifying processes that they feel work well in their organisation (4541 plus stickers) and processes that don't exist or need to be improved (4137 minus stickers).

The number of stickers is too small, if everybody is to produce 10 stickers – this is mainly due to the fact that stickers mentioning the BOOTSTRAP process areas under Technology (see Figure 1) are excluded.

	No. of interviews	No. of people	Plus Stickers	Minus Stickers	Total Stickers
SPU's	50	261	1180	1046	2226
Projects	191	800	3361	3091	6452
Total	241	1061	4541	4137	8678

Figure 2: Scope of the analysis

After each interview session the BOOTSTRAP assessors calculate a maturity profile, i.e. a maturity level for each of the process areas as well as an overall maturity level, based on the BOOTSTRAP algorithms.

Combining the stickers with the maturity levels for projects and SPUs makes it possible to calculate the frequency of plus and minus stickers related to the individual process areas. These calculations will be made for selected process areas based on data where the process areas have been recommended for improvement and based on data where the process areas not have been recommended.

3.2 Maturity Level

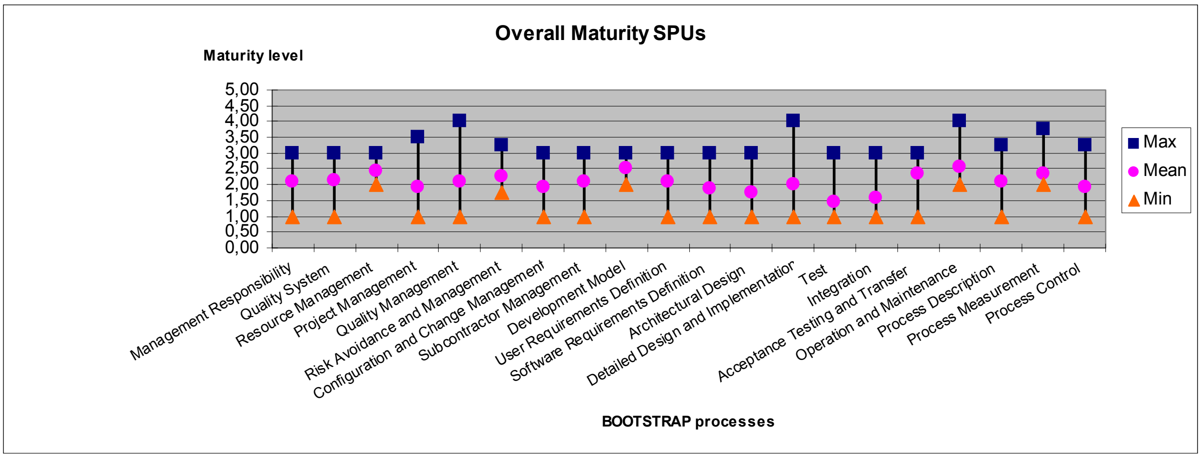


Figure 3 Overall BOOTSTRAP Maturity of all SPUs.

There is a large variation between the maturity levels of the organisations included in this investigation. This is because DELTA as an independent organisation has performed BOOTSTRAP assessment for many different organisations. Projects and SPUs with maturity levels from 1.00 to 3.75 are represented in this investigation. The maximum, mean, and minimum scores for all process areas in the BOOTSTRAP model are shown in Figure 3 for all the SPUs.

The maximum, mean, and minimum scores for all process areas in the BOOTSTRAP model are shown in Figure 4 for all the projects.

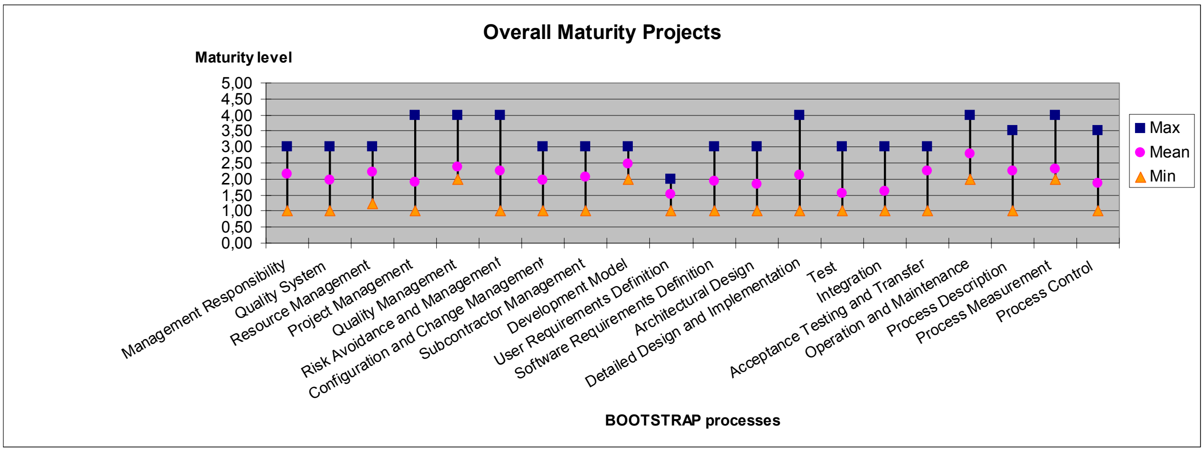


Figure 4: Overall BOOTSTRAP Maturity of all projects

3.3 Recommendations

As a part of each BOOTSTRAP assessment, a number of process areas are selected and recommended for process improvement. The number of recommended process areas for an organisation varies between four and seven depending on the organisation's ability to handle process improvement and the resources needed to improve each process.

Based on the 50 BOOTSTRAP assessments performed by DELTA 294 recommendations were given in total. Of these, 167 recommendations or 57%, were in the following 6 areas:

Process name	No. of Recommendations	Recommended to
Project Management	38	76%
User and Software Requirement Definition	31	62%
Configuration and Change Management	28	56%
Test	26	52%
Quality System	24	48%
Development Model	20	40%

Figure 5: BOOTSTRAP Recommendations

Project Management has been recommended 38 times or to 76% of the assessed organisations. The process areas User Requirement Definitions and Software Requirement Definitions are treated as one recommendation here, because these process areas are very closely related and it can be very difficult to differentiate between them in ‘real life’.

4 Data Analysis

After several discussions and trials we found it suitable to sort the data in groups according to the maturity levels achieved for the assessed process areas. An overview is shown in Figure 6 below.

Maturity level 1.0 forms one group because it is the lowest possible (=no maturity). Maturity levels 1.25 to 1.75 form the next group because they signify some maturity (different from nothing). The next group is formed by maturity levels 2.0 to 2.75 (more maturity), and the last is formed by maturity levels 3.0 and above (none have reached more than 4, through 5 is the theoretical maximum).

Group	Min. maturity level	Max. maturity level
1	1.00	1.00
Below 2	1.25	1.75
Below 3	2.00	2.75
3+	3.00	4.00

Figure 6: Groups of maturity levels

Several hours were used analysing different distributions of the stickers, leading to the main conclusion, that it was necessary to investigate the distribution of the stickers in relation to the number of people able to produce a plus or a minus sticker for a particular process area. This made it possible to provide a picture of the plus and minus “vote” frequency for a given process area in a given maturity group.

In addition to this grouping, it was also decided to group in relation to:

- Management (SPU: Software Producing Unit) versus Projects
- Improvement recommended versus improvement not recommended

4.1 Distribution per Process Area

It was now possible to produce a distribution graph for each process area and for six different groups of data, namely:

- All SPU data
- All Project data
- SPU data where improvement is recommended
- Project data where improvement is recommended
- SPU data where no improvement is recommended
- Project data where no improvement is recommended.

This gives 120 graphs in total.

Figure 7 below shows the principle of the graphs. The maturity group defined in Figure 6 is used as the x-axis. The frequency for the interviewees' stickers within each maturity group is shown on the y-axis. The frequency is calculated for plus, respectively minus stickers as the percentage of available stickers used for the given process area. I.e., if 100 people participated in interviews, 500 plus stickers were available. If 25 of these named Project Management, then the frequency for positive rating of Project Management is 0.05. Likewise 500 minus stickers were available, and if 200 of these were stating Project Management, then the frequency for negative rating of Project Management is 0.4.

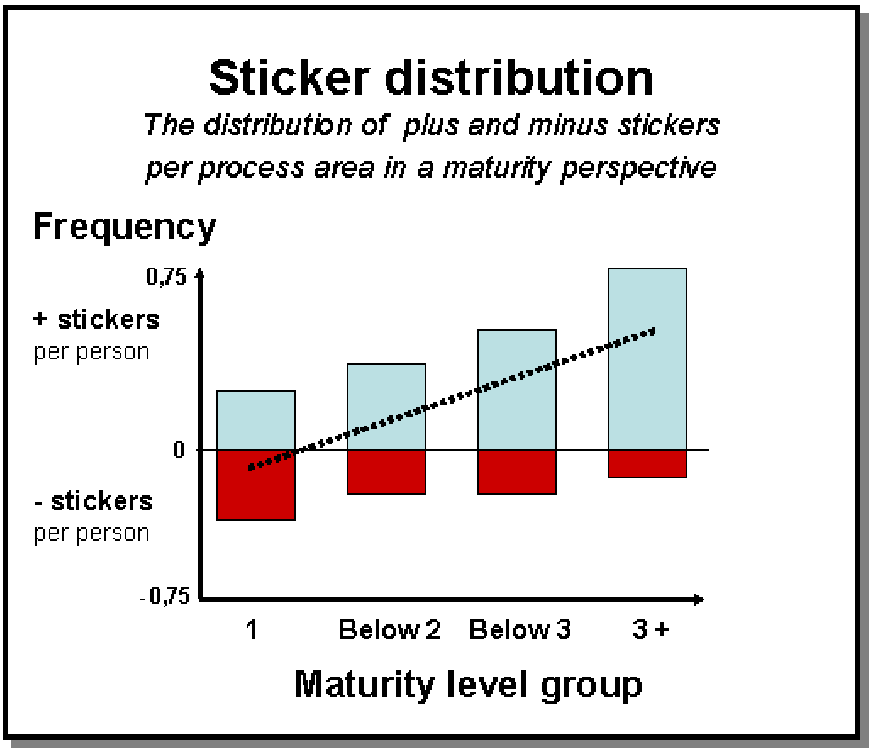


Figure 7: The frequency distribution of plus and minus stickers per maturity level group for a particular process area. The dashed line illustrates the slope and intercept – this is used in next section.

In Figure 8 below, four different examples of these graphs is given.

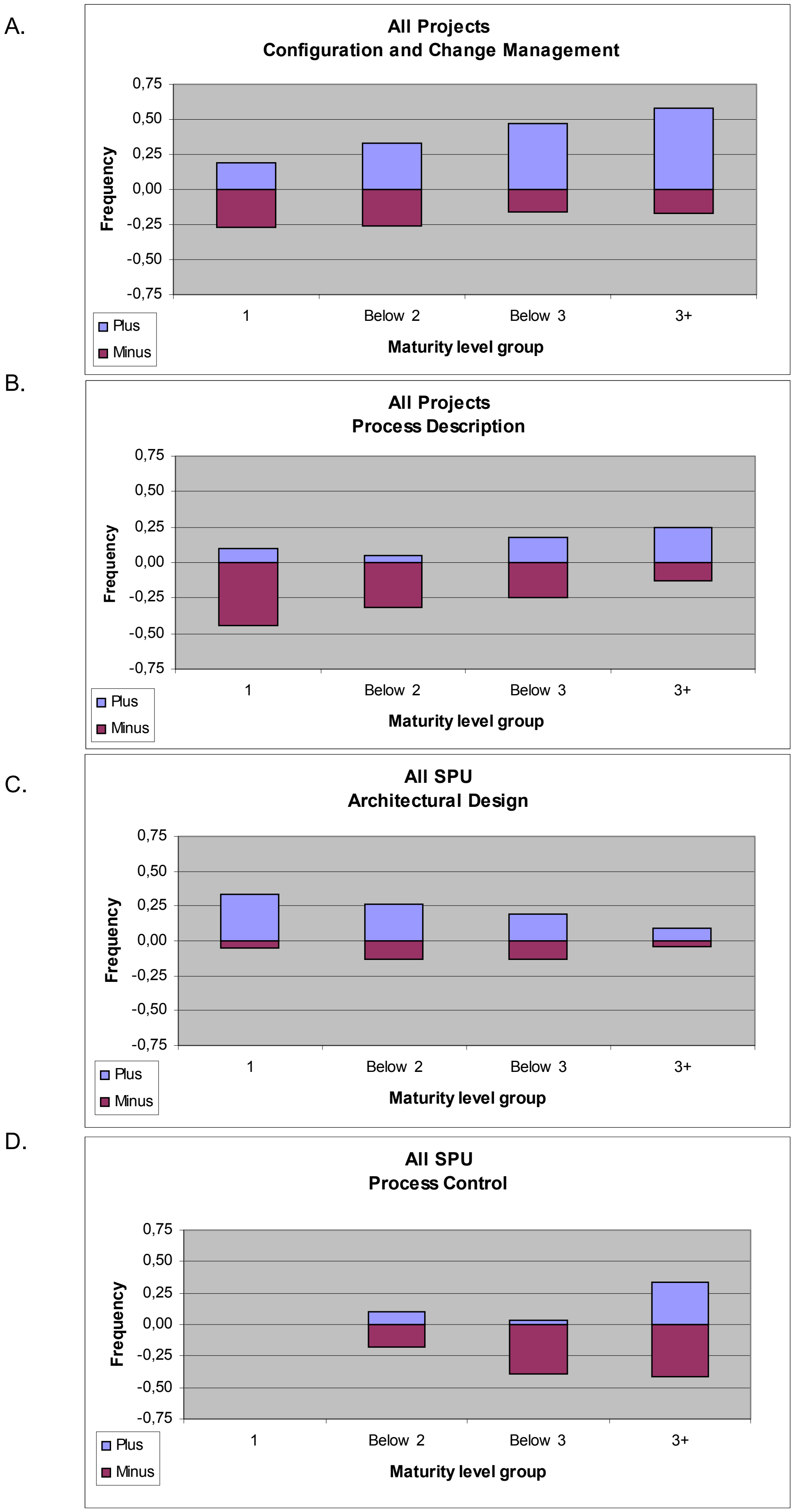


Figure 8: Frequency of plus and minus stickers per maturity level group per process area.

In Figure 8 A**,** we examine the entire plus and minus stickers mentioning the process area Configuration and Change Management in project interviews (whether this process was recommended for improvement or not). The distribution per maturity level group is shown in the table in Figure 9 below.

So - for this process area it can be concluded, that the higher the process maturity level is, the higher is the positive perception of the process area (higher frequency of plus stickers and less frequency of minus stickers).

Group	**% of the available plus stickers given**	**% of the available minus stickers given**
1	26	30
Below 2	48	17
Below 3	55	10
3+	45	29

Figure 9: Distribution of plus and minus stickers per maturity level group for the process Configuration and Change Management for all stickers in project interviews. The data correspond with Figure 8 A.

The same pattern is seen in Figure 8 B for the process area Process Description – also here the stickers are for all project interviews. However, there is a difference from Figure 8 B – the perception of this process area is in general less positive (the distribution is displaced to a lower perception level with more minus stickers and less plus stickers). However, the employees also have higher positive perception of the process area the higher the process maturity level is – but the perception starts at a more negative level.

In Figure 8 C, we examine all the stickers naming the process area Architectural Design in interviews with management (SPU) (whether this process was recommended for improvement or not). This is a different picture. Here the Management have a lower positive perception of the process area the higher the process maturity level is. This could be because of more process knowledge (substance and actual state) the higher the maturity level.

And finally, in Figure 8 D we examine all the stickers naming the process area Process Control in interviews with management (SPU) (whether this process was recommended for improvement or not). This shows the same pattern as in Figure 8 C. Here the Management also have a lower positive perception of the process area the higher the process maturity level is – but it starts at a more negative level.

Individually and in combinations, all these 120 graphs tell a lot – it is, however, difficult to reach an overall conclusions. Because of this we developed another graphical illustration with the purpose of getting a more useful overview. This is shown and used in the next section.

4.2 Applicability of Self-perception

The idea is an illustration for the six groups of data mentioned above, in two dimensions. The two dimensions are:

1. Perception of the process area.

2. Agreement with maturity level.

If we examine the graphs in the previous section, the perception could be expressed as a function of majority of positive stickers, and the agreement as the increase in positive perception with increase in maturity level. After several trials and discussions we found a strong correlation between this and the slope and intercept, respectively, when linear regression was used on the data.

We ended up with the following algorithm:

1. Calculate a point for each quartile (a 0.25 step in maturity) as the number of positive stickers minus the number of minus stickers, and offset these points with - 2.25, because this is the mean maturity level for both SPU and projects.
2. Use these points to calculate the slope and interception using linear regression.
3. Map the process area by (Perception = Intercept, Agreement = Slope) in the coordinate system

This gives the possibility of plotting the data in the two dimensions, and this gives four overall types of combinations as shown in Figure 10 below. If we use the algorithm on the data presented in the graphs in Figure 8, there will be one in each quadrant. The quadrants are indicated with a letter (A to D) corresponding to the individual graphs in Figure 8.

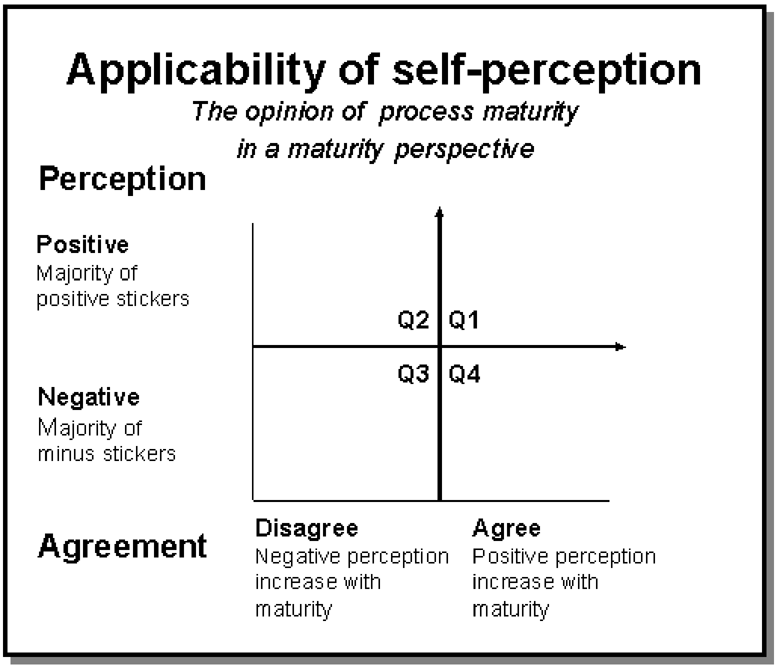

Applicability of self-perception
The opinion of process maturity
in a maturity perspective
Perception
Positive
Majority of positive stickers
Negative
Majority of minus stickers
Q2
Q1
Q3
Q4
Agreement
Disagree
Negative perception increase with maturity
Agree
Positive perception increase with maturity

Figure 10: Illustration of data in relation to applicability of self-perception. A to D correspond with the individual graphs in Figure 8.

Characteristics for the four quadrants:

- **Q1 - Positive Perception and Agree**: The process areas here are generally perceived as relatively strong, and the positive perception increases with higher maturity (ex. Figure 8 A).
- **Q2 - Positive Perception and Disagree**: The process areas here are generally perceived as relatively strong, and the negative perception increases with higher maturity (ex. Figure 8 C).
- **Q3 - Negative Perception and Disagree**: The process areas here are generally perceived as relatively weak, and the negative perception increases with higher maturity (ex. Figure 8 D).
- **Q4 - Negative Perception and Agree**: The process areas here are generally perceived as relatively weak, and the positive perception increases with higher maturity (ex Figure 8 B).

In the following we represent the six groups of data processed in this way, and make some comments on this.

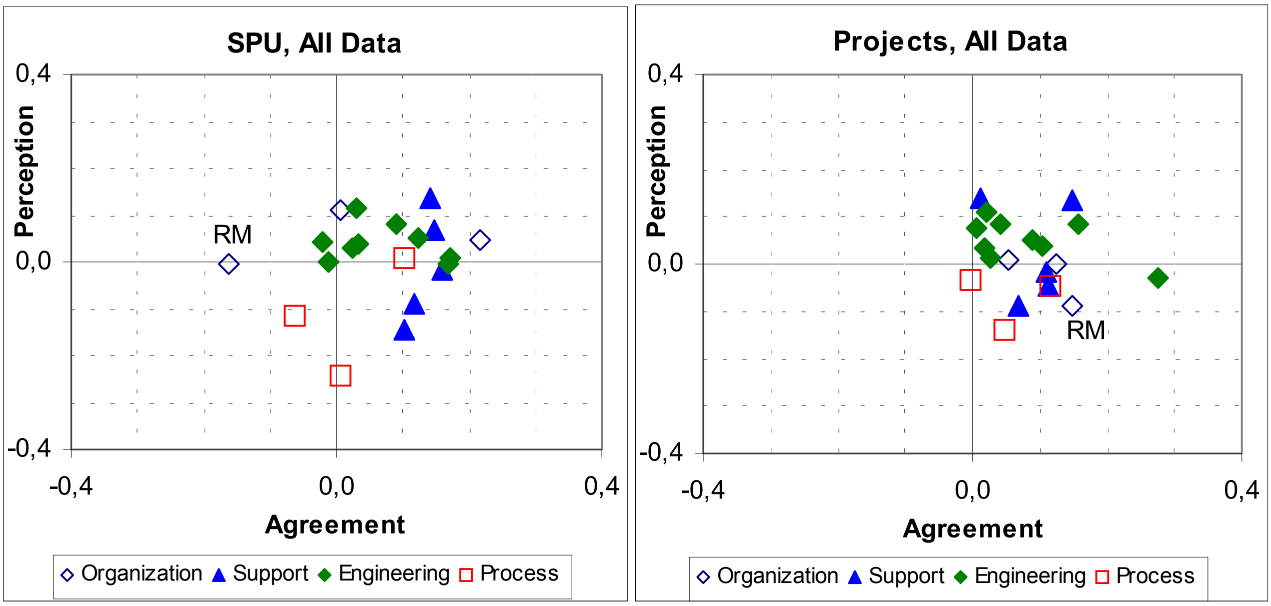


Figure 11: Perception / Agreement for All SPU data and All Project Data.

From Figure 11 several observations are made:

- There is a majority of Engineering process areas above zero in relation to Perception for both SPUs and Projects. The reason could be that these process areas are more visible and more used – especially for the employees in projects.
- The Process areas are in general below zero in relation to Perception for both SPUs and Projects. The position structure is the same for both maps – Process Control to the left, Process Description to the right and Process Measurement at the lowest (this is the same for the following maps, except SPU No recommendation, where Process Control and Process Description are swapped). The explanation could be that these process areas normally are perceived as difficult process areas that depend on other processes and a certain level of maturity.
- The Support process areas are all placed to the right – agreement above zero. They are seen as process areas that it makes good sense to strengthen in relation to getting a higher maturity. Expressed in another way – it is easy to be convinced that Configuration and Change Management, Quality System, Risk Avoidance and Management, Project Management, and Subcontractor Management are important in relation to a higher maturity level.
- The Organisation process areas are much more distributed for SPUs than for projects and the perception is more positive seen from a SPU perspective than from a project perspective. Resource Management (marked RM) is remarkable – Managements (SPUs) do not see this process area as one that will increase maturity if improved. This is not what is concluded for projects (marked RM) – projects see this process areas as one where they agree that a higher maturity will result if improved, and it starts at a very low level. Also the two process areas Management Responsibility and Quality System are seen as more positive from a SPU perspective than from a project perspective.
- Finally we observe that the process areas generally are more distributed all over the map for SPUs than for projects and the agreement in the value of improvement is more distinct for projects than for SPUs. This could be because of more direct “contact” with the processes in projects than in SPUs and therefore more knowledge about the substance of the processes.

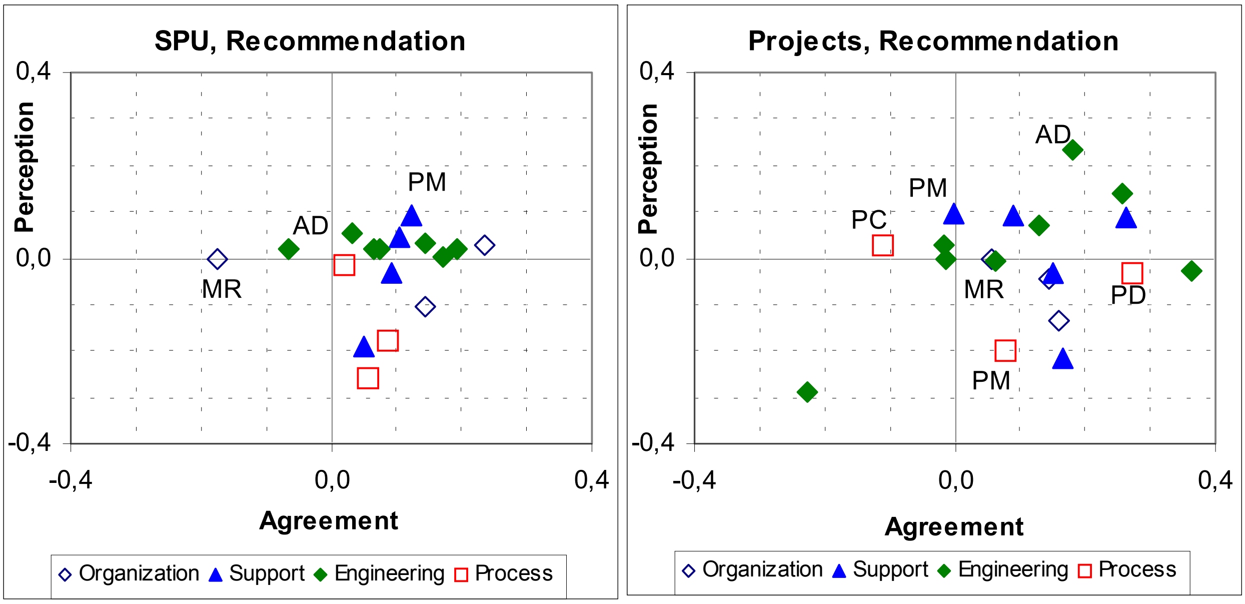


Figure 12: Perception / Agreement for SPU and Projects Recommendation Data.

If we look at the perception and agreement in relation to processes, which has got a recommendation for improvement, there are differences and similarities in relation to All data commended above:

- Also for this set of data the engineering processes are mainly placed above zero in relation to Perception, especially for SPUs, and less characteristically for Projects, for which they are more spread over the map. The main difference between SPUs and Projects is the process Architectural Design (marked AD). The reason for this difference in perception could be that the management assume this as a necessary basic process at low maturity levels (many plus stickers), but with higher maturity comes recognition of a less mature process. Projects know it is an important process but overrates their performance.

- Process processes has nearly the same place in the map as for All Data. As mentioned above Process Description and Process Control is swapped for SPUs. The processes are much more spread for Projects than for SPUs, which could be because SPUs in general see these processes as being at a low maturity level and agree it is necessary to improve them, but not necessarily as the first thing. The Projects feel a need for process description (marked PD), have the same perception as SPUs about Process Measurement (marked PM square), and see Process Control (marked PC) as a process for implementation in the future.

- The support processes are placed in the same quadrants, which indicate an agreement between SPUs and Projects for these processes. Project Management has the most obvious difference in placement – for SPUs the right-most and highest placed process, and for Projects the highest placed process at the y-axis (Subcontractor Management has a larger difference, but there is too few data for conclusions). The Perception is nearly the same, but the agreement differs. This difference could be because the projects in all maturity groups overrate this process and therefore get a slope (agreement) which is nearly zero. SPUs have a lower positive perception at low maturity. If this is the case, the reason is that the Project employees convey too positive a picture of their project manager during the assessment (especially at the low maturity levels).

- Here Organisation has the same reasoning as above with one difference. The process Management Responsibility – here with recommendations for improvement – is now the left-most process in the SPU map, and the left-most organisation process in the Project map. The conclusion must be that the managers convey a too positive picture of their own process, something that decreases with higher maturity.

- The difference in distribution is the opposite of what we saw for All Data. Here the Projects distribute the processes over the map, more than the SPUs do. The reason for this could be that the projects have the most firm knowledge about the processes, where the managers are more diverging in their rating, something that gives a less dynamic picture when the data are averaged.

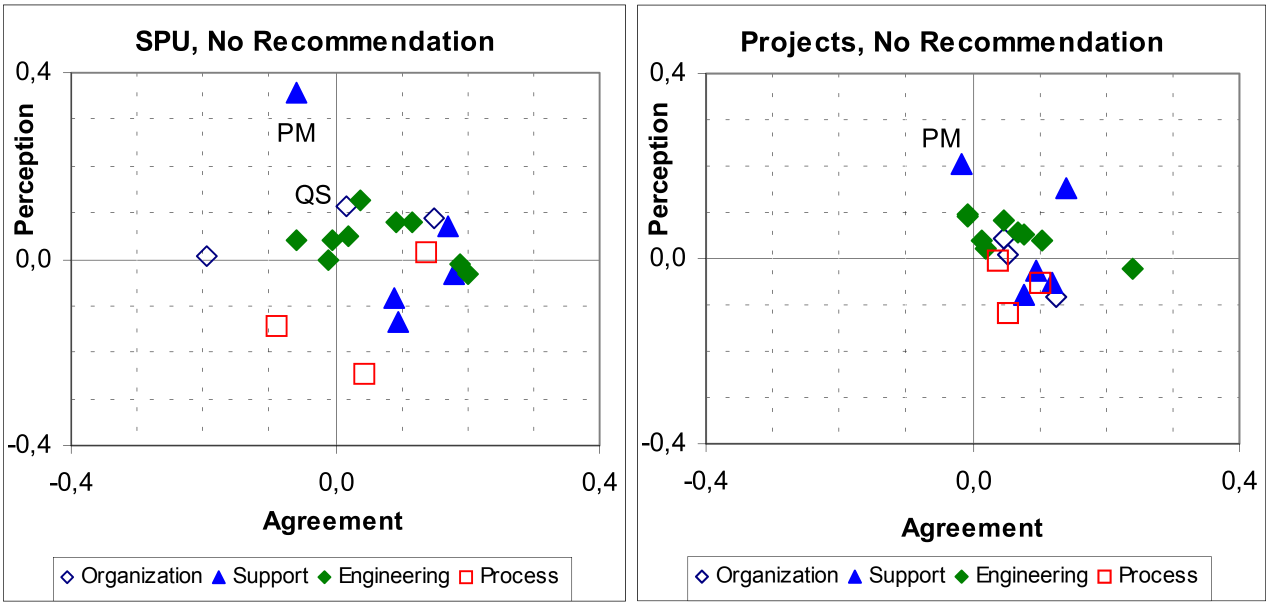


Figure 13: Perception / Agreement for SPU and Projects No Recommendation Data.

Here we look at the perception and agreement in relation to processes that have **not** got a recommendation for improvement. In relation to the situation with recommendations there are the following obvious differences and similarities:

- The Engineering processes are also for this set of data mainly placed above zero in relation to Perception for especially SPUs, but it is also characteristic for Projects, though these generally have a lower positive position and are more distributed over the map. It is remarkable that it is the Architectural Design and Detailed Design & Implementation that are placed together most to the left – that is what we are good at, and it will not help to improve these processes.

- The Process processes have their "usual" place in the map.

- Also here there is a complete agreement in the position of each Support process area, though the exact placements differs a little. The noticeable process is Project Management, which for both SPUs and Projects is placed with the highest perception to the left (marked PM). This process is perceived as very good, and there is no agreement for further improvement – neither is there recommendations for improvement from the assessors.

- For the Organisation process areas we have the same centralisation for the SPUs as for All data (Figure 11), and nearly the same for Projects (Quality Systems has moved a little to the left and up – marked QS). No further discussion.

- It is interesting, that here it is the SPU process areas that have the largest distribution – the opposite of the situation from when we looked at the data for process areas where recommendations are given. The reason for this could be (as above), that the projects have the firmest knowledge about the processes. We are now discussing process areas, which are not recommended for improvement, and therefore the employees in the projects are less uniform in their ratings, something that gives a less dynamic picture, when the data are averaged. The managers are, opposite to this, more uniform in their ratings.

5 Conclusions

If we start with the questions addressed in the abstract, the overall answers are that the organisation is able to identify strength and weaknesses by asking “itself”. This is shown by the intensity of process areas with an “agreement” above zero compared with a lower intensity below zero.

But be aware: the more involved in the process the interviewed person is, the more positively biased the answers will be. The engineering process areas are rated more positively by the projects, just like the organisational process areas are rated more positively by the managers. Add to this that process areas that have low involvement are rated most negatively – for instance the Support process areas Quality Management, Risk Avoidance and Management, and Subcontractor Management.

Yes - it is also possible to identify a correlation between the employees’ perceptions of the strength and weaknesses in the organisation and the recommendations for improvement identified in the assessment. This is done by looking at the average placement of the process areas for SPUs and projects for process areas with and without recommendations given, respectively. The average is lower in relation to positive perception in the group with recommendations (Figure 12 versus Figure 13).

And finally: yes - there are differences between the way management and the project members perceive the situation, especially when we look at the Organisation process areas and some Support process areas.

There are a few other important conclusions that this data analysis opens up for.

If the question was “Did you already know enough?” the answer would have been **no**. If we look at the process areas most often recommended for improvement (listed in Figure 5), they are all placed on the map in quadrant 1, where process areas which generally are perceived as relatively strong, and where the positive perception increases with higher maturity are placed. Although these process areas are perceived as good they are recommended for improvement – something that the interviewees normally agree to at the end of an assessment. This means that the participants during the assessment recognize the need for improvement, and it is the assessment process itself that gives this recognition.

We can add to this that the process areas in this quadrant, are the process areas from the Engineering process area and those from the Support process area, i.e. the process areas closest to the project work. This leads to the conclusion, that the pattern has something to do with a lack of craftsmanship seen from a maturity perspective. A reason for that could be that the education in these skills (internal in the organisations and external at universities as well) is too superficial.

An final conclusion is (the good tidings), that if process areas are improved, then this is actually perceived as an improvement – the process areas are rated higher by the employees working with the process areas. This is the absolute best proof of the maturity model concept.

6 Literature

Anne Mette Jonassen Hass, Jørn Johansen, Ole Andersen (1996). Softwareudvikling i elektronikindustrien. DELTA rapport, D-260.

Mathiassen, L., Pries-Heje, J. and Ngwenyama, O. (2002). Improving Software Organizations: From Principles to Practice, Addison-Wesley.

Palk, M. C., Curtis, B. and Chrissis, M. B. (1995). The Capability Maturity Model: Guidelines for Improving the Software Process, Addison-Wesley.

Pasi Kuvaja, J. Simila, L. Krzanik, A. Bicego, S. Saukkonen, G. Koch (1994).
Software process assessment and improvement: the BOOTSTRAP approach.
Blackwell.

Watts s. Humphrey, Bill Curtis. Comments on ‘A Critical Look’. IEEE Software, July 1991.

7 Internet references

Talent@IT: http://www.talent-at-it.dk/

DELTA: http://www.delta.dk/

8 Author CVs

Jørn Johansen, DELTA

Jørn Johansen is the manager of the Software Technology Department at DELTA. He has an M.Sc.E.E. from Ålborg University and has more than 23 years experience in IT. He has worked for 15 years in a Danish company with embedded and application software as a developer and project manager. Mr. Johansen has been involved in all aspects of software development: specification, analysis, design, coding, and quality assurance. Furthermore he has been involved in implementation of an ISO 9001 Quality System in the company, and was educated to and functioned as internal auditor.
For the last 8 years he has worked at DELTA as a consultant and registered BOOTSTRAP lead assessor and for the last 4 years he has been the department manager. He has participated in more than 30 BOOTSTRAP assessments in Denmark and abroad for companies of all sizes. He was the project manager in the Danish Centre for Software Process Improvement project, a more then 25 person-year SPI project and is currently the project manager of a new Danish SPI project: Talent@IT. This is a 26 person-year project that involves 4 companies, the IT University in Copenhagen and DELTA. Mr. Johansen is also the co-ordinator of a Danish knowledge exchange group: Improving the Software Development Process, which is the Danish SPIN-group.
Jørn Johansen, can be contacted at DELTA; Danish Electronics, Light & Acoustics; Venlighedsvej 4; DK 2970 Hoersholm; Denmark; Phone +45 72 19 40 00
Fax +45 72 19 40 01 E-mail : joj@delta.dk World Wide Web : www.delta.dk

Kai Ormstrup Jensen, DELTA

Kai Ormstrup Jensen has a B.Sc. E.E from the Engineering College of Aarhus. He is examined assessor in SPICE 3.0, Bootstrap 3.0 assessor and EFQM. He has started the education to become an authorized CMMI assessor and has passed the first examination: ”Introduction to the Capability Maturity Model – Integrated, Staged Representation, v1.1”. He also has fulfilled the requirements of the examination in Software Testing developed by The British Computer Society and certified by the Information Systems Examinations Board. In 1999 he wrote the book “Automation of Software Test”.
He worked for Brüel & Kjær (1982-98) on different projects using integrated product development. The projects he participated in were embedded projects and in these projects he worked primarily with software and digital hardware. In 1986 he became software project manager with responsibility for the software functionality and interfaces to other parts of the product.
From 1995-98 he participated in two ESSI PIE’s. The first project was about reduction of software errors using experience-driven test efforts. The other project was about reduction of errors introduced in the requirement phase of the project.
In 1998 he came to DELTA where he is currently working as a consultant inside the area of software testing and test improvement. He also participates in software and hardware process assessment, process improvement, effective and efficient product development etc.
Kai Ormstrup Jensen, can be contacted at DELTA; Danish Electronics, Light & Acoustics; Venlighedsvej 4; DK 2970 Hoersholm; Denmark; Phone +45 72 19 40 00
Fax +45 72 19 40 01 E-mail : koj@delta.dk World Wide Web : www.delta.dk

An Assessment Approach for Software Workforce in an Emergent Organization

Ö. Özgür Tanriöver,
Assoc. Prof. Onur Demirörs
Informatics Institute
Middle East Technical University,
İnönü Bulvarı, 06531, Ankara, Turkey
Phone: +(90 312) 210-3741, **Fax:**+(90 312) 210-3745

Abstract

In this paper, we propose a Software Workforce Assessment Model based on practices and roles defined in SW-CMM (Software Capability Maturity Model) [Paulk 1995] and knowledge units defined in SWEBOK (Software Engineering Body of Knowledge) [Hillburn et al. 1999]. The purpose of the model is to complement the traditional assessment models by providing an assessment approach for the workforce of software organizations. The applicability of the proposed method is shown by means of a case study in an emergent software organization.

Keywords

Software Process Assessment, Human Factors, Software Process Imrovement

1 Introduction

During last decade, software process assessment methodologies and frameworks have been developed based on their underlying process models, such as in the case of SW-CMM and CMM based assessments or [ISO15504 1996] and SPICE. The results of these assessments show opportunities for improving the process component of the assessed organizations.

However, the most important input of software production is knowledge. [Basili 1983] show that the level of talent of the software practitioners is the most important factor for the success of a software project, besides, the performance of software engineers varies 1 to 20. Hence, assessing the process alone is not sufficient for software organizations.

The aim behind the proposed model is to complement the current software process assessment methods as well as people capability maturity models (P-CMM [Curtis] and TSPi [Humphrey]) as none of them focuses on role based personnel assessment.

The proposed model is based on general evaluation and assessment principles [House 1980], [Scriven 1991], [Juan et al. 2000] which are widely used in education for more than a half a century. Examining various examples reveals five main steps to design an evaluation study:

1. Determination of the *Object* under evaluation.
2. Elicitation of the *Characteristics/Criteria* of the object.
3. Determination of the ideal characteristics of the object to be compared which results the *Assessment Baseline.*
4. Selecting or developing the A*ssessment Technique* to be used for collecting information about the actual object.
5. Developing *Evaluation Techniques* for comparison in order to organize and synthesize the information.

We have used similar steps in developing the Software Workforce Assessment Model. In the next section we summarize the assessment model based on the evaluation steps. Then we describe the case study we performed in an emergent software organization as well as the results we obtained. We conclude the paper by providing lessons we learned during the case study.

2. Assessment model

2.1 The Object

The object of a software practitioner assessment method is the *software practitioner*. We define software practitioner as somebody who is performing some of the activities needed to produce a software system, organized according to a particular organizational structure and equipped with knowledge and skills to perform those activities.

This definition although delimits the software practitioner from a higher abstraction level a more detailed and explicit delimitation is required. In order to do this we use two reference models. SWEBOK [Hillburn et al.1999] explicitly defining the sub knowledge components of the software engineering domain which delimits ‘who is software practitioner in terms of knowledge profile’ and SW-CMM [Paulk et al.1995] explicitly defining extend of the activities, that the software practitioner performs which delimits ‘what the practitioner do’.

However, [Brodman et al 1994] has shown that the CMM model that exists currently is not adapted to the situations in small organizations considering numerous roles proposed in level 2 practices. For this reason we propose roles associated with the level 2 and SPE process area of level 3 because it includes the fundamental activities regardless of the maturity of the software organization. Table 1 shows the occurrences of these software roles in these process areas.

Table 1: Roles and their occurrences in cmm L-2 and SPE key process areas

SOFTWARE ROLES	SOFTWARE ENGINEERING (SE)	SYSTEM ENGINEERING (SYS E)	SYSTEM AND ACCEPTANCE TESTING (SAT)	CONF□GURAT□ON MANAGEMENENT (CM)	SOFTWARE PROJECT MANAGEMENT (PM)	SOFTWARE QUALITY ASSURANCE (SQA)	SOFTWARE MANAGEMENT (SM)	SENIOR MANAGEMENT
CMM PROCESSES								
Software configuration management	√			√	√	√	√	√
Software quality assurance	√				√	√	√	√
Software Subcontractor Management					√			√
Software project tracking and oversight	√				√	√	√	√
Software project planning	√				√	√	√	√
Requirements Management	√				√	√		√
Software Product Engineering	√	√	√		√	√		√

On he other hand, we limited our scope to Software Management and Software Product Engineering Knowledge Categories (SPE) of SWEBOK. As the CMM, the first reference model, is applicable to any software organization regardless of the application domain, it does not provide any specific activity to domain specific software production. But it refers to some application domain experts or people who are experienced in developing software for example in real time systems or in artificial intelligence. Besides, the Software management and Software Product Engineering Knowledge categories are required regardless of the application domain as defined recurring areas in SWEBOK.

Having planned how these reference models will be used we implemented the second step in which the criteria to be checked for determining the characteristics/criteria of the software practitioners.

2.2 Determination of assessment criteria

Criteria descriptions establish the basis for developing the software workforce assessment model that will be used as the ‘assessment baseline’ and for developing the assessment technique. There are two questions we asked for criteria determination: What the software practitioners need to know? How is the knowledge acquired? For these two questions we propose three issues to consider:

Roles: each practitioner considered should be assessed according to the roles he is assigned to.

Software activity: analysis of the roles delimited with the activities performed in emergent software organizations. This criterion is composed of the practices of level 2 key process areas appended with SPE process area of the SW-CMM

Knowledge and Skill: criteria required to assess the knowledge and skill of the software practitioner. This criterion is developed based on knowledge units described in SWEBOK. Knowledge and Skill criterion is further decomposed to 5 sub criteria as defined in Table 2.

Table 2: Knowledge and skill criteria descriptions

Formal Education	This criterion represents software practitioner's knowledge and skills acquired from the courses he has taken about a subject matter during her/his formal education. (College, University). It should also include the practitioner's skills which might be acquired such as participating in course projects in which specific Knowledge units acquired and used.
Professional Education & Training	This criterion represents software practitioner's knowledge and skills acquired considering what he/she has learned on the job training. This criteria covers on the job training programs and projects, seminars have been prepared or participated.
Personal Education	This criterion covers subject matters or knowledge and skills that the practitioner is personally interested in. It includes reading scientific and popular literature and publications, subscribed news groups, forums, attending workshops and participating in conferences.
Practical Knowledge	This criterion represents software practitioner's knowledge and skills about the tools or methods, standards regarding a specific knowledge area. Normally it also covers the programming languages, and Domain specific knowledge but these are not considered in this methodology.
Need; Willingness to Learn	This criterion is a special purpose criterion which is required to understand the perception of the knowledge area by the individual. It is used to understand if the knowledge area is useful to the practitioner considering the tasks he/she is to perform during her/his current role in the organization?

Criteria Scales:

Having defined the specific criteria we focused on the scales, in order to evaluate the criteria. For the first four criteria we use scale definitions presented in Bloom's knowledge taxonomy as these are widely accepted. Bloom [Bloom 1956] has defined the taxonomy of educational objectives that describes several levels of knowledge, intellectual abilities, and skills that a student might derive from education. The scale we used is adopted from the Bloom's knowledge taxonomy and is applicable to the sub criteria defined in table 2.

Additionally, we use a scale to determine priorities of knowledge units for a given software role. This scale is used for determining the priorities of knowledge units to be improved either from an organizational perspective or a practitioner perspective.

2.3 Assessment baseline

The assessment baseline used in our method is a software practitioner model obtained from the criteria and scales defined in the previous section. This model is composed of a series of tables detailing all the roles in the form [roles [activities performed or responsible for [knowledge units required, datum/information set]]]. This structure corresponds to the three issues: roles, activities, and knowledge units. Each role has an associated table, which delimits the activities for that role and knowledge requirements for that role.

For example, an extract of the model that demonstrates the determination of target knowledge profile of 'Software Engineering' role is shown in Table 3. As it can be traced from the rationale row the contents of knowledge units explained in SWEBOK 1.0 and the keywords of blooms taxonomy is used to determine knowledge levels. The Table 3 shows only SPE2 activity as an example of how the assessment baseline is obtained. The same elicitation process is done for all activities.

All the practices in SW-CMM are investigated to determine the knowledge level requirements of sofware engineering role by checking the existence of the keywords in blooms scale for knowledge levels.

Table 3: An extract of assessment baseline development

Software Engineering Role involves in	SPE 2. The software requirements are developed, maintained, documented, and verified by systematically analyzing the allocated requirements according to the project's defined software process.
Rationale: Keywords in the Bloom's Knowledge Taxonomy	Practitioner should be able to review the software requirements for ensuring all the issues are identified. This requires *application* knowledge of requirements elicitation and Verification and Validation KU's. He should also be able to identify the effective method for requirements analysis and document the rationale which requires *analysis* knowledge of Software Requirements Analysis KU. He should participate in specification of missing requirements which requires *application* knowledge of Software Requirements Specification KU. He should be able to use the tools for Configuration management reflects the changes to requirements when necessary which requires *knowledge* on Configuration Control KU.
Swebok Unit Required	Software Requirements Elicitation: Application level (3) Software Requirements Analysis: Application level (3) Software Requirements Specification: Knowledege Level(2) Verification & Validation: Application level (3) Configuration Control: Knowledege Level(2)

As the result of this process for example software engineering role is found to require analysis knowledge level in requirements analysis, requirements specification, architectural design etc. and comprehension knowledge level in risk management planning, software configuration control and software process engineering knowledge units.

3. *Assessment technique and case study*

In the workforce assessment model the assessment is done with a questionnaire including an observation section and informal interviews with the practitioner. The Criteria scales and the assessment baseline are used to develop the questionnaire. The choice of levels for each of the criteria is formed on the basis of Bloom's knowledge scales. In this way knowledge scores obtained for each criterion reflect the same construct.

Additionally knowledge unit indicators have been included for all KU. Knowledge unit indicaotrs are proposed based on the SWEBOK 1.0 of SEI but they are eloborated and detailed using SWEBOK of IEEE and ACM. The knowledge unit indicators are included in order to extend and explain the knowledge units. These can be considered detailed requirements for each of the knowledge units in question. In this way first the assessment is more complete, second the practitioner will correctly understand the meaning and requirements of each knowledge unit and third it can be used to validate the scores of the criterias provided by the practitioner.

As an assessment reference framework we used the evaluation process principles defined in ISO 15504 for three reasons. First it has become an industrially accepted standard because it is explicit and complete according to general evaluation process definitions. Second it is based on a similar assessment baseline as the software practitioner framework. Third it uses multiple value assessment technique unlike CMM.

3.1 Assessment planning and data collection

According to the software practitioner model the roles and their knowledge requirements are proposed considering the processes defined in SW-CMM for level 2 software organizations. This reposes that the organization under consideration should have already been assessed according to CMM assessment and use processes similar to SW-CMM process model.

After having performed initial meetings with 7 software organizations we decided that the organizations X follows our criteria. The software organization we selected have been rated as level 2 according to an assessment conducted in 2001 and the main characteristics of the organizational struc ture matches with the emergent organizational structure [Paulk 1995]. Furthermore, this organization uses base practices similar to SW-CMM practices and management is willing to reach to 3rd level maturity according to SW-CMM.

The activities we performed in preparing the assessment are: making initial contact and obtaining general information about the organization in order to be sure the organization matches with characteristics of an emergent software organization. Having assured we explained the positive outcomes of this study, we showed them sample questionnaire sheets, role transition tables and knowledge indicator list. Accordingly, we discussed with them about the data collection procedures and importance of the information they will be provided with by participating in this study. This was important in order to gain management commitment.

After having discussed about the timing details and availability of practitioners participating in the assessment we prepared an action plan and sent to the managers for approval. In the mean time we prepared specific assessment tools including sample questionnaire sheets, role transition tables and knowledge indicator list, for this organization.

After the approval, we determined the practitioners of the organization who will be participating to the assessment with the management. We selected a project X in which 11 practitioners took roles. We initially distributed the questionnaires to each of the practitioner and asked them to return them in 7 days. The questioner included 47 knowledge units to be scored according to 5 criteria based on 0-5 scale and 210 indicators to be checked for existence. After having collected the assessment forms we have checked each data item for completeness and verified that the knowledge unit indicators support the ratings provided by the individual who participated in the assessment. We have also discussed these issues with the management to be sure about the issues identified. Then, we conducted meetings with the individuals for inaccurate and unsupported ratings for these issues. These meetings took 15 -30 minutes depending on the number of issues.

3.2 Identifying and prioritizing improvement areas

Although different approaches could be developed we envisaged mainly two approaches for improvement based on the assessment results. These are organizational based and practitioner specific improvements. The former one aims to determine insufficient knowledge domains from an overall organizational viewpoint. The latter one aims to determine specific insufficient knowledge domains for each software practitioner. Within the scope of this paper we only focus on organizational improvement areas.

The first step is to try to determine general state of the organization against the requirement of the roles. This step will show overall inappropriate knowledge areas for each of the roles of the organization. Since our main purpose is to describe the process in this paper we consider only the Software Engineering role.

Figure 1 shows improvement areas for practitioners performing Software Engineering role subject to further verification using the knowledge indicator occurrences in the assessment forms. Knowledge unit priorities for improving Software Engineering Role in terms of knowledge units can be determined by comparing the median of all practitioner knowledge levels with the assessment baseline. If the difference of knowledge level value is less than 1, 2, 3 than this knowledge unit is identified as Significant, Highly Important and Top priority respectively. Furthermore, the same kind of priority table may be produced for each of the practitioner based on their knowledge profiles if practitioner based training is envisaged. For example; according to Figure 1 Test Documentation can be identified as top priority knowledge unit where Coding Standards and Documentation, Unit Testing and Integration Testing can be identified as high priority knowledge units to be improved.

Further analysis of the assessment forms and knowledge indicators of the practitioners performing software engineering role yields that the practitioners have difficulty, for example, in Requirements Engineering because mostly following knowledge unit indicators were missing related to Requirements Engineering: Identifying, representing and managing the 'viewpoints' of the stakeholders of software projects; Advantages and limitations of different types for interviewing techniques and how they should be conducted; Performing requirement classification according to functional and non functional, level of abstraction, prioritizing the requirements, product or process requirements, volatile requirements.

...

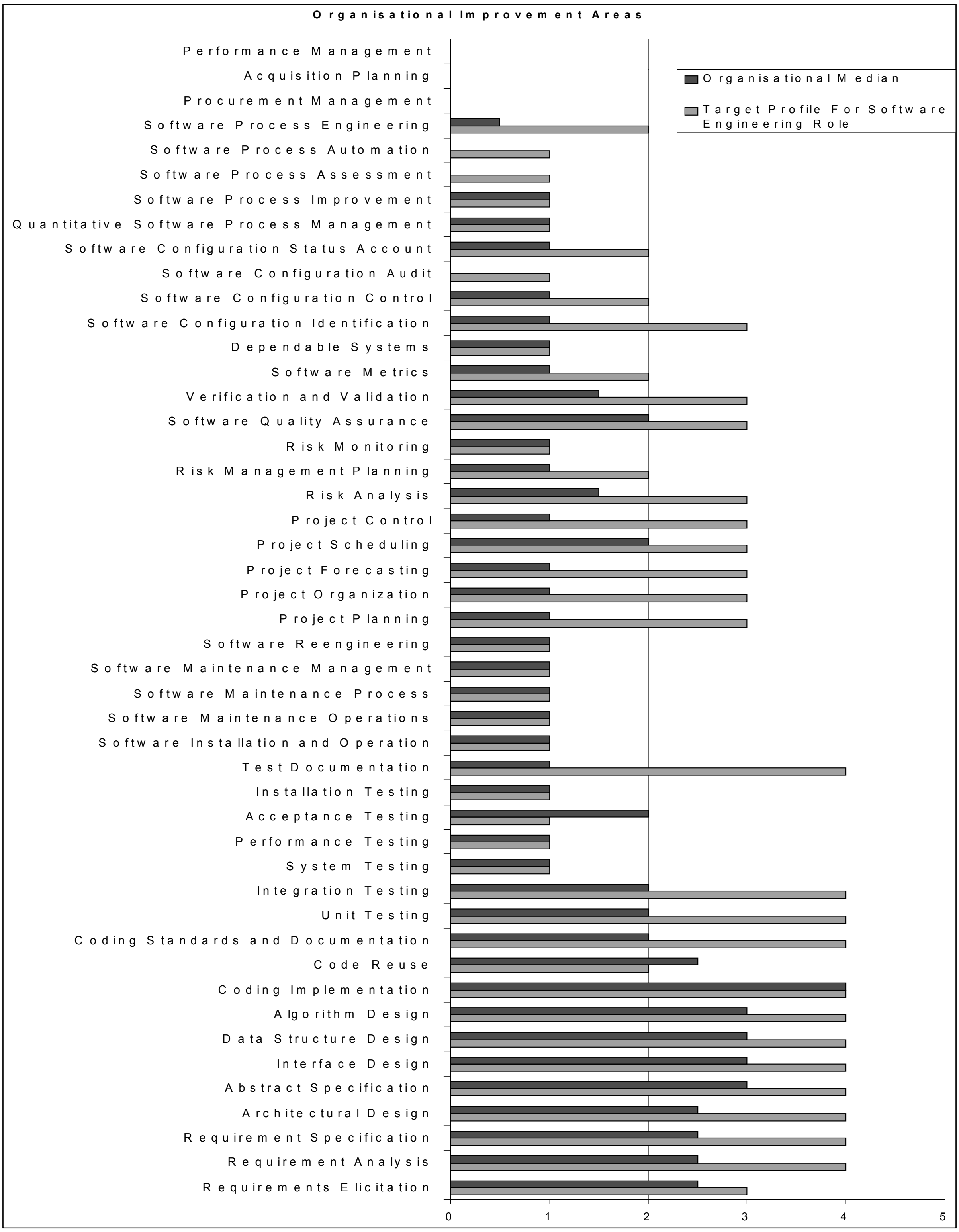


Figure 1: Organizational improvement areas for software Engineering role

3.3 Organizational Improvement Plan

Analysis of the assessment results provides information about the current strengths and weaknesses of the organization’s workforce and indicates opportunities for improvement. Although organizational improvement plan is not within the scope of this case study it is important to provide guidelines how to use this assessment, general principles and possible approaches that could be used by improvement initiators.

In general, improvement areas could be identified based on the following observations:

1) Knowledge units with low ratings: An example of this is Test documentation knowledge unit in which almost all of the practitioners are below comprehension scale.
2) Knowledge units with insufficient rating that are needed to enable the activity aligned with a specific need of the organization: For example; Organization X is willing to improve Configuration Management Activities however 7 of the practitioners have only comprehension knowledge on Configuration Identification Knowledge Unit.
3) Unbalanced specific criteria ratings within a knowledge unit such as low formal education and high practical experience scores: This information could be used to decide the strategy that is needed to be used in planning the training areas. For instance the training programs may give more emphasis on theory than tool and standards or vice a versa.
4) Low knowledge unit ratings across assessed knowledge units that may indicate weakness in specific knowledge categories

Targets for improvement can be quantified for each priority area. These may be either target values for knowledge units for all workforce, or target software role profiles, or combinations of the two. In any case, they should be set in the according to organization's needs which can reasonably be achieved.

4. CONCLUSIONS

As a conclusion, the workforce assessment model is applicable in emergent organizations and produce results, which can be effectively used for process improvement at individual and organizational levels. Specifically during the case study we obtained significant information as an input for improvement plans of the organization under consideration. We found out that the identified problems overlaps with the SW-CMM assessment conducted previously. However, workforce assessment provides the decision makers with more detailed information to improve the workforce as well as the processes. We also identified a shortcoming with the structuring of the knowledge unit indicators. We realized that a further decomposition of knowledge unit indicators is required for accuracy.

In order to develop the assessment baseline, we used activities related to roles and knowledge requirements in terms of knowledge units that are scaled according to scales adopted from Blooms knowledge taxonomy. Furthermore, the assessment questionnaire and knowledge indicators, developed as the assessment technique, are based on the assessment baseline and criteria descriptions. This led to a sound assessment framework. The inclusion of knowledge unit indicators helped the evaluators to be more objective and will help the improvement initiators to propose the contents of the improvement areas.

The study also has some limitations. First of all, the proposed assessment model is based on SW CMM process model considering only Level 2 process areas and Software product engineering process area because the focus of this study is emergent software organizations. For organizations using completely different processes the assessment baselines should be modified. Secondly, Computing fundamentals and software domains knowledge areas are not included. However assessment questionnaire and knowledge indicators can be extended to include them. And finally, in order to determine the knowledge levels of practitioners we used a measurement process, which does ratio operations on the ordinal data. We tried to define this process considering the construct and criteria definition.

Literature

Basili V. and David H., 1983. An Emprical Study of a Syntactic Complexity Family, IEEE Transactions on Software Eng. , vol. SE-9, #6, pp 664-672, November

Bloom, B.S. 1956. Taxonomy of educational objectives: The classification of educational goals: Handbook I, Cognitive Domain. NewYork ; Toronto: Longmans, Green.

Brodman J.G. and Johnson D.L. 1994. What Small Business and Small Organization Say about the CMM, *Proceedings of the 16th International Conference on Software Engineering (ICSE16),* pp.331-340

Curtis, B., William E. H., Sally M., People Capability Maturity Model, Ver. 1.0, Software Engineering Institute, Carnegie Mellon University, CMU/SEI-95-MM-02, September 1995.

Demirors O., Demirors E., Dikenelli O., Keskin B., Process Improvement Towards ISO 9001 Certification in a Small Software Organization, Proceedings of the 20th International Conference on Software Engineering, IEEE Computer Society Press, page: 435-438, 1998.

Hilburn T.B., Hirmanpour, Khajenoori S 1999. Turner R, Quasem A., 'A Software Engineering Body of Knowledge Version 1.0,' Software Engineering Institiute, April 1999

House ER. 1980. Evaluating with Validity. Sage Publications

Humphrey, W.S. 1998, "Three Dimensions of Process Improvement, Part III: The Team Process," Crosstalk 1, 2, 14–17.

Juan A, Garcia R., Juristo N., Lopez M. And Moreno A. 2000. A More Rigorous and Comprehensive Approach to Software Process Assessment, Software Process: Improvement And Practice, vol 5:3.

ISO-15504. 1996. ISO/IEC 15504: Information technology-software process assessment. International Organization for Standardization.

ISO/IEC 12207 1995. Information Technology, Software Life Cycle Processes, International Standard.

Paulk, M.C., Weber, C.V., Chrissis, M.B. 1995. The Capability Maturity Model: Guidelines for Improving Software Process, Addison Wesley, Reading, MA.

Paulk, M.C., Weber, C.V., Garcia, S., Chrissis, M.B. and Bush, M. 1995. Key Practices of the Capacity Maturity Model, Version 1.1, *Technical Report CMU/SEI- 93-TR-25,* Software Engineering Institute, Pittsburgh, PA.

Scriven M. 1991. Evaluation Thesaurus, Sage Publicaitons: Newbury Park, CA

2 Author CVs

Ö. Özgür Tanriöver

He has a PhD in computer science and is a professor at the Department of Information and Technology, Faculty of Technology and Management (PAV K14), Eindhoven University of Technology, Neitherlands, Phone: +31 40 2475517, E-mail: O.Tanriover@tm.tue.nl. Before that Mr Tanriöver worked at the Middle East technical University from 1993 to 2002. The research interest includes Software Process Improvement & Assessment, Workflow Management Systems, Software Quality, Software Estimation, and IT labor & skills assessment, Learning Organizations.

Dr. Onur Demirors

He has Ph.D. and M.Sc. degrees in Computer Science and B.Sc. degree in Computer Engineering. He has been working in the domain of software engineering as a consultant, academician and researcher for the last 15 years. His work focuses on software process improvement, software project management, software engineering education, software engineering standards, and organizational change management. He worked as a consultant for a number of software developing companies to improve their processes based on ISO 9001, ISO 15504 and CMM. He managed a number of research and development projects on software process improvement, business process modeling and large scale software intensive system specification/acquisition. He has over 40 papers published in various books, journals and conferences. Address: 1. Cad. No 14 Ümitköy , Ankara Turkiye, Office: +90 (312) 210 3749, Home: +90 (312) 235 7844, Mobile:+90 (532) 297 6978, e-mail: demirors@metu.edu.tr

Process assessment for use in very small enterprises: the NOEMI assessment methodology

Bernard Di Renzo, Christophe Feltus
Centre de Recherche Public Henri Tudor

Abstract

This paper relates to the development and the experimentation of an IT process assessment methodology especially designed to be used in very small enterprises (VSEs). This methodology, called NOEMI[1], has been developed as a critical part of a public research project of the Centre de Recherche Public Henri Tudor (Luxembourg).

Initially the main objective of the NOEMI process assessment methodology was to contribute directly to the implementation of a collaborative IT-sourcing model, developed in the same research project.

The process portfolio aims at a whole coverage of the usual IT-practices in VSEs. It is business value-driven and designed in five process areas: infrastructure, service support, management, security, and documentation. The processes themselves are based on a combined approach of ISO/IEC 15504 and the IT Infrastructure Library.

The capability model defined in the NOEMI methodology explores the gap between level 0 and level 1 of ISO/IEC 15504 in order to match in a more accurate manner with the reality of VSEs. The capability profile has four levels and is performed for the process areas and not for the processes themselves, so allowing easy comparison between VSEs.

We are now performing the seventh experimentation of the NOEMI assessment methodology. Each case has been a success according to the feedback of the VSEs. And we are considering the transfer of our methodology to French and Belgian partners through dissemination projects.

It leads us to promote the NOEMI assessment methodology as a public package tool especially designed for use in a VSE context, which aims to enhance business value through IT.

This paper introduces the methodology and considerations based on case studies.

Keywords

Assessment methodology, capability model, process portfolio, very small enterprise, service management, SPICE, improvement program, ITIL.

[1] "New Organisation of IT-Production and Support" [Nouvelle Organisation de l'Exploitation et de la Maintenance Informatiques].

1 Introduction

Nowadays, organizations are highly dependent on their Information Technology (IT) services. They expect them not only to deliver high-quality services at the lowest cost but also to enhance innovation in their business.

The Software Process Improvement (SPI) community is dramatically aware of these requirements and deploys huge efforts to meet this ambitious objective. Numerous companies are involved – in some cases for a long time already – in process improvement. Software processes have been closely targeted by fine-tuned methods. The international standard ISO/IEC 15504 [1], as the result from the Software Process Improvement and Capability dEtermination (SPICE) initiative, plays an important role in SPI.

IT Service Management (ITSM) area is also largely – if not entirely – covered by other standards. Many companies have also significantly improved their investment in this way. In that context, the IT Infrastructure Library (ITIL) [2], which has been produced by the Office of Government Commerce (OGC) in the United Kingdom since the late 1980's, is probably the most comprehensive approach on providing IT-services. The ITIL is publicly available and has become the worldwide *de facto* standard for IT Services Management.

Standards contribute dramatically to enhance business value through IT-disciplines, enabling efficient software development, provision of high-quality services at the lowest cost, and business innovation [3]. Furthermore, implementing some of these powerful standards is obviously not an easy task. Implementation projects are resource consuming, require skills, and are expensive. Large organizations have usually the means to match their ambitions and they can resolutely engage themselves in such ways.

The situation is dramatically different for very small enterprises (VSEs), which have no IT-dedicated staff. They are often ready to invest but they have rarely the time and never – or almost never – the skills. Even if they have the same expectations than big companies, it is very difficult for them to implement standards and to improve significantly IT-quality.

Founded in 1987, the Centre de Recherche Public Henri Tudor[2] has the prime mission of strengthening the economic structure of the Grand-Duchy of Luxembourg through technology development and innovation, especially in the information and communication technology (ICT) fields. The Centre is notably involved in providing assistance to VSEs of the Grand-Duchy of Luxembourg as well as the "Grande Région" which includes Wallonia in Belgium, the Lorraine region in France and the Saar in Germany.

In that way, the Centre set up a public research project called "New Organisation of IT-Production and Support" [Nouvelle Organisation de l'Exploitation et de la Maintenance Informatiques (NOEMI)], which aims to develop a model of collaborative IT-management for use in VSE clusters in order to optimise the management of their respective information systems in terms of quality, reliability and cost [4]. Through this project, the Centre focussed on the improvement of IT-practices for the use of VSEs. The developed model is currently being experimented with VSEs in Luxembourg. It proposes to the VSE-partners to gather their IT-services and manage them in a collaborative and cross-participative manner. Pragmatically, the collaborative management can take two basic forms: either the partners establish a common organization that insource their respective IT-activities or they can outsource their clustered IT-activities to an external provider. The results of the experimentation are promising and should allow the validation of the model in its final release by June 2004.

One of the most critical requirements for the model is the homogeneity of the partners' IT-capabilities. Indeed, in the case of sourcing to an internal common IT-organization, the heterogeneity of IT-capabilities of the partners could lead to management issues, particularly in the financial area. So assessing the IT-capability on an extended IT-scope, with an holistic approach and a focus on business value is one of the model keys of success.

[2] Called hereafter the Centre. See www.tudor.lu and www.citi.tudor.lu

This paper describes the assessment methodology developed as a critical topic of the NOEMI project with particular focus on the objectives, the links with ISO/IEC 15504 standard and ITIL, and the results of the experimentation of the methodology through case studies.

2 *Objectives of the NOEMI assessment methodology*

The assessment methodology must be aligned with the scope and objectives of the project, which allows VSEs to control IT in terms of cost and quality. The assessment must also contribute directly to the implementation of the model within the VSE-partners. According to these requirements, a specific assessment methodology was then developed as a critical topic of the NOEMI project.

The operational implementation of the NOEMI model is based on the Deming cycle and consists of a five-stage project for each VSE partner [4]:

1. Start-project assessment of the IT-practices including the first capability determination and the improvement program topics
2. Action Plan 1 consisting of the implementation of a first part of the improvement program
3. Mid-project assessment driving to the second capability profile and evaluation of the partner satisfaction
4. Action Plan 2 consisting of the completion of the improvement program as the continuity of the first Action Plan
5. End-project assessment driving to the third capability profile

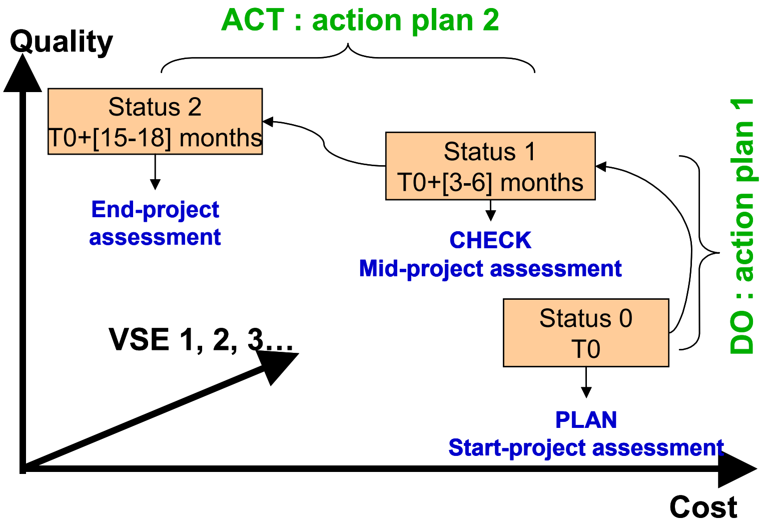


Figure 1: Operational implementation of the NOEMI model

Assessment occurs at stages 1, 3 and 5 in the implementation project of the NOEMI model. It allows one to objectively measure the gap bridged by both Action Plans on both quality and cost axis, according to the dual objectives of the model i.e. cost control and service quality optimisation.

Furthermore, the assessment itself, extracted from the model, has been performed within several VSEs in Luxembourg. The success of the methodology allows us to consider the NOEMI assessment as a fully independent service especially designed for usage in VSEs. The NOEMI assessment could obviously drive to the creation of a repository of capability profiles of Luxembourgish VSEs. This point of view is illustrated later in the case study.

Moreover, the NOEMI assessment methodology is ready to be promoted outside the Grand-Duchy of Luxembourg using a few transfer projects from the Centre. In this context, several regions in France and provinces in Belgium are very interested in the NOEMI assessment methodology.

3 Links between NOEMI assessment methodology and ISO/IEC 15504, service management and security management

The international standard ISO/IEC 15504 provides a framework for the assessment of software processes [1]. As a standardized and public approach, it provides a shared methodology of assessment and understanding of software process portfolio. Process assessment within this standard targets two main purposes: process improvement and capability determination. Even if ISO/IEC 15504 is considered nowadays as a software-oriented standard, it is relevant that it is suitable for use in any other context of process assessment. The NOEMI assessment methodology is directly inspired from ISO/IEC 15504 standard, tailored for use in VSE.

Created in the late 1980's in the UK, the IT Infrastructure Library (ITIL) [2] [5] is a public library that focuses on IT Services Management (ITSM). Today, it is recognized as the worldwide *de facto* standard for high-quality service provision. The main volumes of the ITIL concern the Service Support and the Service Delivery. These two domains are organized in comprehensive and exhaustive sets of processes. Service Support area covers incident management, problem management, configuration management, release management and change management. The Service Delivery deals with capacity management, availability management, financial management for IT-services, Service Continuity for IT Services and service level management. The ITIL has clearly a best practices-based approach and encourages continuous quality improvement.

As the NOEMI assessment deals not only with software processes but also with IT-service processes, the ITIL has been selected to provide matters that are not covered by ISO/IEC 15504 [6]. In addition to the SPI and the ITSM, security management, as a third discipline, is considered in the NOEMI assessment in order to get an IT-wide process scope. ISO/IEC WD 18028, ISO/IEC JTC1/SC27/WG1 [7] provides matter for this discipline. Outputs of other projects of the Centre bring a complementary view to the methodology [8].

4 The NOEMI assessment methodology

4.1 Defining an holistic process portfolio for use in VSEs

An initial tailoring of the ISO/IEC 15504 concerns the process portfolio in order to make it fit within the scope of the NOEMI model. According to the objectives of the project, it leads to get the most holistic coverage of the usual IT-activities in VSEs.

Based on the experience of the Centre through numerous project outputs [4] [6] [8], five process areas were defined to fully cover the IT-scope of VSEs: infrastructure, service support, management, security, and documentation.

So a lean-process portfolio covering an holistic IT-approach were defined within the NOEMI assessment methodology. The processes are driven by ISO/IEC 15504, the ITIL, and security discipline standards. Other quality management standards [11] [13] [14] [15] bring complementary topics to fulfil the matter in order to define the processes. They are tailored for use in VSEs without any dedicated IT-staff. The formalism of the process is taken directly from ISO/IEC 15504.

The following figure shows the areas of processes within the NOEMI model. The structure is directly related to the business value brought by the process area.

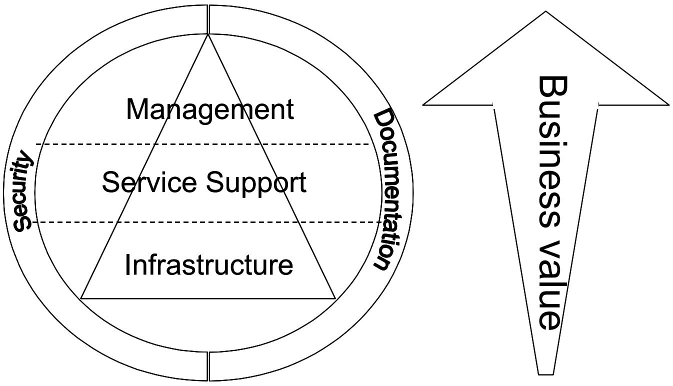


Figure 2: NOEMI process areas

The infrastructure area covers the following technical processes: network management, system management, connected-workstations management, and groupware management. These technical processes are based on best practices collected in this area [2] [9] [10]. As the bottom of the pyramid, the infrastructure area deals with the first processes to consider for VSEs in order to control IT. It is the base of the pyramid and will contribute very little to the business value. Nevertheless, it is necessary to allow the upper areas – service support and management – to deliver their own added-value.

The service support area includes the five ITIL processes in this discipline i.e. incident management, problem management, change management, configuration management and release management [2] [5] [10]. As the centre of the pyramid, the service support area offers a middle value to the business i.e. more than the infrastructure area and less than the management one. Both the infrastructure and service support areas are necessary for the management one to be efficient.

The management area deals only with the most usual processes used by VSEs . It includes provider management, acquisition management, project management and IT-financial management [1] [2] [8] [16]. As the top of the pyramid, this area drives the largest value for business but expects a good capability of lower areas.

In order to fit with the understanding that VSEs usually have of this field, the security area [7] [12] focuses on the basics. It includes only the following processes: server availability management, electrical power management, backup and restoration management, anti-virus management, technical room management, resource access management, external access management. Transverse to the pyramid, the security area impacts directly on all the other areas of the model. Moreover, these processes are compliant with recommendations of the Luxembourg government with respect to the security field.

The documentation [11] area is based on document inspection. It consists of a deep analysis of the documents, based on criteria related to structure, completeness, consistency, and clarity.

4.2 Defining a capability model for use in VSEs

The international standard ISO/IEC 15504 defines a capability model on 6 levels: 0-incomplete, 1-perfomed, 2-managed, 3-established, 4-predictable, 5-optimising.

Based on the field experience of the Centre, the capability level of the most VSEs in Luxembourg is between the level 0-incomplete and the level 1-performed for almost any IT processes, and rarely up to the level 2-managed. The gap between levels 0-incomplete and 1-performed defined in ISO/IEC 15504 appears then to be too deep for use in VSE-context with any relevant accuracy.

The NOEMI capability model is derived directly from standard ISO/IEC 15504 by an *ad hoc* tailoring in order to fit with the reality of the VSE's world. Additional levels between level 0-incomplete and level 1-

performed have been defined as described in the following table.

ISO/IEC 15504		NOEMI
5-optimising	<->	Not used
4-predictable	<->	Not used
3-established	<->	Not used
2-managed	<->	Not used
1-performed	<->	**3-green**
	<->	**2-orange**
	<->	**1-red**
0-incomplete	<->	**0-dark**

Figure 3: Links between ISO/IEC 15504 and NOEMI capability models

The NOEMI capability model defines intermediate levels before to get the level 1-perfomed, allowing a more accurate quantification of the capability of the VSE. The levels are colour-coded (dark – red – orange – green) thus allowing intuitive understanding by people without any knowledge of or practice with capability models.

The capability profile is achieved by the process areas and not by processes as in ISO/IEC 15504. It aims to get the same scope of assessment for any VSE with the determination of a capability level for each one of the five process areas. This pragmatic approach allows an easy comparison of the VSE within a global IT-coverage.

The NOEMI capability levels are defined as following.

Level	Description
3-green	An area has a level 3-green capability when its processes are developed and performed in a globally satisfying way. Proactivity and mean term planning are notions usually related to such an area. In term of business value, it offers a competitive advantage to the VSE.
2-orange	An area with a capability at level 2-orange means that the VSE achieves considerable efforts to develop and perform the processes of the area. Reactivity and short term are notions inside a level 2-orange capability area. The business value of such an area is less than what the VSE could expect from it.
1-red	An area has a level 1-red capability when its processes are very partially performed, usually following specific or local needs. Most requisites are missing. Lakes of efficiency and productivity inside the company counterbalance the business value brought by such a level. Moreover it can lead to potential risk for the business.
0-dark	Any process of a level 0-dark capability area is performed. The area is a brake for the efficiency of the VSE. It is a dramatic risk for the company.

Figure 4: NOEMI capability model

4.3 Defining the assessment process for use in VSEs

The NOEMI assessment process is based on ISO/IEC 15504. All the processes are systematically assessed in any performance. It allows having a global approach of IT-activities in VSEs according to the project's objectives.

The assessment process schema is illustrated in the following Figure.

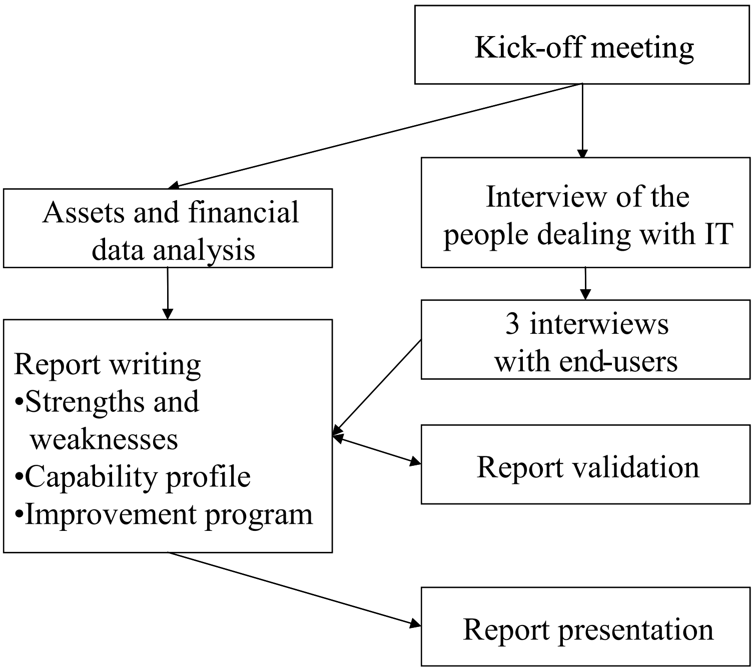

Kick-off meeting
Assets and financial data analysis
Interview of the people dealing with IT
3 interwiews with end-users
Report writing
•Strengths and weaknesses
•Capability profile
•Improvement program
Report validation
Report presentation

Figure 5: NOEMI assessment process schema

The report is driven from a structured based template including chapters dedicated to each area of processes and a conclusion. For each area, there is a summary of the strengths and the weaknesses of the area, a detailed analysis, and the recommendations. The conclusion chapter introduces a synthesis with the consolidated table of strengths and weaknesses, the capability profile, and the improvement program.

5 Case study

5.1 Assessment context

In a VSE, a manager, an accountant or a member of staff showing interest and having acquired knowledge by practice are managing the IT-system. No real IT-professional was met during the assessments.

This statement is due to the size of the assessed IT-infrastructures, which is limited to an average of 40 PCs, some servers and one or two business software (see Figure 6).

	VSE A	VSE B	VSE C	VSE D	VSE E
Number of PC 's	15	23	50	43	59
Number of servers	2	1	4	4	2
Network type	Fast-ethernet	Fast-ethernet	Fast-ethernet	Fast-ethernet	Fast-ethernet
Internet access type	ISDN Router	Stand-alone PC	ISDN Router	ISDN Router	Router
Number of business software's	1	1	1	2	2

Figure 6: Compositions of the IT infrastructure by encountered companies

From the above table, it can be seen that in all assessed companies, the useful manpower needed for the IT-management was insufficient to justify hiring a full-time IT-employee.

5.2 Reactivity of the VSEs

Assessed companies realized, before the submission of the final report, the potential improvements within their IT-production. This led to actions taken in order to cure critical situations even before the report was delivered.

For example:

- VSE D, which did not have an up-to-date inventory, used the time between the assessment and the report presentation to bring it nearer to reality.
- VSE A, which did not have an anti-virus protection on its servers, did not wait for our written recommendations to set up effective anti-virus protection.

5.3 Capability profiles

A similarity in the assessed areas' capability level was observed inside each company.

The infrastructure is the strongest area in three out of five VSEs. In the assessed companies, IT-infrastructure is too restrictively considered at hardware and software. Taking into account and investing in elements such as the service support or documentation remains a luxury. Companies allocate resources for business processes more readily than for IT-processes because an added-value is easily quantifiable.

IT-documentation remains the weakest area in four out of five VSEs. Documentation is limited to a more or less up-to-date inventory, some maintenance contracts and the assets or software invoices. Analysis shows a lack in the basic rules such as document identification, storage, and utility for the enterprise.

Practices regarding the service support are also only slightly put in place in four out of five VSEs. An efficient help-desk is always lacking and limited to a deputy member of staff, whose mission is "fire fighting" and lending first "assistance" when the IT system malfunctions. Once more, the added-value of such a service remains difficult to quantify and priorities are placed on business processes.

The table below illustrates VSEs' capability profile per area. The minimum capability is 1 and the maximum is 3.

	VSE A	VSE B	VSE C	VSE D	VSE E	Average
Infrastructure	3	3	1	2	1	2.0
Service support	1	2	1	1	1	1.2
Management	2	2	3	2	2	2.2
Security	1	2	2	1	1	1.4
Documentation	1	1	3	1	1	1.4

Figure 7: VSE capability profile per area

5.4 Example of assessment recommendations

This chapter presents the more frequent assessment recommendations classified by areas.

5.4.1 Assessment of management practices

Several recommendations were made in each VSE within the area of the assessment of management practices. Among these, there are three relating to project management, financial management, suppliers and acquisitions management.

Improvement of project management

Project management is the success driver of all IT-projects. Effective project management makes it possible for companies to make a maximum profit from their investment.

Several weaknesses were observed. Figure 8 shows three of them:

VSE	Weaknesses
VSE A	Delegation of the software selection project management directly to the software users
VSE B	Insufficient time for the project leader to enable him to conclude his work
VSE C	No training for the project leader. Consequently, he does not effectively establish and manage the users’ requirements

Figure 8: Encountered weaknesses regarding project management

Improvement of knowledge in project management is the most important recommendation among the management practices assessment area.

Improvement of financial management.

Several weaknesses were observed regarding financial management, here are four of them:

VSE	Weaknesses
All VSEs	Financial resources are not management
All VSEs	IT budget not established
All VSEs	Long-term visibility is lacking
All VSEs	Investments are decided on a case-by-case by management according to needs and opportunities

Figure 9: Encountered weaknesses regarding financial management

Encountered companies are rarely aware of the hidden costs due to the losses of non-productivity. These costs represent a loss generally higher than the annual cost of the IT-infrastructure. The assessment recommends thus to improve financial management.

Improvement of suppliers and acquisition management

Suppliers and acquisition management is also not a well-established practice among the companies assessed:

Here are two interesting encountered cases:

VSE	Weaknesses
VSE B	By acquiring its business software, VSE B underestimated implementation costs. The consequences of this under evaluation were that several years after the initial acquisition, the software is still not entirely in production. The fact that the software implementation cost is higher than its acquisition cost is not always clear for a non IT-professional
VSE D	VSE D became aware a short time after acquiring its business software, that it does not correspond entirely to its requirements.

Figure 10: Encountered weaknesses regarding suppliers and acquisitions management

Consequently, the assessment report proposes to establish an acquisition policy. This one includes acquisitions guidelines such as acquisition frequency and material types.

With regard to software acquisitions, it is also recommended to improve project leaders the know-how via training.

5.4.2 Assessment of the service support practices

Several recommendations were made in each VSE within the area of the assessment of the service support practices. Among this, here are three relating to help-desk, incidents reduction program, and the assessment of the IT-users satisfaction level.

To establish a recognized and accessible help desk

The management of incidents is the help-desk's first mission. Its objective is to restore the correct functioning of the IT as quickly as possible.

In all of the assessed companies, a minimum help-desk was in place. It was staffed by one or sometimes two "power-users". The help-desk activity was a secondary task for them. In case of incident, the users either request the help-desk or solves themselves the problem.

The setting up of a recognized and accessible help-desk was a recommendation of the assessment. It included the required means (financial, time, technical training) and the education of the users to systematically refer to it.

To establish and set up a program of incidents reduction

Within the assessed companies, the estimation of the number of incidents per month and user varies strongly considerably depending on view: the users give the higher number than the help-desk as shown in Figure 11.

	VSE A	VSE B	VSE C	VSE D	VSE E
Minimum	N/A	5	10	15	N/A
Maximum	N/A	10	28	20	N/A

Figure 11: Number of Incidents per month and users

An incident is defined according to ITIL criteria's, namely: "Any event that is not part of the agreed service". It is thus neither depending on the time for resolution, nor the solution found to cure it, nor of the person or service which it solves.

The recommendation proposed, on the one hand, to get the average number of incidents ranging between 2 and 5 per month and per user and, on the other hand, to establish a help desk working in a "pro-active" way. This means that it identifies the problems at the source of many incidents and solves them definitively by implementing call abatement projects.

The setting up of an incident reduction program requires additional resources (with necessary competences) during its implementation.

To assess the IT-users satisfaction level

None of the assessed companies measures their IT-users satisfaction level.

The level of User Satisfaction is a perception, always subjective, of the effectiveness and quality of the use of the IT. Its measurement allows one to apprehend, without large effort, the hidden IT-costs related to incidents loss of productivity, lacks of knowledge, or inadequacies between staff needs and software tools.

Moreover, concrete improvement actions resulting from users' satisfaction assessments have motivating effects on company staff.

A report recommendation is put forth in all the assessments currently carried out.

5.4.3 Assessment of the IT-infrastructure

Several recommendations were made in each VSE within the area of the assessment of the IT-infrastructure. Among this one, here are two relating to the operating system (OS) standardization and the use of the files server.

OS standardization

Repartition of the operating system running under the workstations is presented above.

	VSE A	VSE B	VSE C	VSE D	VSE E
Windows 95	12 %	20 %	0 %	54 %	N/A
Windows 98	69 %	0 %	0 %	16 %	
Windows NT	0 %	0 %	0 %	14 %	
Windows 2000	0 %	80 %	22 %	16 %	
Windows XP	19 %	0 %	78 %	0 %	

Figure 12: Distribution of the operating systems

Assessed VSE do not have a strong operating system guideline. For the main part of the PC's, the operating system is generally the one delivered with the workstations, so that by analysing the figures of Figure 12, it can be seen that companies B and C have renewed their IT infrastructure recently relatively to 78 and 80 percent.

On the other hand, companies A and D have workstations running under almost the whole Windows OS family. This situation is difficult to manage. It complicates maintenance and consequently increases security risks. These two companies were advised to make an OS standardization.

To develop the use of the file server

Advantages of the files servers are: documents' unity and accessibility, secure files storage and up-to-date information.

All assessed companies have a file server. But the way they use it is nevertheless not the same from one company to another. It is usual that users save professional information on the workstation hard drives.

It is recommended to create awareness in order to use the file server to store professional documents. This file server must, moreover, be structured in an efficient way and explained on the VSE-guidelines.

5.4.4 Assessment of practices regarding security

Several recommendations were made in each VSE within the area of the assessment of practices regarding security. Among this one, here are three topics relating to password procedure, anti-virus protection and back up operation.

To improve the password procedure

Password is the first protection against intrusions.

The use of passwords' pratices is the most divergent topic within the assessed companies. In one VSE out of five, the password was used conscientiously and the employees kept it completely confidential. In the others, the password management procedure was in place but the confidential was limited.

The assessment recommends setting up a policy which forces staff to keep the password personal and confidential. This confidentiality is extended to the system administrator.

To improve anti-virus

Anti-virus is important so as not to get corrupted files on the internal network.

Among all of the security area procedures, anti-virus is the one most commonly in place. It was not installed in a consistent way from one VSE to the other. One out of five of the assessed VSEs used up-to-date anti-virus software on the entire infrastructure. The others did not use an anti-virus on all PC's and servers: they used an anti-virus protection that is more-or-less up-to-date, on almost the entirety of the infrastructure.

Because all of them had an Internet access and can import corrupted files, a recommendation was each time introduced.

To optimise the back up operation

A back up is a second copy of the data onto a tape or another media making it possible to restore data in the event of loss or destruction of the original media.

This procedure exists in the all assessed companies, but the rigour and the quality varies from one to the other. A certain number of gaps are identified, among those: daily complete back up rather than differential, lack of awareness of the staff about the data taken into account in the back up, restore procedure not systematically tested, or out-of-date back up media.

Recommendations regarding these gaps were made to improve the process by integrating all of the above elements.

5.4.5 Assessment regarding document management

Only one recommendation was made in each VSE within the area of the assessment regarding document management. This recommendation was the improvement of the document management.

To improve the document management

Document management is the weakest point of all the companies. In each one, the documentation is stored without any particular rule: invoices are stored at accounting; contracts are kept by the management; technical documentation stored by IT staff. The documents are on an electronic or a paper format according to possibilities. They only are partially identified and the version numbering was always missing.

This generalized weakness documentation has a negative impact on VSE’s efficiency. Moreover, the lack of documentation represents a risk in a hypothetical departure of someone from the “IT-staff”.

6 Conclusion

The NOEMI assessment methodology was specially designed to cover the global IT-activities of VSEs as a critical topic of a public research project of the Centre Henri Tudor in Luxembourg. The initial objective of this methodology was to directly contribute to the implementation of a model of collaborative IT-sourcing developed in the same project.

The process portfolio is mainly derived from ISO/IEC 15540, the IT Infrastructure Library, security standards, and outputs of other projects of the Centre. The capability model deeply explores the gap between both the lowest levels of the ISO/IEC 15504 model in order to match closely with the reality of VSE. It focuses on five areas of processes: infrastructure, service support, management, security, and documentation. The assessment process itself is directly based on ISO/IEC 15504 and systematically involves all processes defined within the five areas.

However at the time of writing this paper, the methodology was already experimented with five times. Each performance was a success according to the positive feedback of the VSEs. Two other assessments are currently in progress. And more than ten VSEs are interested in being assessed using our methodology, which is a significant number with regard to the current promotional effort. Furthermore, some of the Centre’s research and dissemination partners are interested in experimenting with it in France and Belgium.

This success story of the NOEMI assessment methodology with the VSEs leads us to consider it as an independently packaged tool that can contribute pragmatically to the efficiency and competitiveness of the VSE thanks to IT-quality improvement in software processes, in service management, and in security. Overall, it serves to enable business value through IT-innovation.

Today we can already consider the dissemination stage of the NOEMI assessment methodology. Private consultants – as free-lance or service provider – could be trained to use this methodology within an *ad hoc* partnership in order to ensure the related deontological framework. In a similar approach, the NOEMI assessment methodology is ready to be transferred to non-profit or public organisations as a new public assessment methodology that closely matches the needs of very small companies.

7 Literature

[1] ISO/IEC JTC 1/SC7, ISO/IEC TR 15504, 1998

[2] IT Infrastructure Library – Service Support, Service Delivery, Application Management, ICT Infrastructure Management, Office of Government Commerce, United Kingdom, 2000 – 2002

[3] Henderson J.C., Venkatraman N., Strategic Alignment: Leveraging Information Technology for Transforming Organisations, in: IBM Systems Journal, Vol. 32, No. 1, pp. 4-16, 1993

[4] Di Renzo B., Feltus C., Development and experimentation of a model of IT collaborative management for use in SME, in: Proceedings of the International Conference AIM 2003, Grenoble, France, 2003

[5] IT Infrastructure Library – Practices in small IT unit, Central Computer and Telecommunication Agency, 1995

[6] Barafort B., Di Renzo B, Merlan O., Benefits resulting from the combined use of ISO/IEC 15504 with the Information Technology Infrastructure Library, in: Proceedings of the International Conference Profes 2002, Rovaniemi, Finland, 2002

[7] ISO/IEC WD 18028, ISO/IEC JTC1/SC27/WG1, 2001

[8] Barafort B., Hendrick A., A SME dedicated framework to evaluate the internal usage of IT, in: Proceedings of the International Conference Software Process Improvement, SPI'98, Monte-Carlo, Principauté de Monaco, 1998

[9] Lyke H., Cottone D., IT automation: the quest for "lights out", Prentice Hall, Upper Saddle River, NJ 07458, USA, 2000

[10] Bonjean C., Help desk: assistance utilisateurs et relations clients, Hermes Science publication, Paris, France, 1999

[11] NFEN ISO 9001:décembre 2000

[12] IT Infrastructure Library – Security management, Office of Government Commerce, United Kingdom, 2002

[13] The EFQM Excellence Model – Small and Medium sized Enterprises, EFQM publications, 1999-2001

[14] Mark C. Paulk, Bill Curtis, Mary Beth Chrissis and Charles Weber, Capability Maturity Model for Software, Version 1., SEI, CMU/SEI-93-TR-24, Pittsburgh, Pa, Feb. 1993

[15] SPIRE Project Information Pack - ESSI Project 23873 – April 1997

[16] Sturm R., Morris W., Jander M., Foundations of service level management, SAMS edition, Indianapolis, USA, 2000

8 Author CVs

Bernard DI RENZO

Bernard Di Renzo graduated as a “Civil Engineer” in electronics and computer science from the University of Liege (Belgium) in 1989. He first worked as a project engineer in the engineering company of an international industry group. There he managed several projects related to the IT discipline. From 1993 to 1999, he was the head of the IT-affiliate of a banking group in Luxembourg. In this function he developed an accurate focus on quality service provision. He got high quality results through process approach, software automation and staff empowerment. Then he joined an international clearinghouse for wireless operators as the Head of Worldwide Client Support. Since 2001, he has worked in the CITI (Centre d’Innovation par les Technologies de l’Information) department of the Centre de Recherche Public Henri Tudor (Luxembourg). He has managed several projects related to quality in the IT discipline with a special focus on process assessment and improvement program, security management, risk management, IT clustering… He is a SPICE Qualified Assessor, ITIL Foundation Certified, and Quality System Auditor certified. He is the coordinator of a project portfolio related to quality and certification. He also the head of a research unit related to process assessment and improvement, service management, operational risk management, strategy management, and new organisation framing.

Christophe FELTUS

Christophe Feltus graduated an Electromechanics Engineer from the Institut Supérieur Industriel des Art et Métiers Pierrard in Virton, Belgium. He worked for several years in private companies as : Production Head at Pfizer SA in Jette, Belgium, Project Coordinator at Nizet Entreprise in Louvain-la-Neuve, Belgium, and Assessor for the Civil Belgium Aviation Administration in Brussels, Belgium. He joined the Centre de Recherche Public Henri Tudor in the Grand-Duchy of Luxembourg in 1999 to work in the Centre d’Innovation par les Technologies de l’Information (CITI). There he has taken part in a project called Prisme Tremplin for the accompaniment of SMEs towards the information society. In 2002, he integrated the project Noemi (Nouvelles Organisation de l’exploitation et de la Maintenance Informatique) as Assessor and he now leads the projet AIDILL (Aide, Information & Développement de l'Informatique Libre au Luxembourg) which aims to promote and develop free software in Luxembourg.

A CMM® Project Takes Off
A Project Milestone Report

Christian Knüvener, method park Software AG

Abstract

Large amounts are being spent on internal projects focusing on software and hardware process improvement. Nowadays, this fact applies especially to the automotive sector, both car manufactures and suppliers. Today's main concern is to adapt quality models as well into the company's specific development as engineers are able to understand the new, quality-assurance procedures as useful and easy-to-adapt instead of heavy-handed bureaucracy. If engineers would recognise the usefulness of the new quality-assurance procedures, they would start conforming to them before being requested or instructed to do so – simply because the guidances, templates, examples and check lists are found to be helpful and the description of the actions are found as supporting daily work. Then these procedures become self runners and the project takes off. There is no need to convince your people any more but you concentrate on further optimization of the main processes.

This desirable scenario occured in a software process improvement project of a medium-size automobile supplier. Important for the projects' success was the continuous, unrestricted commitment from the management. This made it possible to encourage even people with constant doubts. Very remarkable is the extent of the broad support for the project up to today, 15 months after it started.

This report describes the starting situation and the serveal steps in this CMM improvement project. Despite some problems at the beginning, the project proceeded successfully. Therefore this led to an extension of the project goals at the end of 2002. Additionally, main success factors are listed and an outlook is given.

Keywords

Software Process Improvement (SPI), Improvement Project, Capability Maturity Model (CMM), Capability Maturity Model Integration (CMMI), Development of mechantronic systems

1 Starting situation

A German medium-sized automotive supplier with approximately € 1 billion annual turnover has a department, where about 40 employees develop control units for one of their mechatronical products. The situation at the beginning of the project in 2001 was characterized by substantial internal efforts for troubleshooting. Huge stress and intuitive ways of performing tasks led to poor processes.

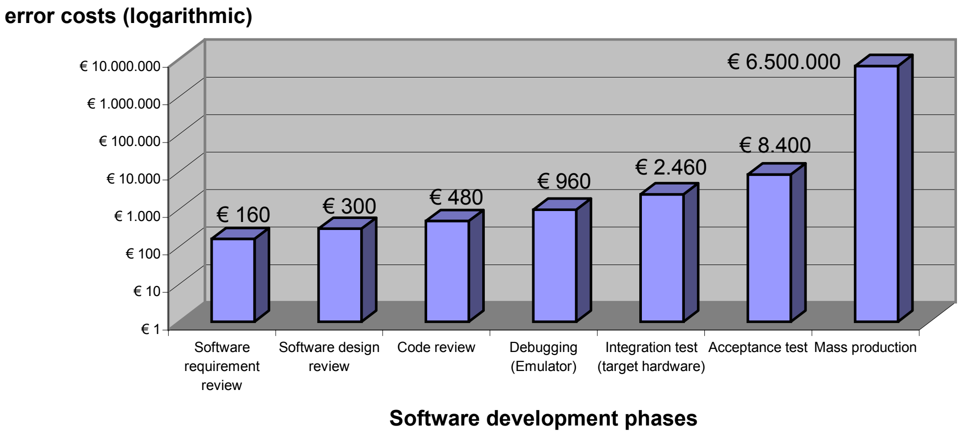


Figure 1: Average error correction costs of each development phase for a serious fault, measured by a German automotive supplier in 2000.

Certain circumstances helped initiating the project. Two back-to-back errors caused serious defects in mass production. Therefore, the efforts put into fault correction were measured for the first time over several projects. A survey prooved that the costs of correction increased nearly exponentially in each development phase, and the costs rose even worse in mass production. Only one of these errors costs more than € 6.5 million. This awareness increased the willingness to invest in error prevention, early testing and reviewing, and especially process improvement enormously [7].

The management's commitment to a software process improvement (SPI) project was also supported by an experiment conducted by a few motivated employees who voluntarily reviewed all relevant documents during a customer project. Therefore, costs rose about 22% during specification and implementation phase. But, in contrast, the number of errors found during system testing decreased about 52%. This led to a Return on Investment (ROI) of 3 to 1, based on a very conservative estimation. This result motivated staff and management and paved the way for more efforts into quality assurance.

The management decided to use the Capability Maturity Model (CMM) Level 2 [8] as a guideline for their software process improvement. Reasons therefore were:

- the starting situation was unsatisfying, especially the efforts for error correction
- the above-mentioned experiment with the result of 3:1 ROI
- the employees' suggestion of trying CMM
- a competing department also started an CMM project
- the very high international importance of CMM, especially in the automotive sector

In January 2001 a first draft of a roadmap contained the quite competetive goal of reaching CMM Level 2 within 15 months of process improvement. After 7 months of project planning and preparation (milestones "CMM Project Kick off" to "Design Release CMM" in Figure 2) there were another six milestones planed for the six key processes to be defined. According to the plan there would have been five month left for piloting and introducing the new processes. For December 2002 there was a short-assessment planed to verify that the projects work in conformance to CMM Level 2.

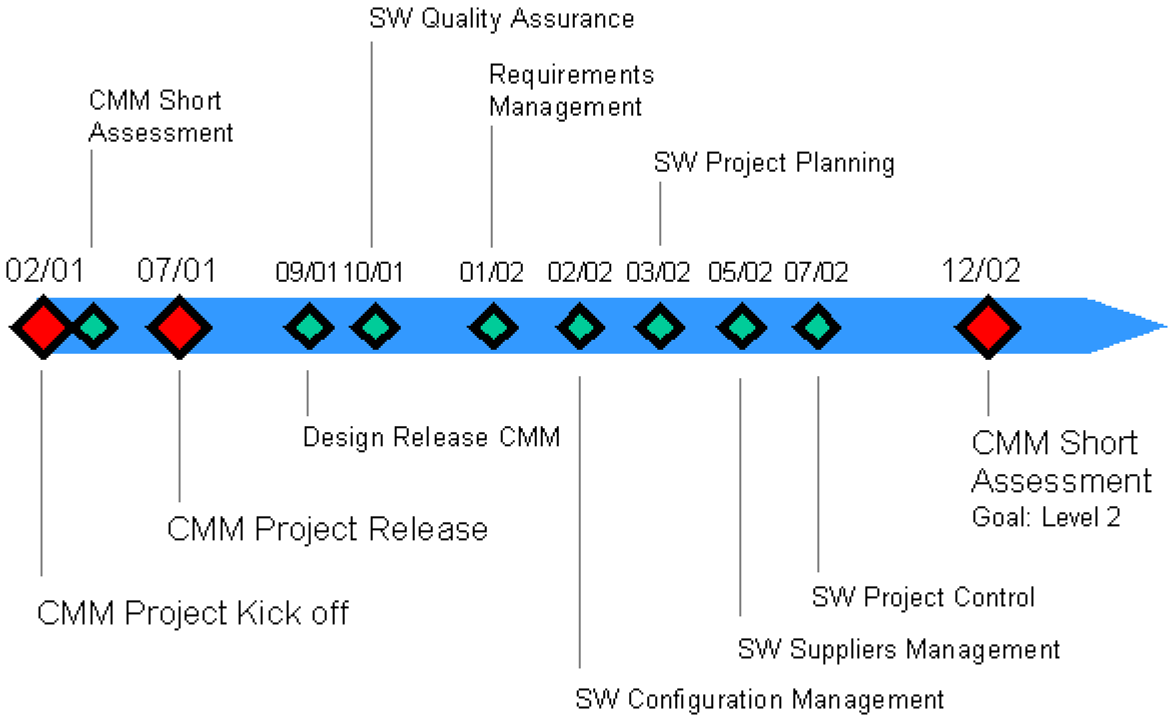


Figure 2: CMM project roadmap in January 2001

2 Approach

For the conception and realisation of process improvement external help was brought in. At first, the six key processes of CMM were used to define three packages, each containing two key processes [2,3,6]. Two consulting companies were assigned, each with the purpose to handle one package. The third package was put off. Method park was one of these two consulting companies, and started supporting the CMM project in September 2001.

It was continuously necessary to motivate as many key employees as possible to get them involved in the improvement process. Some SPI basics were missing: definition of roles, phases, required artefacts, corresponding responsibilities and methods. Afterwards the key processes were discussed separately in several workshops.

These additional tasks for perfoming these SPI basics caused a delay in the original roadmap. Thus, the goal for 2002 had to be redefined: instead of passing a short-assessment at the end of 2002, which would have required established processes, all CMM key processes had to be at least ready for piloting (Figure 3).

In April 2002 the third consulting package and some other tasks were assigned to us, as well. The other consulting company will conduct the short-assessment in the first quarter of 2004. The fact that two consulting companies were involved certainly created some redundancies and small difficulties but made it easier for the customer to compare the services and use this experience for later offers.

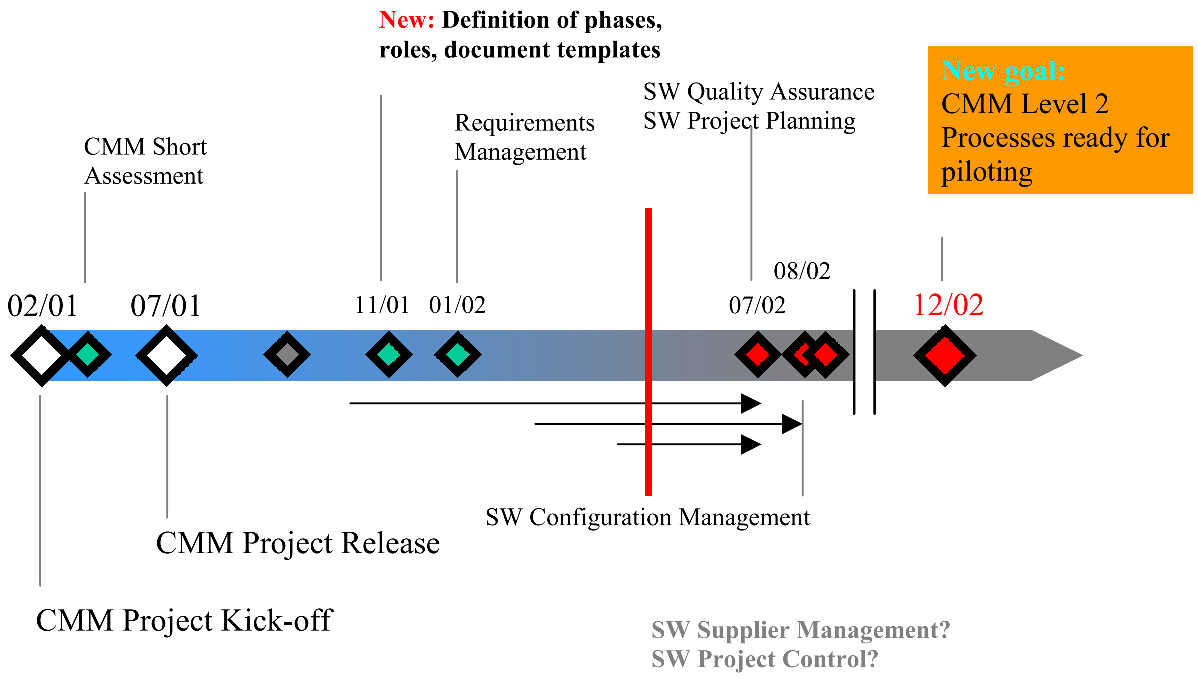


Figure 3: CMM project roadmap in April 2002

2.1 Workshops

In average three workshops were needed to define a key process. The first workshop introduced the activities of the key processes and some approaches to meet their requirements. Then, all participants cooperated for drafting an approach how to meet the customers special business needs. Subsequently, tasks such as creating document templates, examples, check lists, and step-by-step descriptions with guidelines were assigned to the participants.

The agenda of a workshop typically contained the following:

- Introduction of the key process goals, activities, and approaches
- Draft of the company specific process definition
- Assignment of tasks
- Open issues
- Workshop feedback.

The latter made it possible to evaluate the success of the workshop and optimize the agenda.

In the second workshop the results of the tasks were discussed and – if necessary – redefined, but this time more detailed. The third workshop was used to re-involve the affected groups and ensure their commitment by inspecting the results. Additional requirements were listed, as well. Then, the descriptions, guidelines, templates, examples, and check lists were published in the online process workbench where they were made available to the entire team (**Figure 4**). According to the limited resources of the customer a rhythm of two workshops each month worked best. This made it possible to define up to two key processes within a three-month period.

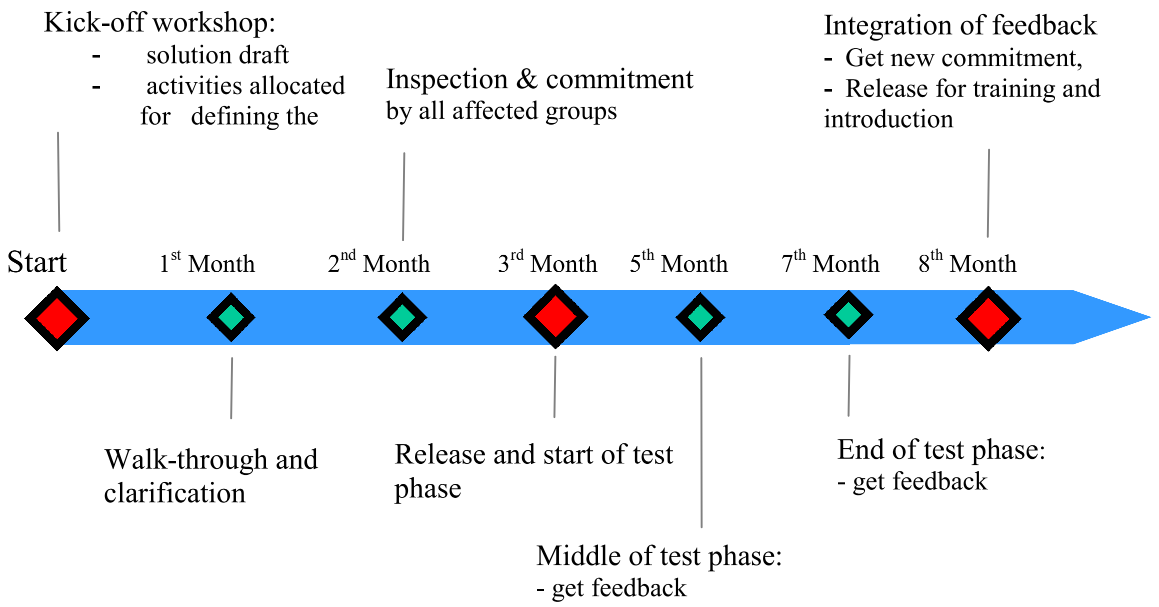


Figure 4: **Roadmap for conception and introduction of one key process**

2.2 Piloting and Training

The piloting of the key process "requirement management" started in 2002. The aim of the pilot project was to refine, optimize, and adapt the defined process to the special business needs of the development departments. Three projects were selected for piloting one key process. These projects were estimated to take about 12% additional time and resources for the affected activities.

At the beginning of the piloting, the required tools were evaluated, installed, and the respective people trained. The piloting team was trained in the new process, as well. During the piloting, feedback was systematically collected on a regular basis. This is important because the team members found the more constructive suggestions for improvement not after but during the pilot phase. As soon as possible, this feedback were integrated carefully into the process definitions and templates.

If the company's culture implicitly or explicitly blames employees for mistakes, huge resistance and unusable feedback has to be expected. Often this happens covertly. This kind of behaviour must be replaced by a culture of blaming the process instead of the person.

After successful piloting the team become coaches for the roll out. A suitable exposure for the new processes can help by making the piloting successes known and valued.

Trainings were adapted to the different situations and roles of the staff. The training program takes into account both the specific training needs of each role and already existing specific knowledge. As a result for each process area there are basic trainings on the one hand and special ones – in-depth trainings – on the other hand.

3 The milestone at the End of 2002

In December 2002 we reached the main milestone of the year. All process areas of CMM Level 2 have been defined. The descriptions, templates, and check lists have been revised and are ready for use. Further successes have been:

- The staff has supported the quality enhancing activities greatly. Despite a full workload they still have found the time to prepare for and participate in workshops and reviews, and finish their additional tasks in time.

- The staff has shown a surprising reaction to a personnel change: the manager of the CMM project left in mid 2002. The improvement team took over responsibility for the project very quickly. Especially the people who had been critical at the beginning were then ready to fully cooperate within the project. Obviously they recognised and valued the use of the project.
- A better improvement culture was established. A high amount of both positive and constructive feedback has been given. For example improvement proposals have been posted and managed in a list which is open to everybody. In addition, feedback is required regularly and systematically, and these proposals are integrated into the process descriptions.
- The new descriptions and templates were in use even before they were reviewed, piloted, or released. Instead of waiting for the release, the staff started using the templates, simply because they recognised their value.
- Reviews were performed more intensive and more efficient than they were in 2001.
- The newly developed tool for estimating efforts has been used several times. It was found to deviate from actual efforts no more than 10 percent. Thus, it is more reliable than a “rule of thumb” method.
- In the fall of 2002, an OEM conducted the first audit at the customer. 467 of 500 points were reached on the automotive manufacturer’s scale.

4 Extended Goals

These successes made it possible to both continue the existing project and define new goals for 2003. Starting in January 2003, the test and design processes will be looked at in more detail. Also, people from outside the current focus have shown interest in our approach. The hardware development team wants to use the defined processes, as well. Of course the processes need to be enlarged to meet the hardware-specific needs (Figure 5). A further department, responsible for developing the controls for another product, also wants to adapt and use the descriptions and processes for their purposes.

When you start changing things, it can have surprising “side effects”. The customer’s employees have started using the defined processes in areas for which those processes had not been developed. For example they used risk analysis not only for development projects, but also for the CMM project itself. The main risk for this project was, whether the management would accept, that the main project efforts will be shifted to the start of the project. It has to be accepted that a lot of work is done in requirements engineering, project management and reviews of several documents, before the engineers start to implement the system. Since some of the greatest risks concerned the management, the boss was involved to help developing suitable counter-measures (Figure 6).

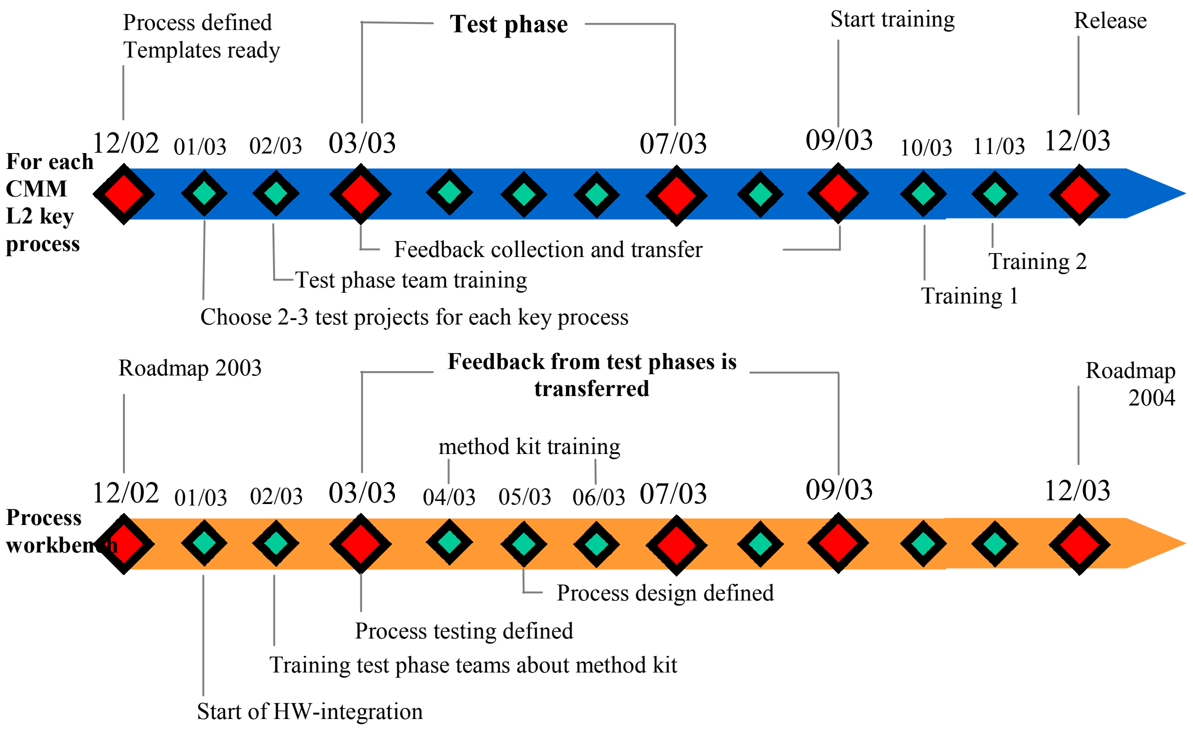


Figure 5: CMM project: roadmap for 2003 (Status in December 2002)

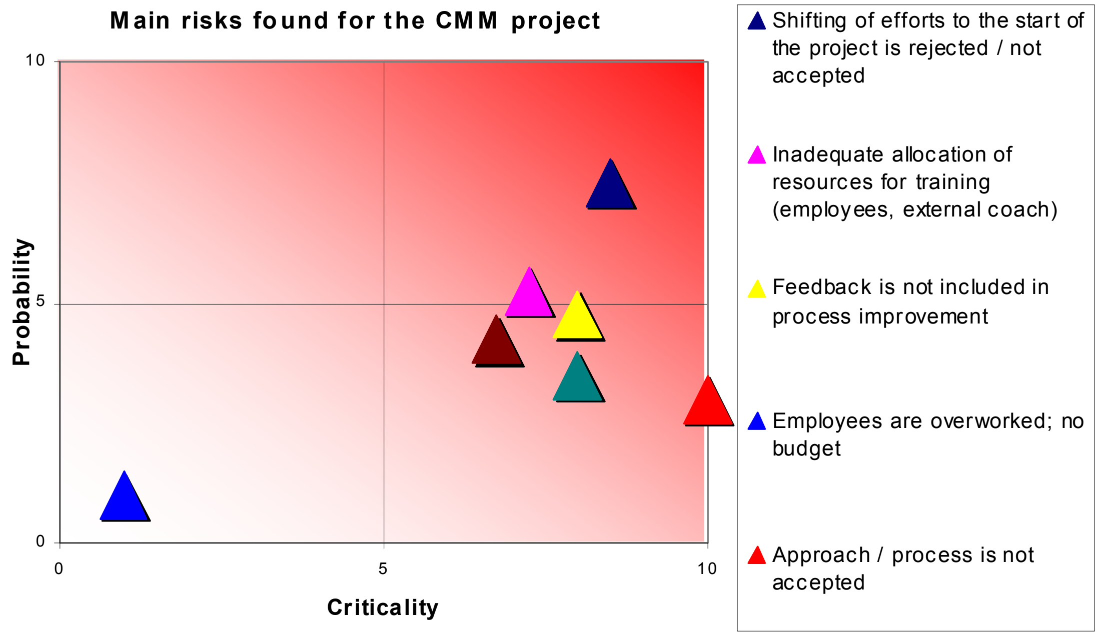


Figure 6: Main risks found for the CMM project in 2003 (Status in December 2002)

5 Success Factors

Management commitment is the key for success. If the process improvement project is not at least as important as a customer project, the employees will always have (or will always be given) more pressing things to do. Even if the management does not say it explicitly, the employees soon find out whether they are measured more by the success of the customer project or by the success of the process improvement project. The management should regularly show its support for the new process.

Organizational measures can additionally support the success of the project. Those affected by the project should become involved or at least be adequately informed about the goals and ways to reach them in a kick-off meeting. A project manager coordinates the different internal and external jobs. Somebody must be responsible for each key process and must be a contact person for questions regarding corresponding activities. These “responsibles” are involved in the definition of the approach for this area and maintain the descriptions, templates, examples, and check lists.

Another important success factor is to plan deadlines and resources realistically. Every month, this project required about 0.4 person months of external consulting (without training) and 1.2 to 1.5 person months of internal efforts. Experience has shown that even though such a process can be implemented with little effort it will not be adequately adapted to the company’s needs. Therefore, the employees will only stick to the process on paper (if at all); it will not be maintained and improved adequately.

Good process and product metrics support the development. Not everything that one thinks it should be measured, needs really to be measured. Rather, the progress of the entire project and the subprojects should be made transparent in a simple, sensible way. This applies to both customer and process improvement projects. Other success factors were, for example, the company’s culture in regard to mistakes, regular collection of feedback, and piloting, as described above.

6 Summary and Outlook

Whoever starts defining and measuring will initiate change. It is important to use these changes for a cultural shift by establishing improved processes step-by-step. Changes must not become a matter of form or a fashion symptom, but the new activities should become a characteristic of a cultural change (Figure 7).

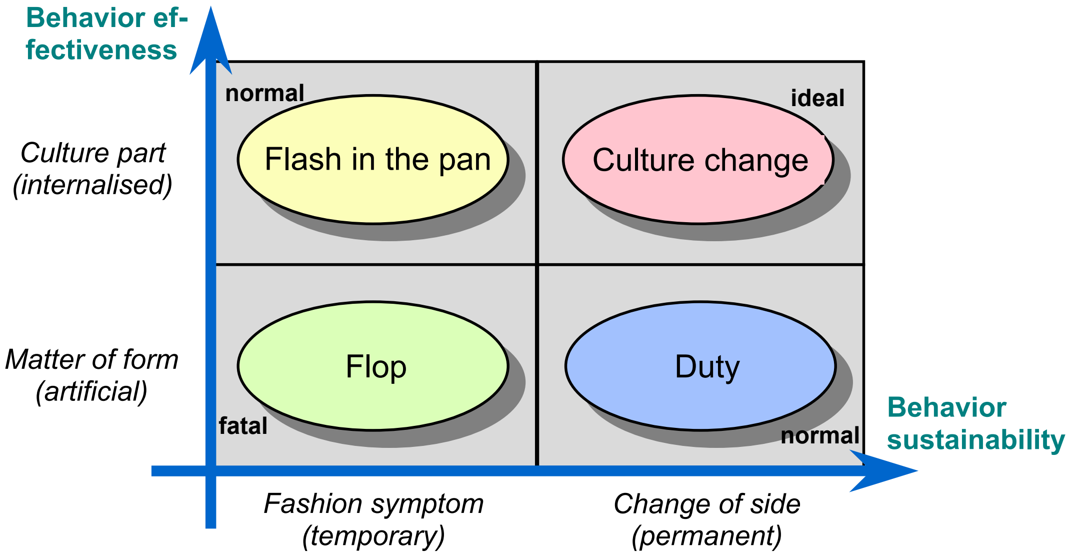

Behavior effectiveness
Culture part (internalised)
Matter of form (artificial)
normal
Flash in the pan
ideal
Culture change
Flop
fatal
Duty
normal
Behavior sustainability
Fashion symptom (temporary)
Change of side (permanent)

Figure 7: The chance for cultural change

“The only person pleased with changes is a baby with wet diapers [9].” Therefore it is essential to show employees how to benefit personally. The achieved results and successes need to be communicated clearly. In the project outlined in this paper, the employees have recognized the advantages of the new processes, already make intensive use of the templates and checklists, and give a lot of constructive feedback, which makes it possible to further improve the descriptions. As soon as the new processes have been established, the focus will shift to the Capability Maturity Model Integration (CMMI) Level 3 [4,5]. The great support gives reason to hope that the employees will continue to accept changes instead of having to be convinced to do so. Therefore it is reasonable to expect that the process improvement project will continue successfully.

7 Literature

[1] Ahern, Dennis M., CMMI Distilled: A Practical Introduction to Integrated Process Improvement, Addison-Wesley Pub Co, 2001, ISBN: 0201735008

[2] Caputo, K. (1998), CMM Implementation Guide, Addison Wesley Longman

[3] Chrissis, Mary Beth, CMMI: Guidelines for Process Integration and Product Improvement, Addison Wesley Professional, 2003, ISBN: 0321154967

[4] Kneuper, Ralf (2003), CMMI: Verbesserung von Softwareprozessen mit Capability Maturity Model Integration, dpunkt Verlag

[5] Knüvener, Christian (2003), Eine Gegenüberstellung von CMM und CMMI, F.A.S.T. GmbH & ASQF, Reihe Softwaretechnik, Band 13, “Software Process Improvement: CMM & SPICE in Practice“, Verlag UNI-Druck, München, ISBN-3-87821-323-9

[6] Paulk, M. C., Weber, V. B., Garcia, S. M., Chrissis, M. B., Bush, M. (1993). Key Practices of the Capability Maturity Model, Version 1.1, Pittsburg: Carnegy Mellon University

[7] Potter, Neil S., et al, Makting Process Improvement Work, Addison-Wesley, ISBN: 0-201-77577-8

[8] Software Engineering Institute, Introduction to the Capability Maturity Model – Integrated Staged Representation, Pittsburg: Carnegy Mellon University. www.sei.cmu.edu

[9] Klaus Kobjol, winner of the European Quality Awards 1998

Author's CV

Christian Knüvener

Christian Knüvener studied computer science at the University of Erlangen-Nürnberg. Trainings focusing on software quality and software process improvement gave him his specialist knowledge. Software development projects, which were mainly customer driven, were the base of experience for several projects to norm software product families, software installation and software maintenance concepts. The establishment of software quality departments, integration of software tests and software quality activities into the software development process were all performed focusing on efficient use of software quality norms and software quality models. He became a SPICE assessor, and is also specialised in CMM and CMMI. He is the manager of the ASQF special interest group maturity models. He is currently working in the consulting, training, coaching and performance assessments of software process improvements, software quality and software tests.

Using the eSourcing Capability Model[1] to improve IT enabled business process outsourcing services

Dr. Miklós Bíró, Ms. Gáborné Deák, Mr. János Ivanyos, Dr. Richard Messnarz, Ms. Ágnes Zámori

Abstract

Memolux, a leading Hungarian accounting and software development firm has run software process improvement practice for more than ten years. The ICT department achieved maturity level 3 of Bootstrap methodology and ISO 9001 certification by 1998 and has been successfully participating in EU research projects. Based on the lessons learnt from process improvement, a development project called PASS3Mill was launched to support the company's other main business activity, the payroll outsourcing service. The effects of the development on the outsourcing business environment were measured against the framework of the eSourcing Capability Model for IT-enabled Service Providers (e^{scm}) and this article summarises the results and consequences by presenting practical knowledge management experiments. Specific attention is paid to focus on the specialities coming from the outsourcing type and the SME size of the service company. The authors present examples of the practical usage of the e^{scm} practices and describe how the SPI knowledge and the supporting tools can be used to design and set up new business process outsourcing services like monitoring of collaborative projects in the accession countries.

Keywords

eSourcing Capability Maturity Model, Process Improvement Experiences, Self-Assessment

[1] The eSourcing Capability Model is a Trademark of the Carnegie Mellon University, Pittsburgh, USA

1 Experiences with SPI in Memolux

Memolux, established in 1989, is a Hungarian private SME company with professional experience as a service provider in finance and public accountancy, management organization, software development and information system engineering. In Hungary, Memolux is ranked after the "Big Four", the four greater advisory firms in public accountancy. Memolux is a member of several economic chambers (AMCHAM, BCHH, CCCH). The payroll and accounting service lines represent the biggest one in the Hungarian market provided by independent Hungarian SME with about 150 clients.

Memolux has run software process improvement practice for more than ten years. The ICT department achieved maturity level 3 of Bootstrap methodology[1] and ISO 9001[2] certification by 1998 and has been successfully participating in EU research projects. Memolux was the prime user and contractor of the PASS project, which was the first Central and Eastern European ESSI Process Improvement Experiment (FP4 PIE) project directly supported by the European Commission[3]. Memolux was a co-developer of the Media-Information sans Frontiéres (NQA based teamwork) system and was the technical coordinator of the Media-ISF Best Practice (FP5 IST Take-up) project[4].

The baseline PASS project to the software process improvement activities started in 1997. Its business purpose was to develop a modular, platform independent, integrated networked software system satisfying the functional requirements of EU standards in public accountancy and applicable for the Hungarian as well as for the international market.

Objectives and expected results were improving the control of the development process (QA Unit, structured system analysis, improved testing process, efficient project planning, ISO 9001 documentation), raising the maturity level to 3 and achieving compliance with ISO 9001 requirements at this high level of maturity.

In the PIE project the quality of Memolux software development processes was enhanced to become well defined and predictable, and by the dissemination of EuroSPI conferences in 1997 and 1998, this PIE was used as a master example to adapt Eastern European processes to the high quality norms of Western Europe, this way facilitating the integration of Eastern Europe into a joint EU in the long term.

The lessons learnt from the above experiment were the following:

- The approach of considering the improvement of the maturity level as the principal objective and the achievement of ISO 9001 certification as a side-effect was valid from the efficiency point of view.
- Even if ISO 9001 certification was not the principal objective of process improvement, it might be worth capitalizing on its high recognition by allocating appropriate resources to its achievement.
- According to international experiences, there is usually a significant decline of attention towards the quality system after the ISO 9001 certificate is granted. The approach of considering certification as a side-effect of overall process improvement helped to avoid this trap.

2 Strategic Project for Improving Business Services – PASS3Mill

The company built its success around the accounting and payroll outsourcing needs of Hungarian and foreign start-up companies following the social and economic transformation of the 1990's. The full time professional staff, the nimble organization, the innovative culture and their strong IT foundation enabled Memolux to maintain a stable growth and to adopt quickly to changing market requirements.

Due to a conscious and consistent integration of business and technology development efforts in the company's strategy, Memolux was able to build and maintain a competitive advantage in its markets. In 1999 the new project called PASS3Mill[5] started based on the new definition of the company's Payroll business strategy[6]. By the help of the ICT staff the management of Memolux Payroll Depart ment developed a vision listing the major characteristics of the business Memolux wanted to run:

- Maintaining the current (40%) growth rate of the business line based on continuous performance improvement, improved risk management and the introduction of new value-added services
- Maintaining the current customer base while targeting new market segments and services
- Personalized services for client HR staff and management – like one stop shopping
- Providing self service opportunities for client staff and management – from accessing data and basic services to professional consulting
- A modular design of service offerings – to enable customised services to individual needs
- A nimble organizational structure – that allows quick market response and dynamic growth
- The operation of a sophisticated quality assurance system that enables the company to offer the best quality service on the market
- Best value – to become the best service provider at the lowest price
- Low-risk service acquisition – generic and reliable processes that will accommodate the seamless transfer/acquisition of outsourced client payroll activities
- Integration of client and Memolux processes – setting up processes that can quickly adopt to changes made within client organizations and practices
- A full integration of business processes and technology – every step of the business process is supported by information technology

The payroll accounting service system should be based on the Hungarian legal regulations and European standards and achieve:

- that requests from Memolux clients are fulfilled in a way that elicits the greatest possible contentment and within the shortest possible period of time
- that Memolux offers tailor-made solutions for big companies employing over 1000 employees, too
- that the service is provided on the basis of traceable and documented procedures, via methodical and professional administration
- that the IT support of the service is based on the most modern possible background, with the involvement of the Information and Communication Technology Department and the employment of the most modern available technological opportunities

The aim of the PASS3Mill was the implementation of business processes fulfilling defined quality standards and using IT solutions employing the new technological opportunities. The main tasks of the PASS3Mill project were:

- The careful examination of the processes of the Payroll Accounting business branch of Memolux and improvements re-using present successful processes from the previous process improvement initiatives.
- The extension of the scope of the quality assurance system (compliant with ISO 9001), to cover the entire business segment of the Payroll Accounting and ICT Departments applying defined e-services processes.
- The Re-organisation of the IT service provided for the Payroll Accounting Department, which is a significant extension compared to the previous IT support, system-development and system-maintenance activity.

The planning of each improvement step within the PASS3Mill project started with the business risk analysis to select those areas of IT development, which meant the greatest risk for the payroll accounting service. The risk analysis[7] had to be done on the basis of the “Payroll Accounting Functional Process Description”[8] measuring the effect of the following risk factors on the business processes:

- growth of volume
- change of the method of work
- the enlargement of the service activity
- non-compliance of the IT support
- change of technology

Measurement was done by applying 3 (0-2) risk categories for all of the above mentioned factors in relation with the primary processes.

0. the risk factor has no direct impact on the efficiency of the business process belonging to the primary process.
1. if the effect of the risk element is direct, but does not have decisive influence on the result of business processes
2. if the effect of the risk element is direct and decisive in respect of business processes

Adding up risk values attached to primary processes, we got impact values in a scale from 0 to 10.

Based on the above, rates 9-10 characterized high-risk primary processes, which had to be dealt with in the succeeding improvement step/phase, i.e. a certain new improvements had to be made on the processes as well as on the underlying technology solution, so that the risk level should decrease in the shortest possible time frame. If primary processes reached rates 7-8, only the refinement of processes had to be ensured, while the change of the underlying IT solutions was postponed until the next improvement step. Primary processes with rate 6 had to be taken care of only when the tasks related to a higher risk category had been already solved.

By the end of 2002, the PASS3Mill project achieved its main objectives by having maximum 6 of risk value level for any of the primary business processes. This meant that the ICT related risk factors were now under control.

The effects of the PASS3Mill project on the capability of the outsourcing business environment were measured against the framework of the eSourcing Capability Model for IT-enabled Service Providers (e^{scm}) [9] in the beginning of 2003.

3 The e^{scm} Self Assessment

The e^{scm} framework contains nearly two hundred practices for IT enabled outsourcing providers that address the critical capabilities of the service. All the practices are organised into three outsourcing phases and five organisational elements and each practice is associated with a capability level (see Figure 1).

The objective of the self assessment was to define the measurable effects of the previous PASS3Mill project. In 1999 when PASS3Mill project started no applicable capability models (such as e^{scm}) were available and could be applied to define the level and the necessary improvement steps of business related services such as the payroll outsourcing service.

From business terms the PASS3Mill project achieved the main goal of providing enabled IT technology and process support for the redesigned outsourcing business process at predictable cost. During the migration period the fast growing (40% per year) volume of business transactions were processed by a much lower increase (12% per year) of human resources.

The business success motivated us to investigate the capability aspects of the improvement, and to make visible our results applying the e^{scm} model.

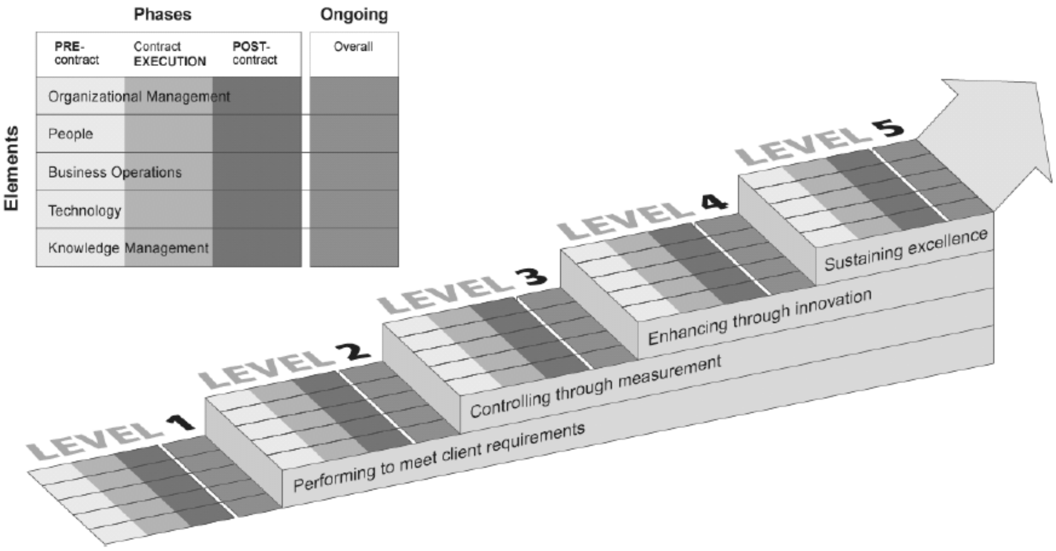


Figure 1: The e^{scm} Framework

The self assessment used the e^{scm} practice descriptions as a basis. The assessment kept in mind not just to define a capability level, but to measure the effects of the performed PASS3Mill project on the achievement of e^{scm} levels.

We also tailored the practice definitions, when they were not adequate to the size (25 people) of the outsourcing organization or not applicable to the business (payroll outsourcing) characteristics. The self assessment results and derived new improvement opportunities are summarized in form of a fulfilment map:

		Organizational Management	People	Business Operation	Technology	Knowledge Management	
4	Overall	●●●●●	○○	●	●	●	9 (11)
	PREcontract						
	Contract EXECUTION						
	POSTcontract					●	
3	Overall	●●●●●●●	●●●○●●○	●		○●○●○○○	24 (31)
	PREcontract	●	●			●	
	Contract EXECUTION	●		●●●	●		
	POSTcontract			●			
2	Overall	●○●○	●●●●	●●●●●●●●	●●●●	●	49 (51)
	PREcontract	●●	●	●●●●●		●	
	Contract EXECUTION	●●	●●	●●●●●●●●●●●●●	●		
	POSTcontract		●	●●			
	Totals	20 (22)	14 (18)	34 (34)	7 (7)	5 (12)	82 (93)

Figure 2: The e^{scm} Fulfilment Map of Memolux Payroll Outsourcing Services

The effect of PASS3Mill showed a high level achievement (82 from 93) in the implementation of the e^{scm} practices. The practice definitions of e^{scm} were applicable for the Memolux service type.

In general we observed that the relatively small size of the outsourcing organization had an effect on e^{scm} so that more level 4 practices were implemented on the executive level and it was difficult to fulfil some of the level 2 practices.

The clear and well-defined business orientation and strategy underlying PASS3Mill together with the supporting ICT project led to a "full" implementation of the Business Operation and Technology organisational elements within e^{scm}.

In the improvement planning we then focused on e^{scm} practices, which were missing or only partially implemented.

The Technology element does not mean a big issue for further improvement inside Memolux, as the necessary technology practices "just" request a normal subcontracting type contribution and cooperation from a maturity level 3 ICT organisation. However, the further implementation of the People and Knowledge Management practices is a must to achieve the other results of the external effects of the development and support project.

Thus a major focus was put on the knowledge management aspects in the next improvement step.

Capability levels	*ID*	*Practice*
4	*ppl_over_1_4*	Establish and implement a policy to encourage and support innovation and entrepreneurship across the organization.
	ppl_over_13_4	Analyse the effectiveness of the training in achieving its intended objectives.
3	ppl_over_8_3	Establish and implement a policy to provide personnel with opportunities for career development.
	ppl_over_11_3	Establish and implement a reward program that encourages the achievement of organizational objectives.
	km_over_3_3	Establish and implement a knowledge system to identify, control and disseminate information.
	km_over_5_3	Identify and reuse work products.
	km_over_7_3	Analyse and use knowledge about clients and end-user.
	km_over_8_3	Analyse and use feedback from clients and other stakeholders throughout the sourcing process.
	km_over_9_3	Analyse the perceptions that current and prospective clients have about the organization and its services.
2	*org_over_3_2*	Establish and implement a policy on risk management.
	org_over_13_2	Establish and implement disaster recovery procedures.

Figure 3: Identified Gaps by Capability Levels Using e^{scm} Self Assessment

3.1 Observations by Capability Levels

The most of the practices on **level 4** are essentials for an effective, successful business. If they were missing, the result would be a sure business failure. Other interesting findings are that the most practices belong to the responsibility of the Executive Manager in case of a small business company. We think that it will be needed to shift them to other organizational roles in case of a future growth. According to the management policy it is planned to nominate a HR (Human Resource Services) manager responsible for payroll outsourcing related tasks when the payroll staff exceeds the 30 people size.

There are some practices on **level 3**, which the outsourcing service provider organization might not regard as an essential issue, and so a wish for tailoring options might come up.

This is the level, where the most effects of the PASS3Mill projects have been achieved. The missing practices are HR (Human Resource Services) delegation related and knowledge management issues.

The external effort of the development project [5] shall be replaced by the proper knowledge management solutions. This was the result from investigating the fulfilled level 3 practices by subtracting direct operational management-authority practices. Remaining practices need external support, such as process improvement, which is the best example for demonstrating applicability of knowledge management as a potential outsourcing service. The distributed collaboration seems a presumption of e^{scm}.

Practices	Topic	input from	output to	uses activity in
org_over_16_3	process improvement based on reviews			
2.a.	identifying sources, classes of basic information (organization's performance measures) to define improvement	org_over_7_3 org_over_10_3 ops_over_2_2 ops_over_6_3 ops_exe_6_3 tech_exe_2_3 km_over_4_3 km_over_8_3 km_exe_1_2	km_over_3_3	
2.f.	prioritising improvement opportunities based on organizational objectives	org_over_1_3		
2.h.3)	resource definition for improvement plan based on necessary supporting roles, competencies, work environment and information			org_over_4_2 ppl_over_4_2 ppl_over_7_2 km_over_2_2
3.	reviewing implementation actions according to the measurement procedures	org_over_7_3		ops_over_2_2
4.	verifying consistent implementation accross the organization according to the periodically review procedures			org_over_10_3

Figure 4: Derived Process Improvement Map Using e^{scm} Self Assessment Level 3

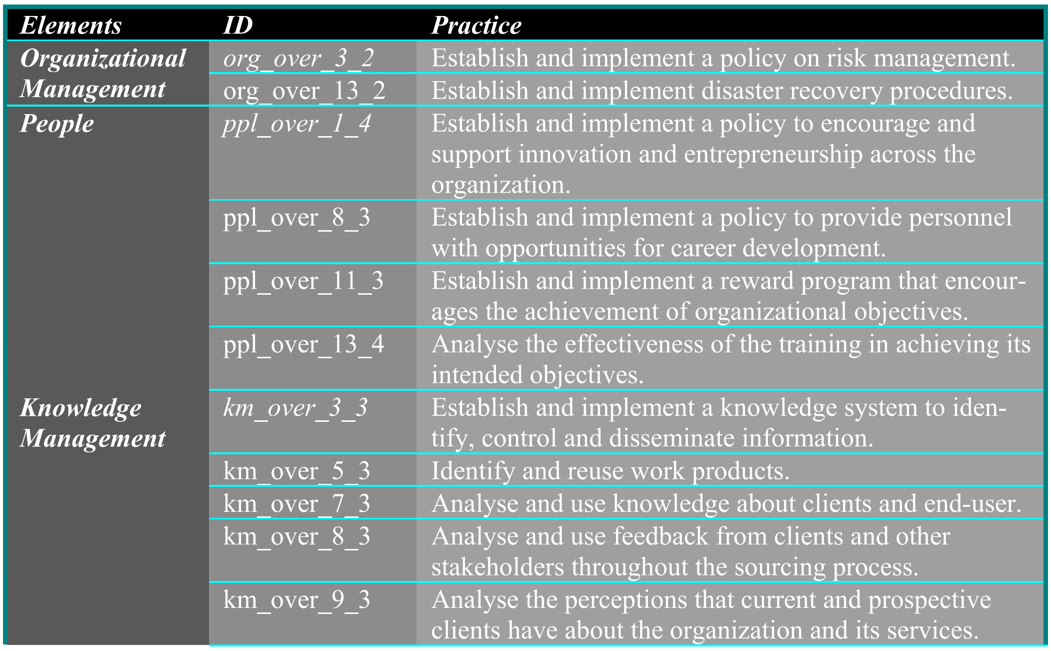
The missing practices of **level 2** (see Figure 3) can refer to different risk levels depending on outsourcing types, such as a bank transfer could not be performed without rigorous replication solutions and if the timeframe and conclusions are not so strict, whereas the disaster recovery is not a high priority topic. For example when the basic payroll activities can be performed manually or at low computerised level like via spreadsheet applications, the disaster recovery plan is based on ensuring the availability of human resources e.g. as overtime work. At a big volume operation the business continuity plan requests availability of high quality IT resources both, in terms of technical and human terms.

3.2 Observations by Organisational Elements

As identified previously, the missing **Organizational Management** practices are very dependent on the outsourcing type. The right balance between human resource and technology dependence shall be considered.

The most **People** practices belong to the responsibility of the Executive Manager at a small business organisation like Memolux Payroll. We think that it will be needed to delegate to an HR Department with the future growing.

Knowledge Management issue is identified as the main organisational element necessary for retaining the former results and improve the outsourcing capabilities. By the closure of the external development ICT project, the question is how the achieved capability level can sustain and further develop.



Elements	*ID*	*Practice*
Organizational Management	*org_over_3_2*	Establish and implement a policy on risk management.
	org_over_13_2	Establish and implement disaster recovery procedures.
People	*ppl_over_1_4*	Establish and implement a policy to encourage and support innovation and entrepreneurship across the organization.
	ppl_over_8_3	Establish and implement a policy to provide personnel with opportunities for career development.
	ppl_over_11_3	Establish and implement a reward program that encourages the achievement of organizational objectives.
	ppl_over_13_4	Analyse the effectiveness of the training in achieving its intended objectives.
Knowledge Management	*km_over_3_3*	Establish and implement a knowledge system to identify, control and disseminate information.
	km_over_5_3	Identify and reuse work products.
	km_over_7_3	Analyse and use knowledge about clients and end-user.
	km_over_8_3	Analyse and use feedback from clients and other stakeholders throughout the sourcing process.
	km_over_9_3	Analyse the perceptions that current and prospective clients have about the organization and its services.

Figure 5: Derived Process Improvement Map by Organisational Elements Using e^{scm} Self Assessment

Based on the self assessment results we then analysed how the knowledge management methods and tools proved in IT environment could be further used.

4 Knowledge Management Focus

The intention of identifying knowledge management experiments was to present how the knowledge management system is important in outsourcing capability. We investigated what are the practices relate to Knowledge Management. We had an assumption, that knowledge management system is a key element to an effective outsourcing because it enables dynamic knowledge sharing.

Practices	Topic	input from	output to	uses activity in
org_over_1_3	organizational objectives		km_over_3_3	
org_over_6_4	improvement program (resources, personnel assignment)			km_over_3_3 km_over_1_2
org_over_7_3	measures (resource consumption)	km_exe_1_2		
org_over_12_3	client relationships (business case, marketing information, relationship building and maintaining – analysing feedback and client perceptions)	km_over_7_3 km_over_8_3 km_over_9_3 km_pre_1_3		
org_over_16_3	process improvement based on reviews	km_over_4_3 km_over_8_3 km_exe_1_2	km_over_3_3	km_over_2_2
org_over_15_4	value creation for clients analysis			km_over_7_3
org_pre_1_3	guidelines for negotiations	km_over_7_3 km_pre_1_3		
org_exe_2_3	cultural integration – feedback from the client			km_over_8_3
ppl_over_2_3	competencies needed to achieve organizational objectives	km_over_6_3		
ppl_over_3_3	developing workforce competencies -competency inventory		km_over_6_3	
ppl_over_5_3	planning and delivery training-competency inventory		km_over_6_3	
ppl_over_8_3	career development – feedback from personnel			km_over_8_3
ppl_over_12_2	information needed for client interaction team			km_over_2_2
ppl_pre_1_3	information required service team	km_over_6_3 km_over_7_3		km_over_2_2

Practices	Topic	input from	output to	uses activity in
		km_over_9_3 km_pre_1_3		
ppl_pre_2_2	information required negotiation team			km_over_2_2
ops_over_6_3	identifying, documenting and analysing problems – knowledge systemization	km_over_4_3 km_over_8_3	km_over_4_3	
ops_over_7_4	proactively identifying and preventing problems – data gathering	km_over_4_3 km_over_8_3		
ops_exe_1_3	reusable work products and service components	km_over_5_3		
ops_exe_2_2	planning to design and deploy the service (efforts)	km_exe_1_2	km_exe_1_2	
ops_exe_3_2	tracking plan to design and deploy the service (efforts)		km_exe_1_2	
ops_exe_5_2	designing processes, policies and procedures of service	km_over_2_2		
ops_exe_6_3	potential failure handling procedure and documentation		km_over_4_3	
ops_exe_15_2	resource estimation of the service, compliance tracking	km_exe_1_2	km_exe_1_2	
ops_post_2_3	service continuity – lessons learned passing to the client	km_over_4_3		
tech_exe_2_3	identifying technology gaps		km_over_4_3	

Figure 6: Relationship Between Knowledge Management and e^{scm} Practices

One of the identified main topics is about the **Client Relationship Management**. It requires managing knowledge regarding the client and end-user needs, feedbacks and perceptions of the current and prospective clients about the organization and its services. This knowledge would help defining the current and future market position and expectations. This issue was mainly handled at the executive management level in Memolux payroll service so far, and has in future to be delegated and further supported during the organisational growth.

Another major topic concerning the organizational processes is the people assignment issue, where a knowledge management system could firmly support the right **Competency Analysis and Quality Assurance**. Knowledge Management should provide the right access rules, extract necessary information to personnel about the client, service meetings and client expectations, and also legal requirements. We also found that Knowledge Management has a dominant role in **Service Control and Improvement** as storing and sharing information about lessons learned.

4.1 Client Relationship Management

The CRM is always a critical issue for the outsourcing business processes. The client keeps high level expectations concerning the quality and availability of products of the outsourced services, and usually clients are very reluctant to fulfil all quality requirements concerning their own contribution. The expectations of both sides are usually documented in a Service Level Agreement and it is very important to measure the clients' contribution (resource, data, etc.) the same way as they measure the service outputs.

Any type of communication between client and provider shall be documented and evaluated. At a minimum this includes the common minimum level interface between client and provider, a commonly accepted communication method shall be set up and performed based on the practice of the worst human reactions. The right knowledge management system fed by all the communication history data provides not only the applicable communication rules, but also a measure of client contribution performance and ability.

The knowledge management system receives all historical information regarding inputs and outputs of the service and thus provides a basis to review data and decisions. Before that the steps remain quite dependent on human factors. As the client has its own ability to even analyse his own failures, the overall service process will improve significantly. In general the outsourcing service can achieve a high capability level under the assumption that agreed requirements are kept on the client side without changing the contract or resources. The knowledge management can provide a framework and assessment method to control not only the providers' procedures, but the clients' as well. (e.g. late changes in requirements or contracts become visible and can be analysed from both sides)

4.2 Competency Analysis and Assurance by NQA/TEAMWORK method and tool

The Competency Analysis and Assurance principles and methods provided by the NQA (Network based Quality Assurance) [10a, 10b, 10c] knowledge management applications had been proved and used by the Memolux ICT department for many years. Based on the process improvement experience of the NQA concept focusing on role centred team work management paradigms for quality management in the field of distributed virtual work environments, the next step was to apply it for the outsourcing services as well.

The NQA approach is based on the following three principles:

- Role and information flow based team work process management
- Development of products by configuration
- Re-Use pool concept

The production of any project deliverable is not seen as just a sequence of tasks with a planned result, but it is the result of an integrated teamwork environment. The organisation is broken down into work scenarios and each scenario is designed with

- Roles who have responsibilities
- Work steps to which roles and resources are assigned
- A network of work steps forming a work-scenario
- Results produced by roles performing a certain work step in the work scenario

The information and communication technology also supports selecting, measuring and controlling the processes.

The required technologies should support also knowledge and information storage, for de-centralised information access and retrieval, as well as for the short-term merging of distributed knowledge.

The selected platform for managing the outsourcing competence assurance has a great deal of built-in functionality including the following:

- A dynamic structure for presentation of documents and links that allows customised views of information
- A clear separation of information and its presentation
- Documents are stored as objects containing information, metadata and functions
- A built-in user and group based security mechanism (access control)
- A scalable architecture for connecting many information servers together into one server pool
- A channel mechanism that supports passive information retrieval (notification)
- Users can create new information through linking documents together
- Collaborative authoring is supported with integrated document versioning, locking, and configuration management
- Object Oriented programming methods allow development of new applications and/or the extension of existing functions

Memolux management found NQA/TEAMWORK as the most promising method and tool to further implementation of the People and Knowledge Management practices in order to retain the high capability results and effects of the external development and support project.

4.3 Service Control and Improvement

The lessons learnt during improvement of our Payroll outsourcing service show us the importance of clients' attitude and capability how to perceive our initiatives. It is the reason why we think that escm capability determination and improvement should be extend to so-called IT enabled clients. This is the point where we refer back to earlier publications about virtual organisations (Knowledge Enterprise model) concerning Clients Relationship Management and Competence Assurance in the Service Control and Improvement. As the level 3 practices of IT enabled outsourcing cooperation request the same level of measurement of contribution and participation of both client and service provider sides, the outsourcing business model can be easily adapted to the below described Knowledge Enterprise business model implemented by the MEDIA-ISF Best Practice Action (IST 5FP) [4].

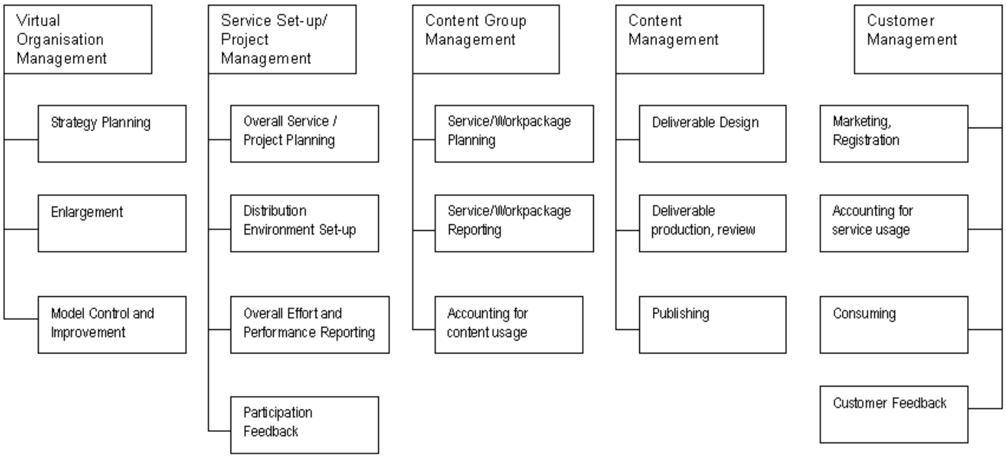

Virtual Organisation Management
Strategy Planning
Enlargement
Model Control and Improvement
Service Set-up/ Project Management
Overall Service / Project Planning
Distribution Environment Set-up
Overall Effort and Performance Reporting
Participation Feedback
Content Group Management
Service/Workpackage Planning
Service/Workpackage Reporting
Accounting for content usage
Content Management
Deliverable Design
Deliverable production, review
Publishing
Customer Management
Marketing, Registration
Accounting for service usage
Consuming
Customer Feedback

Figure 7: Knowledge Enterprise Business Model

Here we highlight the importance of the level 4 practice of Knowledge Management: "**Establish and implement procedures to transfer to the client, during contract completion, the knowledge gained from the specific client engagement**" (km_post_1_4).

We found that the e^{scm} framework can be further extended by implementing the "Accounting for service usage" process of the above Knowledge Enterprise business model. This type of activity is just mentioned in this post contract phase e^{scm} practice as "**ensuring verification and accounting of knowledge transferred**", however this activity is periodically (monthly or yearly) performed in a typical outsourcing service.

The additional finance model to the referred Knowledge Management practice can be simply based on user subscription and/or transaction fees. The used technology platform for IT enabled outsourcing services shall have two main functions from financial accounting point of view:

- Measuring and monitoring the contributions from the different knowledge authors (both clients and service providers) cooperating in the content (electronic form of knowledge elements) re-use and production workflows
- Measuring and monitoring the usage of the common knowledge pool

By such measures the content-based service delivery chain can be simply followed by the transactions performed by the active users of the implemented outsourcing services. The monitoring concept of the electronic content (electronic form of knowledge elements) distribution and usage processes assumes the common understanding of the quality requirements and its support via teamwork technology, which leads to higher performance and outsourcing effectiveness.

This knowledge management focus and use of a technology platform for supporting this led to the design and exploitation of even new business services.

5 Applying e^{scm} Practices for Knowledge Enterprise Services

There are four main lessons learnt from using e^{scm} model to assess Memolux outsourcing capability:

- High capability level (3) practices can't be achieved without external process improvement support.
- The high capability level can't be kept without running knowledge management system.
- The practices of the e^{scm} framework are well adaptable for any virtual organisation model as the high capability level outsourcing cooperation of service clients and providers implements a real knowledge-based virtual organisation.
- The verification and accounting of the transferred knowledge resources are critical issues for IT enabled outsourcing sustainability.

Based on the lessons learnt, Memolux is focusing on how its process improvement skills and the adapted technology having been developed for many years can be utilized as a professional new outsourcing service. One of the business intentions is to provide knowledge-based skill and technology support for SMEs and NGOs planning to participate in public-private cooperation projects from the accession countries. The new outsourcing service organisation is naturally running on the knowledge management platform, which is part of the service itself. The uniqueness of the proposed service is that it assumes cross European partnerships both in the service providing organisation and the users sides.

By using the business process outsourcing Services as a delivery mechanism of the knowledge management related best practices for the accession SMEs, NGOs, public bodies, etc., the objectives of the new knowledge management outsourcing service are the following:

- Improving the impact of gained knowledge in the integrated (enlarged) Europe
- Providing critical mass and flexibility
- Simplifying procedures for fast organisational reactivity

The implemented Knowledge Enterprise services provide the accession communities a gateway to:

- New collaboration relationships at European level
- Teamwork tools and processes for inter- and intra-enterprise cooperation
- Flexible organisational structures empowering knowledge communities and radical innovation
- Product & service design tools supporting their own business models

These Knowledge Enterprise methods and tools have not been available for the accession business communities. They still have been the advantage of the big multinational consulting network-organisations, which have no real solution to the local SME and NGO business environments in the enlargement countries. However successful local business process outsourcing firms have been growing in size and services in the accession countries as well, frequently joining to international groups to scope with the requests of globalisation. These local service providers are aiming at the SME and NGO communities as well, by offering value added services at reasonable prices via traditional outsourcing business models.

Based on the tailored requirements of the fundamental quality standards (ISO9001[2], ISO12207[11], AQAP160[12], AQAP170[13] and EFQM (European Foundation for Quality Management)[14]) the new business service model is following already proven valuable standards, while its basic content - knowledge and intellectual capital management for SMEs and NGOs - offers important contribution for the innovation strategy especially in virtual production, supply chain and life-cycle management, interactive decision-aid systems, development and rapid manufacturing.

6 Summary and Outlook

The ICT business line of Memolux was looking at several innovative business ideas emerging around the concept of the knowledge economy. The former innovation around collaboration technology which led to the NQA network based quality assurance applications, proved to be the most promising path to improve running outsourcing services and to the development of a new service such as Knowledge Enterprise Enlargement Network.

NQA is a Web based knowledge management and collaboration suite that allows setting up high capability level outsourcing organisation with well defined roles and procedures, supported by sophisticated knowledge management and workflow based quality assurance applications.

Outsourcing of Payroll services has allowed Memolux to gain valuable experience about incorporating knowledge management and sharing into its client service offering.

Such a services enable clients to realize the following benefits:

- improved performance – compensating up front investments by reduced operating costs
- effective knowledge and content management – capturing individual knowledge and transforming it into knowledge capital that becomes of a reusable asset of the outsourcing processes
- improved client service – outsourcing organizations are equipped with processes, procedures and tools that can incorporate their clients into these practices without any extra effort
- improved ownership – better motivation of employees through a flat/information based and participative control structure, a radically different management style and a hierarchy based on knowledge
- improved value – in this structure the sum of the parts greatly exceeds their individual worth

Memolux has been using NQA to support its software development activities for years and currently it is extending this knowledge to bring together with new services. The lessons learned and the know-how generated allowed to search for opportunities outside of current business offerings and to turn this newly acquired knowledge and set of skills into other business itself.

One of the next steps along this path for Memolux is to offer its know-how to assist the 3200 member Local Government community of Hungary to set up their own “virtual organization” to share their resources, human capital and knowledge while providing their basic services to citizens and to acquire National and EU funds for their development projects. We believe that the know-how and tools will help them to benefit from the above listed advantages and to make them compliant with the needs of their customers, partners and funding agencies.

Hungary is about to become an equal member of the EU next year, which will generate the need for an outsourcing service SME like Memolux to professionally operate on global level as well.

7 Literature

[1] The Bootstrap Assessment scheme, Bootstrap methodology paragraph www.bootstrap-institute.com

[2] ISO 9001:1994 Quality Systems. Model for quaility assurance in design, development, production, installation and servicing

ISO 9001:2000 Quality management systems. Requirements.

[3] Biró,M.; Ivanyos,J.; Messnarz,R. Pioneering Process Improvement Experiment in Hungary. Software Process: Improvement and Practice (John Wiley & Sons, Ltd.) Volume 5, Issue 4, 2000. Pages: 213-229. (http://www3.interscience.wiley.com/cgi-bin/abstract/76503384/START)

[4] Cs. Baksa, J. Ivanyos, Zs Ziaja, R. Messnarz: Measuring e-business effectiveness: Adaption of process driven approach in web-publishing. EuroSPI 2002, 18th-20th September 2002 in Nurenberg

[5] Project Initiation Report of Pay-Roll Accounting and Settlement System 3rd Millennium (PASS3Mill). May of 1999. {internal}

[6] Payroll Business Strategy. May of 1999. {internal}

[7] Risk AnalysisI, II September 1999, May of 2001. {internal}

[8] Payroll Accounting Functional Process Description, July of 1999. {internal}

[9] Elaine B. Hyder, Bennet Kumar, Vivek Mahendra, Jane Sieges, Keith M. Heston, Rajesh Gupta, Habeeb Mahaboob, Palanivelrajan Subramanian; eSourcing Capability Model (escm) for IT-enabled Service Providers v1.1 21 October 2002; CMU-CS-02-155

[10a] Messnarz R., Stubenrauch R., Melcher M., Bernhard R., Network Based Quality Assurance, in: Proceedings of the 6th European Conference on Quality Assurance, 10-12 April 1999, Vienna , Austria

[10b] Messnarz R., Nadasi G., O'Leary E., Foley B., Experience with Teamwork in Distributed Work Environments, in: Proceedings of the E2001 Conference, E-Work and E-commerce, Novel solutions for a global networked economy, eds. Brian Stanford Smith, Enrica Chiozza, IOS Press, Amsterdam, Berlin, Oxford, Tokyo, Wash-ington, 2001

[10c] M. Biro, R. Messnarz (2000), Key Success Factors for Business Based Improvement, in Software Quality Professional, Volume Two, Issue two, American Society for Quality, March 2000

[11] ISO/IRC 12207:1995 Information technology. Software life cycle processes,

[12] AQAP160 Edition 1 NATO Integrated Quality Requirements for Software throughout the Life Cycle; July 2001

[13] AQAP170 Edition 2 NATO Guide for the Delegation of Government Quality Assurance; September 1997

[14] The EFQM Excellence Model. Companies version, ISBN 90-5236-082-0

8 Author CVs

Dr. Miklós Biró

Dr. Miklós Biró (miklos.biro@bkae.hu) is a professor at the Department of Information Systems of the Budapest University of Economic Sciences and Public Administration with 26 years of software engineering and university teaching (including professorship in the USA), and 16 years of management experience. He has a Ph.D. in mathematics (operations research) from the Loránd Eötvös University in Budapest, an Executive MBA (Master of Business Administration) degree from ESC Rouen, France, and a Master of Science in Management degree from Purdue University, USA. He is fluent in Hungarian, English, and French.

He is a Bootstrap and SPICE (Software Process Improvement and Capability Determination - ISO/IEC 15504) assessor. He gives Ph.D. courses and company training courses on software quality management, and on the Capability Maturity Model - Integrated (CMMI - service mark of Carnegie Mellon University, USA).

He initiated and managed the Hungarian participation in numerous European multinational projects and organisations committed to software process improvement (European Software Institute, Bootstrap Institute). He was the initiator and head of the Information Society Technologies Liaison Office in Hungary for the European Union's 5th Framework Programme. He is invited as expert consultant by Hungarian and international organizations (European Commission; Irish National Policy and Advisory Board for Enterprise, Trade, Science, Technology & Innovation~Forfás; Communications Authority of Hungary; Hungarian Committee for Technological Development; Investment and Trade Development Agency of Hungary; Hungarian Airlines; United Nations Industrial Development Organization UNIDO; International Software Consulting Network).

He has numerous publications in international scientific and professional journals (Software Process Improvement and Practice, Software Quality Professional (1, 2), Software Process Newsletter, European Journal of Operational Research, Zeitschrift für Angewandte Mathematik und Mechanik, Optimization, Information Processing Letters, Discrete Mathematics, Journal of Advanced Transportation, Acta Cybernetica) and conference proceedings. He is the co-author of Hungarian and English language books on operations research models, software engineering, software process improvement and business motivations.

He is member of the Editorial board of the journal Software Process Improvement and Practice published by John Wiley & Sons, and founding president of the professional division for Software Quality Management of the John von Neumann Computer Society. He is the Hungarian member of Technical Committee 2 (TC-2) Software: Theory and practice of the International Federation for Information Processing (IFIP) . He is member of several other professional bodies and societies.

Gáborné Deák

Ms Gáborné Deák (csilla.baksa@memolux.hu) was graduated as mathematics-physics teacher in 1984, and programmer mathematician in 1987. She participated in Computer Information Systems Auditor training in 1998 and got IQA IRCA Lead Assessor and Quality Auditor and Certifier certification in 1999. From 2002 she is Certified ACL (Audit Command Language) trainer.

After working as system designer and analyst for several years she joined to Memolux in 1996. She is responsible for quality assurance, process improvement and project management assistance of Memolux projects, including Whirlpool, EDS, PASS, MEDIA-ISF etc.

János Ivanyos

Mr. János Ivanyos (janos.ivanyos@memolux.hu) is one of the founders of company Memolux and the managing director responsible for Information and Communication Technology support and development since 1989.

He was graduated as an economist at University of Economics, Budapest in 1984. He was working for the Computer Center of the National Planning Office for 4 years and then established his first computer service company.

He managed several IT projects including IT outsourcing services to Unilever Hungary, Heating Works of Budapest, National Employment Office, Ministry of Finance, etc.

He was the Project Manager and Contractor for the PASS Esprit project during the period of 1997-1999. He was the project manager of the MEDIA-ISF (IST-2000-29651) project.

Ágnes Zámori

Ms. Ágnes Zámori (agnes.zamori@memolux.hu) was graduated as Art and Design Manager at School of Applied Arts and Design, Hungary. From 1989 she worked for the Hungarian Academy of Sciences as an Assistant at the UNIDO Office organizing the local trainings of engineers from South-East Asian countries coming to Hungary for development programs. From 1991 she was working for the National Bank of Hungary at the World Bank Department assisting World Bank loans management. From 1998 she was running her private business in Canada for two years. In 2003 she joined to Memolux as Sales Manager of the ACL and NQA products.

Dr. Richard Messnarz

Dr. Richard Messnarz (rmess@iscn.com) is the Executive Director of ISCN LTD. He studied at the University of Technology Graz and he worked as a researcher and lecturer at this University from 1991 - 1996. In 2 European mobility projects (1993 and 1994) he was involved in the foundation of ISCN, and he became the director of ISCN in 1997. He is/has been the technical director of many European projects: PICO - Process Improvement Combined Approach 1995 - 1998, Bestregit - Best Regional Technology Transfer, 1996 - 1999, TEAMWORK - Strategic Eworking Platform Development and Trial, 2001-2002, Media-ISF – e-working of media organisation for strategic collaboration on EU integration, 2001-2002. He is the editor of a book "Better Software Practice for Business Benefit", which has been published by IEEE (www.ieee.org) in 1999 (the leading research publisher in the USA). He is the chairman of the EuroSPI initiative and chair of the programme committee of the EuroSPI conference series. He is also a lead SPICE assessor and has worked as consultant over 12 years for many automotive firms.

The tale of E-Work in European SME’s

Patrizio Di Nicola

Abstract

To maintain or to improve their competitiveness in the knowledge economy, SMEs face the challenge to adopt new working methods, review their work processes and adapt existing skills to new usages based on a growing use of ICT. This is the reason why the e-GAP consortium run five parallel surveys in selected Regions in Finland, France, Italy, Hungary and UK (plus a comparative investigation in Japan), looking for variables that might influence the management of the SMEs to move towards new methods of work like e-Work. The findings highlite that most of the sample companies located in the five European regions of e-Gap project have a very high level of technological growth and a remarkable knowledge of Telework, but a very low percentage of them actually use it for some of their employees. If we admit that technological equipment is no longer an impediment to Telework implementation in European companies, what are the elements that prevent it from spreading?

Keywords

E-Work, Telework, ICT, SME, Cultural Issues

1 Introduction

To maintain or to improve their competitiveness in the knowledge economy, SMEs face the challenge to adopt new working methods, review their work processes and adapt existing skills to new usages based on a growing use of ICT. This is the reason why the e-GAP consortium is running five parallel surveys in selected Regions in Finland, France, Italy, Hungary and UK (plus a comparative investigation in Japan), looking for variables that might influence the management of the SMEs to move towards new methods of work like e-Work. Specifically, e-Work has to be taken as the criterion to measure and understand the reasons that influence the management decision process to accept and adopt new working methods. Understanding and clearly identifying why SMEs hesitate to participate widely in the knowledge economy would enable the EU to recommend a practical process to the management of those firms and help them to become socially responsible in the sense of the Green Paper on a European Framework for Corporate Social Responsibility.

With the present paper we intend to compare the outcomes of surveys, in order to bring out the reasons why and the ways (within specific professional environments) how telework and teleactivities meet difficulties in implementation and provide positive outcomes for cooperation.

2 Objectives

The e-Gap project compares the attitude of small and medium size firms towards Telework in five European countries: France, Italy, Hungary, Finland and United Kingdom.

In each country, we have chosen only one region; we have usually chosen regions with a high level of technological development and with excellent economic performances.

The five regions of e-Gap project are:

1. - Rhône-Alpes in France;
2. - Emilia Romagna in Italy;
3. - Central Transdanubia in Hungary;
4. - Tampere Region in Finland;
5. - Greater West London Wedge in the United Kingdom.

The study carried out by the e-Gap project was both qualitative and quantitative. Nevertheless it did not look for precise quantitative comparisons, but within various perspectives, made experiments in these environments and according to the meaning given by their protagonists. Such a study should highlight some of the best ways for use of new technologies and the overcoming of major barriers or inhibitors to their usage.

In order to explore the two above-mentioned levels, three types of investigation have been used:

1. Data collection and classification about political, legal, social and financial environment of telework and e-work at the context level.

2. A quantitative and representative survey of SMEs in a significant socio-economical area, at the actors level, with a sample of about 300 establishments in each of the participating country, using a questionnaire and computer-aided telephone interviews. Quotas of establishments weighted by business sectors ensured that the results has not been dominated by specific ones (like IT sector).

3. In-depth interviews with various selected people among enterprises, according to a significant sample: workers, managers, executives, and trade unions (actors level). About 60 people per country will undergo semi-directive in-depth interviews. Maybe less outside of official Europe countries.

3 Methology

The survey was undertaken via telephone interviews using a common questionnaire. The first part of the questionnaire referred to the use of information and communication technologies (ICT), the second part inquired about work organization, the third part investigated features of Telework practices and managers' opinions on them, whereas the fourth part investigated organizational background of participating firms to the survey.

The aim of the questions about company's technological equipment were: first, to survey equipment currently available in firms (e-mail, access to the Internet, corporate website, etc.) and second, to collect information to classify companies according to their attitude towards technology. The questionnaire also includes a question about previsions for technological improvements and a subjective evaluation of the level of corporate technology compared to competitors' one (older, similar, better). We also asked interviewees if it is possible to access to company information services from remote stations, and if so, who could benefit from it.

In order to understand management attitude towards technology, we insert two Likert's attitude scales, too: the first one, about disadvantages of ICT technologies (waste of time surfing on the Internet, risk that employees may use information services for their own use, data security-related issues), and the second one, related to problems connected to ICT technologies' introduction in workplaces (generational gap, too little time for training, workers' resistance, etc..).

The second part of the questionnaire concerned work organization, in order to draw a picture of corporate culture within companies. We asked the level of interconnection with other companies, the use of flexible working time, the utilization of project work and the level of autonomy granted to employees.

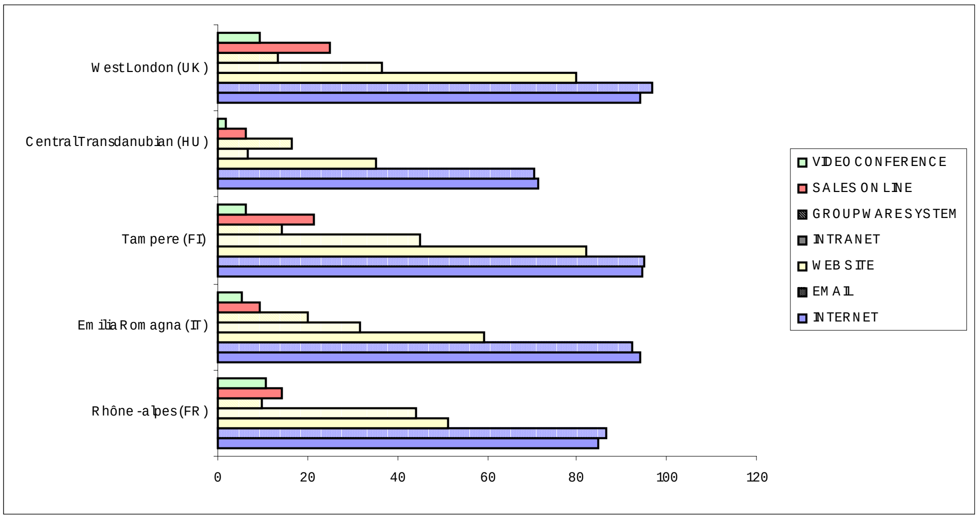
A final part of questionnaire concerned Remote Work and Telework. First of all, we analysed behavioural patterns towards working at home through a Likert's attitude scale. Then, we asked about familiarity with Telework and advantages/disadvantages of this form of work.

4 Usage of ICT in SME's

In recent years, the use of information technology has continuously increased in European companies, especially in regions enjoying rapid economic growth such as those in the e-Gap project. In each country, most of the companies interviewed possessed what we considered to be "common place" technologies: e-mail and the Internet are used in almost every company, especially in Finland and United Kingdom (Graph 1). Even in Hungary, where e-mail and the Internet are less widespread, they are used by almost 75% of the company.

As the English research team states, there is near-universal connectivity. However, we have to remember that some sectors, which are well known to be low users of e-mail and the Internet, such as small retailers and construction firms, are not included in our survey sample.

Graph 1 – Technology equipment



However the other ICT instruments are less frequent and differently widespread in the regions of the survey. The use of the **intranet** is relatively less widespread everywhere. In Hungary it's quite uncommon (below 7%). **Web site** and **sales on line** are less popular also, except for Finland and United Kingdom, where web presence scores a level of penetration close to 80%. **Co-operative work systems** are less widespread everywhere, except for Italy (20%). **Video conference** is not yet used very often: it is rare in Hungary (2%) and relatively low in Italy and Finland. It is used by about 10% of the companies only in France and United Kingdom.

Among preconditions of introducing Telework, general level of firms' ICT use is just as important as its depth. A particularly significant detail concerning the level of technological innovation is the possibility for all, or part of the staff, to gain access to corporate information services. The information is very relevant, because companies that grant corporate access from the outside (checking out e-mails or other corporate information) may correspond to those readily inclined to promote e-work or Telework.

Figures (*Table 1*) allow a certain degree of optimism just for France where for more than half of the companies, **everybody** can use ICT equipment remotely. In the other countries, especially in United Kingdom and Hungary, this possibility is granted to a lower percentage of employees: what is surprising is that nearly 40% of English firms don't allow any remote usage.

Table 1 - Who is permitted to use ICT via remote access?

	Rhône -alpes (FR)	Emilia Romagna (IT)	Tampere (FI)	Central Transdanubian (HU)	West London (UK)
Everybody	**55,4**	27,2	21,6	14,4	15,8
Only senior management	24,1	11,6	23,0	24,2	**31,7**
Even middle management	10,5	9,2	22,0	8,8	11,2
Even supervisors	7,5	16	0,0	1,6	1,0
Nobody	10,9	33,6	31,1	32,0	**37,6**
N	294	500	305	248*	303

* The percentage of *not answer* is very high (19, 0%)

5 Attitudes to remote work

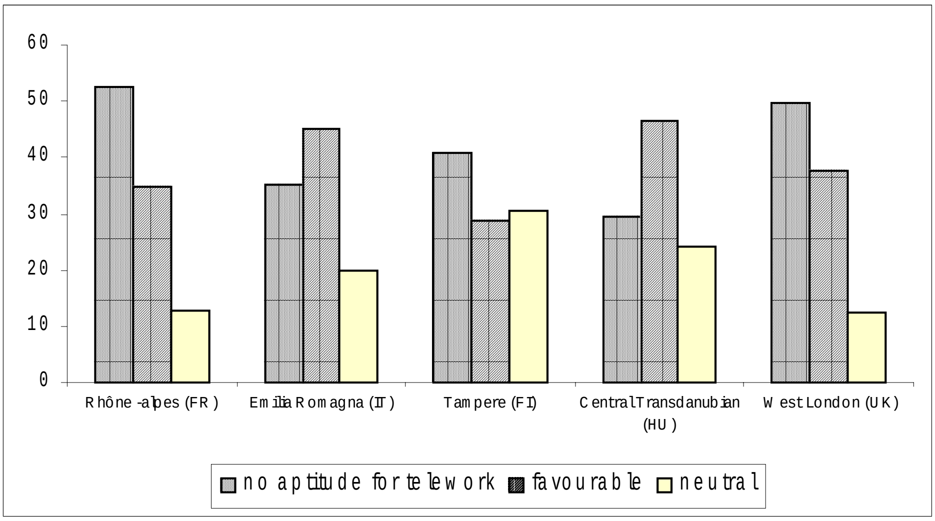
To measure the attitudes towards work at distance, we used a Likert's scale, through we recorded the agreement/disagreement rate on a set of statements (Table 2). The main agreement among the managers concerns the statement according to which: **working at home is only suitable for trustworthy and close employees of the company**. This is true almost everywhere, but in Hungary. In United Kingdom, more than 90% of managers interviewed agree with this assertion. It is important to remember that the people questioned are top managers or assistants. The statement Hungarian managers agree most with is the one claiming that **people should work less at home because there is no managerial supervision**. What is surprising is that a wide percentage of English managers also agree with this statement. Another important assertion, in order to understand the positive aptitude towards work at distance, is the one claiming that **home worker's efficiency is higher because they can organise themselves autonomously**. The percentage of agreement to this assertion is close to 45% in France, Italy and United Kingdom; it's a little bit smaller in Finland and especially in Hungary.

Table 2 - Attitude Towards Remote Working (percentage of agreement)

	Rhône -alpes (FR)	Emilia Romagna (IT)	Tampere (FI)	Central Transdanubian (HU)	West London (UK)
- People work less from home because they don't feel controlled	16,3	23,6	17,0	**47,4**	**40,3**
- Working at home, is very difficult to match work and leisure time	52,7	35,0	40,7	29,4	49,8
- Work at home is suitable just for trustworthy and close collaborators	61,3	58,2	41,3	23,9	**90,8**
- You can work better at home because office is a source of distractions	20,0	13,8	28,9	31,0	30,0
- Home workers' efficiency is higher because they can organise themselves autonomously	44,6	44,2	31,1	**26,8**	45,2

We have aggregated these figures in a synthetic index of **inclination towards working at home**. According to this index, Hungarian firms seem to be more favourable to work at home, while more than half of French and English entrepreneurs shows some hesitation toward it (Graph 2). These findings are surprising, but we have to note that Likert's scale in telephone interviews may lessen the effects of social desirability of statements. This could be especially true when statements are less familiar, as in Hungarian experience.

Graph 2 – Inclination Towards Working at Home



To be able to launch e-work the company has to adopt new ways of operation, that is, reorganize its work. Change and reorganization of work often pose problems in connection with distributed business and ework arrangements. Organizing things in an old established manner often feels easier than making rearrangements.

6 Aptitude and interest in teleworking

Despite the fact that most of the sample companies located in the five European regions of e-Gap project have a very high level of technological growth and a remarkable knowledge of Telework, a very low percentage of them actually use it for some of their employees. If we admit that technological equipment is no longer an impediment to Telework implementation in European companies, what are the elements that prevent it from spreading?

Actually, not all high technological companies - as technological determinism would have it - rank among those who show a propensity for Telework (Table 3). Italian figures show that the number of firms not inclined to Telework is even smaller in companies with a low level of technology. We even notice that companies in favour are *not always* more easily found in high technology firms.

Table 3 - Propensity For Remote Working according to the Level of Technological Equipment.

Rhône -alpes (FR)	low	medium	high	Total
No aptitude for Telework	93,3	54,7	-	35,5
Favourable towards Telework	-	-	54,0	24,7
Neutral towards Telework	-	30,8	32,8	27,1
Emilia Romagna (IT)	low	medium	high	Total
No aptitude for Telework	**19,4**	36,6	35,8	35,0
Favourable towards Telework	44,4	41,5	**49,1**	45,0
Neutral towards Telework	36,1	22,0	15,1	20,0
Tampere (FI)	low	medium	high	Total
No aptitude for Telework	43,8	37,5	**42,6**	40,7
Favourable towards Telework	25,0	30,8	27,8	28,9
Neutral towards Telework	31,3	31,7	29,6	30,5
Central Transdanubian (HU)	low	medium	high	Total
No aptitude for Telework	30,1	32,6	22,2	29,4
Favourable towards Telework	50,5	41,1	**51,4**	46,4

Neutral towards Telework	19,4	26,2	26,4	24,2
West London (UK)	low	medium	high	Total
no aptitude for Telework	50,0	50,0	50,6	50,3
Favourable towards Telework	37,5	37,5	38,9	38,3
Neutral towards Telework	12,5	12,5	10,5	11,4

So, there are companies who have a high level of technology but they are not inclined to use Telework, just as there are others that have a low level of technology but are more eager to test it. Therefore, the relation between these two variables is not linear. Nevertheless, if we put together the figures obtained thought the two indexes (*technological level* and *inclination to work at home*), we obtain a model of **propensity to Telework** (table 4)[1]. We obtained this typology aggregating some of the cells created by crossing the two indexes, as you can see in Table 10. The first type "**no interest** to Telework" are those who have no technological equipment for Telework (low or medium) and little or no inclination towards it. In this case we think that lack of technology actually compromises any telework experimentation. While **Resistant** are those who are not in favour in spite of having technological facilities.

The *most significant category is the* **Interested** one, composed by those who are in favour of Telework and have a medium-to-high technological equipment, while those who expressed no opinion regarding work from home can be regarded as **Indifferent,** in spite of their technological equipment.

Table 4 - Model of propensity to Telework

Favourable	**Low**	**medium**	**High**
no aptitude for Telework	**No interest (28,1%)**		**Resistant (19,1%)**
inclined/favourable		**Interested (32,6%)**	
neutral	**Indifferent (20,1%)**		

According to this classification, Italian firms are more interested in Telework (above 40%), followed by the English (35%) and the Hungarian ones (above 30%). Italy has a low percentage of firms indifferent to Telework (15%). It is quite surprising that Hungary has the lowest percentage of firms indifferent to Telework (only 5%) but we could explain this data if we remember that the number of firms with a high technological level in Hungary is very low, compared to the other countries. However the percentage of Hungarian companies not interested in Telework – those with no technological equipment and little or no inclination to Telework – is much higher than in the other countries (close to 40%). On the other hand, it is quite surprising that the number of this kind of firms – not interested to Telework – is very high in France too (above 35%). It is also interesting to note that Finland has a high percentage of firms classified as indifferent (above 30%).

There is a strong relation between inclination on Telework and the **chance of employees to organize their own working time**, too. Almost everywhere, companies that have no interest in Telework admit more frequently that nobody can organise his own working time. However, firms that are

[1] In this Table, Italian data were proportioned to 300 units not to affect the general typology composition, when it is considered in its total distribution. In the other data processing we didn't make this operation because we always worked on percentage data and because data were processed on single databases. The reduction to 300 cases, referred to ***propensity to telework*** typology, produced variations under 1%.

Favourable to Telework grant this chance more frequently to all employees – or at least to middle managers. Only in Finland does this relation not exist.

7 Conclusion

The survey carried out in these five regions let us set up some general conclusions, in spite of some differences in the results, as outlined in previous paragraphs. Basic technology level is very high in all countries, especially if we think about Internet and e-mail diffusion. Anyway, indispensable technologies for e-Work development seem to be still not very widespread, as in the case of web sites and collaborative tools. It is also very important that entrepreneurs, as we shown in national reports, state that the main problems, introducing new technologies, are the lack of time for training, the resistance to changes and the generational gap. These answers indicate a low level of propensity towards innovation, in spite of high levels of technological equipment.

From an organizational point of view, inclination towards innovation seems low: project work is not widespread yet; management seems still very hierarchical and the only widespread form of flexibility is related to new kinds of contract. Moreover, the most common management model is direct supervision and visual control of work: that is why working site-off is not yet a common practice and it is used mainly by managers and sales force.

Although the majority of interviewees – especially in service sector - know the meaning of the term “telework”, among entrepreneurs prevails a stereotyped vision of this kind of work, according to which employees are the main beneficiaries and economic benefits, well known in the literature, are considered almost insignificant.

We must note that only a third of interviewees (especially employees in economic sectors close to e-work) could be classified as interested in telework, while most of those interviewed were not interested, indifferent or resistant towards telework.

The outcomes of our survey are quite clear: technology is not sufficient for telework diffusion, and the reason for this low diffusion must be researched in the effect of organizational cultures more than in technical problems. In fact, the inclination towards telework doesn’t change in significant ways together with changes in technological equipment. This centrality of the cultural issues, raise, in our opinion, the need for future EU founded projects to focus even more on the approach of people to ICT, dedicating every effort in investigating the cultural aspects of new technologies.

8 Author CVs

Patrizio Di Nicola

sociologist, is Managing Director of Futuribile srl, a research firm based in Rome. He is Professor of Sociology of Organisation at the Faculty of Communication Sciences - University of Rome "La Sapienza" and of Sociology of Economic Processes and of Work at the Faculty of Architecture “Ludovico Quadroni”. At the Link Campus - University of Malta in Rome is Director of the MA in E-Business and Professor of Sociology of Organisation and e-Work. He was scientific director for many researches on telework, virtual enterprises, distant learning and the post -industrial labour market, and was managing consultant of enterprises and Government bodies, including the Ministry of Regional Affairs, Telecom Italia, Isfol, the International Training Centre of the ILO, the European Commission. From 1996 is National Co-ordinator for the *European Telework Development Initiative* and is involved since 2001 in the international projects BISER, E-GAP, ITENETS. Is author or contributor of many books and is editor of the Italian version of the TCA' *Teleworking Handbook* (1997 and 1999). Most recent publications include: *Telelavoro domiciliare*, Ediesse, 2002, *Teleworking best practice handbook* (2001); "Telecom Italia Telephone Directory Service", in *Work is Changing the Future is Now*, Proceedings of the Seventh European Assembly on Telework & New Ways of Working - *Telework 2000; Telework between Law and Contract* (1998)¸ *Telework: Working On progress, 1998* (CD Rom); *New International Perspectives on Telework*, 1998; *Internet 2000*, (1997). Actually is writing two new books: *Sociology of Telework* and *Learning in the Net Society.* Since 1995 he created and maintain the most famous Italian Telework Website (www.telelavoro.rassegna.it). Many of his articles can be found at www.dinicola.it.

Skills Assessment Based Human Resource Management

Richard Messnarz (ISCN), Eva Feuer (SZTAKI), Damjan Ekert, Natalia Costas Lago (ISCN), Eugene O'Leary, Brian Foley, Nuala O'Shea (TecNet), Ignacio Sanchez, Gonzalo Velasco (FGUVA), Bruno Wöran (Danube)

Abstract

CREDIT [2],[3],[5] was an EU Multimedia project running from 1998 – 2001. It analysed skills assessment processes applied in Europe and also developed a generic platform which can be configured for different skill sets (according to DTI UK skill card standards), supports skills assessment, evidence collection, and advise through an Internet environment. The project results has further been developed into a commercial environment called Capability Adviser System.

e-Commerce Jobs is an EU Leonardo da Vinci project (2002 – 2003) [4] which develops skills cards for professions like e-commerce engineer, e-business manager, and e-strategy manager and uses the Capability Adviser platform to offer online access to the skills cards, online seölf assessments, online formal advise , and online learning path generation.

In this paper we outline how important such new approaches for skills management are concerning human resource management. Human resource services are an important business case addressed by the e$^{\textbf{\textit{scm}}}$ [1] (e-Services Capability Model), and human resource management in general is an important factor in SPICE for reaching a level 3.

Keywords

Skills, Skills Assessment, Human Resource Improvement Plan, Learning Paths

1 Human Resource Management

The e^{scm} (e-Services Capability Model[1]) [1],[7],[8],[9] provides an improvement framework for IT-enabled outsourcing services, such as outsourced IT-intensive business processes, projects and tasks that use Information Technology as an enabler for designing services, coordinating service deployment, and delivering services.

One of the important IT-enabled outsourcing service segments comprises **Human Resource (HR) services** (including back office operations and strategic HR function outsourcing).

For such a service the e^{scm} describes a set of key processes:

- Organisational Management
- **The People element** provides guidance on personnel acquisition, training and development, which is critical to success since personnel play a key role in the design and delivery of IT-EOS.
- Business Operations
- Technology element
- Knowledge Management

For such these key outsourcing relevant processes the e^{scm} describes a set of capability levels:

- Level 1 - Initial;
- Level 2 - Performing to meet client requirements;
- Level 3 - Controlling through measurement;
- Level 4 - Enhancing through innovation; and,
- Level 5 - Sustaining excellence.

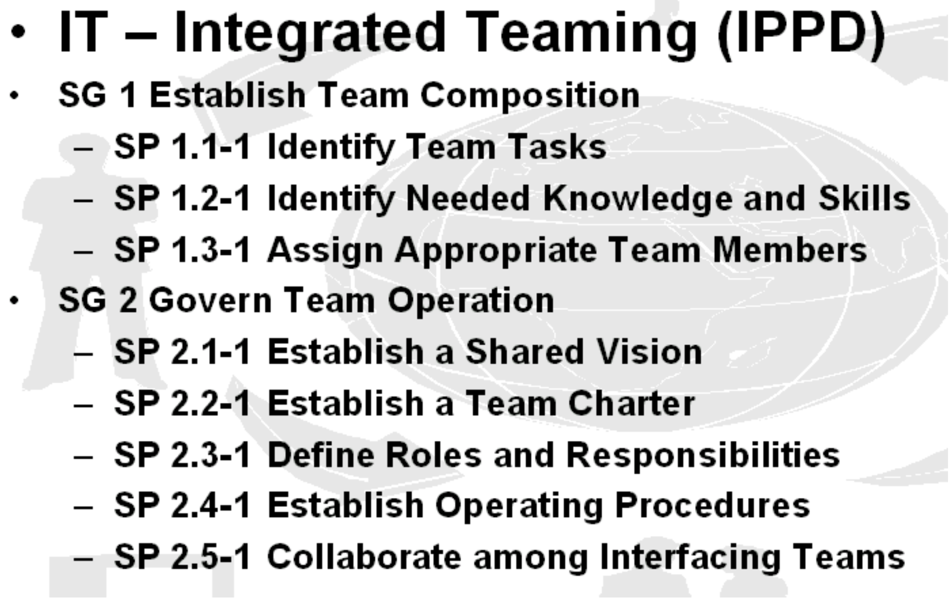
The Capability Adviser System supports an e-service concept with a defined model for

- Defining job roles with required qualifications and offering skills assessments against job roles online
- Gathering evidences to prove competencies and to receive online formal guidance and mentoring.
- A database of skills profiles (against different job roles) so that the personnel selection and acquisition of personnel for projects can be based on available skills profiles of persons (measured against required skills for job roles).
- A system that bases on an existing skills profiles generating a recommended learning path for a person, to enable a computer supported training plan generation.

Because the system is online the human resource selection, mentoring, and learning plan establishment can be traced and used even in largely distributed situations (people outsourced to different places in the world).

[1] e^{scm} is a trademark of the Carnegie Mellon University, Pittsburgh, USA

In **CMMI** (Capability Maturity Model Integrated[2]) the two processes **Integrated Project Management** and **Integrated Teaming** deal with job roles, responsibilities, and their collaboration in teams. Dependent on the responsibilities of job roles they will have certain skills and training requirements.

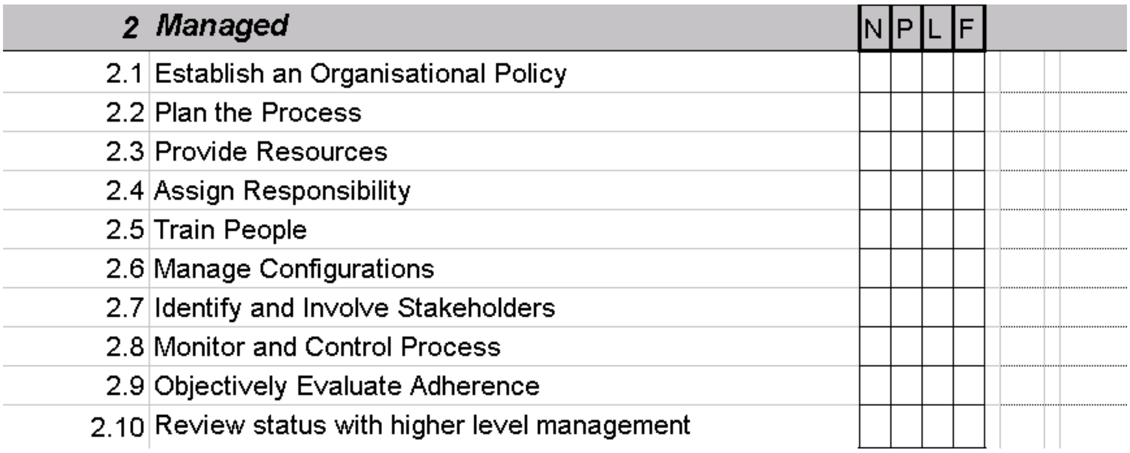


- IT – Integrated Teaming (IPPD)
- SG 1 Establish Team Composition
  - SP 1.1-1 Identify Team Tasks
  - SP 1.2-1 Identify Needed Knowledge and Skills
  - SP 1.3-1 Assign Appropriate Team Members
- SG 2 Govern Team Operation
  - SP 2.1-1 Establish a Shared Vision
  - SP 2.2-1 Establish a Team Charter
  - SP 2.3-1 Define Roles and Responsibilities
  - SP 2.4-1 Establish Operating Procedures
  - SP 2.5-1 Collaborate among Interfacing Teams

Figure 1: Specific Goals and Specific Practices Related with Job Roles in CMMI

These processes are required for achieving a CMMI staged level 3. However, the level 2 generic practices (applied on all processes to evaluate a capability level 2) already include human resource related criteria such as:

- Provide Resources
- Assign Responsibility
- Train People



2	Managed	N	P	L	F
2.1	Establish an Organisational Policy				
2.2	Plan the Process				
2.3	Provide Resources				
2.4	Assign Responsibility				
2.5	Train People				
2.6	Manage Configurations				
2.7	Identify and Involve Stakeholders				
2.8	Monitor and Control Process				
2.9	Objectively Evaluate Adherence				
2.10	Review status with higher level management				

Figure 2: Generic Practices Related with Job Roles and Skills in CMMI

The Capability Adviser System [14] supports a structured and defined process to

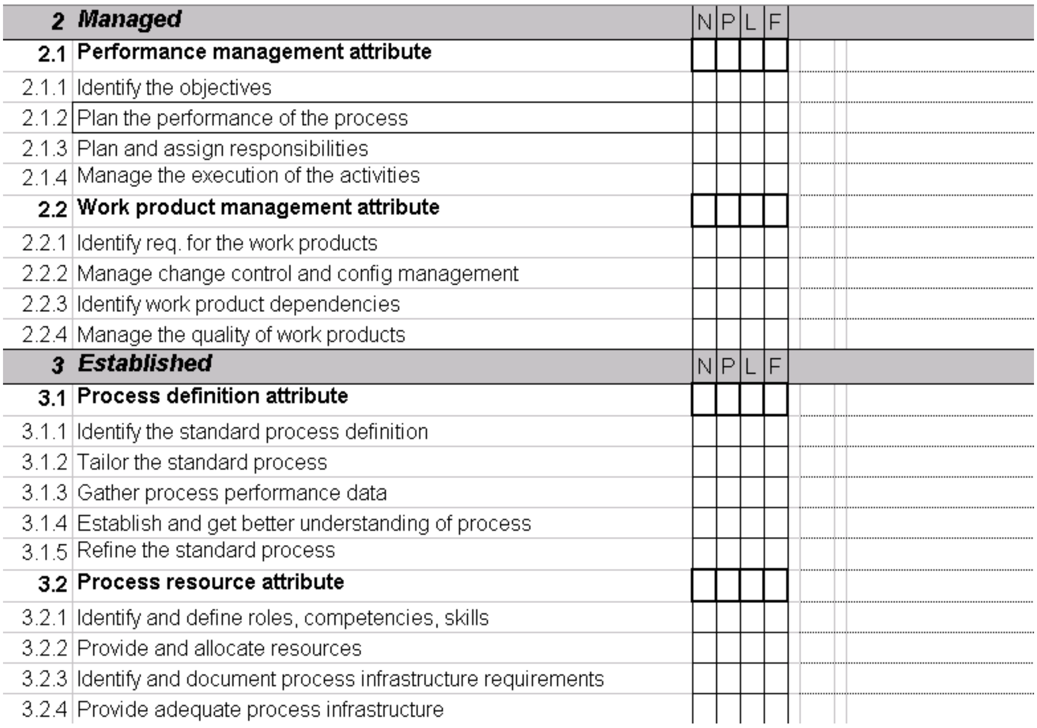
- Defining skills sets of job roles (assign responsibilities)
- Allow skills assessment – to identify skills gaps and establish training plans based on standard procedures supported through a system (train people)

[2] CMMI is a trademark of the SEI, Carnegie Mellon University, Pittsburgh, USA

- Allow managers to select best staff for the project based on their available skills profiles

Because it is a system based approach the human resource decisions follow a structured, re-usable, and traceable process.

In **SPICE** (ISO TR 15504 , Software Process Improvement and Capability determination) the human resource management is part of the process attributes 2.1 and 3.2 on the capability levels 2 and 3. So you cannot achieve a capability level 3 on any process without properly managing the human resource management issue.



2	***Managed***	N	P	L	F
2.1	**Performance management attribute**				
2.1.1	Identify the objectives				
2.1.2	Plan the performance of the process				
2.1.3	Plan and assign responsibilities				
2.1.4	Manage the execution of the activities				
2.2	**Work product management attribute**				
2.2.1	Identify req. for the work products				
2.2.2	Manage change control and config management				
2.2.3	Identify work product dependencies				
2.2.4	Manage the quality of work products				
3	***Established***	N	P	L	F
3.1	**Process definition attribute**				
3.1.1	Identify the standard process definition				
3.1.2	Tailor the standard process				
3.1.3	Gather process performance data				
3.1.4	Establish and get better understanding of process				
3.1.5	Refine the standard process				
3.2	**Process resource attribute**				
3.2.1	Identify and define roles, competencies, skills				
3.2.2	Provide and allocate resources				
3.2.3	Identify and document process infrastructure requirements				
3.2.4	Provide adequate process infrastructure				

Figure 3: Process Attributes and Management Practices on Levels 2 and 3 in SPICE

2 Skills Assessment and Learning Process

2.1 The Underlying Skills Definition Strategy

Processes are managed by teams, and team members take certain job roles with responsibilities. These job roles require certain skills which are being structured in form of a required skills set [10],[11],[12],[13],[14].

For structuring a skills set we followed the EU standards for skills cards.

A **Domain**, contains

Job Roles, which contain

Units, which contain

Elements, which contain

Performance Criteria, which must be proven by

Evidences.

Figure 4: Basic elements of the skills definition model

Domain: An occupational category, e.g. childcare, first level management or software engineering.

Job Role: A certain profession that covers part of the domain knowledge. E.g. domain = automotive, job role = automotive SW project leader

Unit (UK standards): A list of certain activities that have to be carried out in the workplace. It is the top-level skill in the UK qualification standard hierarchy and each unit consists of a number of elements.

Element (UK standards): Description of one distinct aspect of the work performed by a worker, either a specific task that the worker has to do or a specific way of working. Each element consists of a number of performance criteria.

Performance criterion (UK standards): Description of the minimum level of performance a participant must demonstrate in order to be assessed as competent. A performance criterion may have different relevant contexts.

Evidence: Proof of competence.

2.2 The Underlying Skills Assessment Model

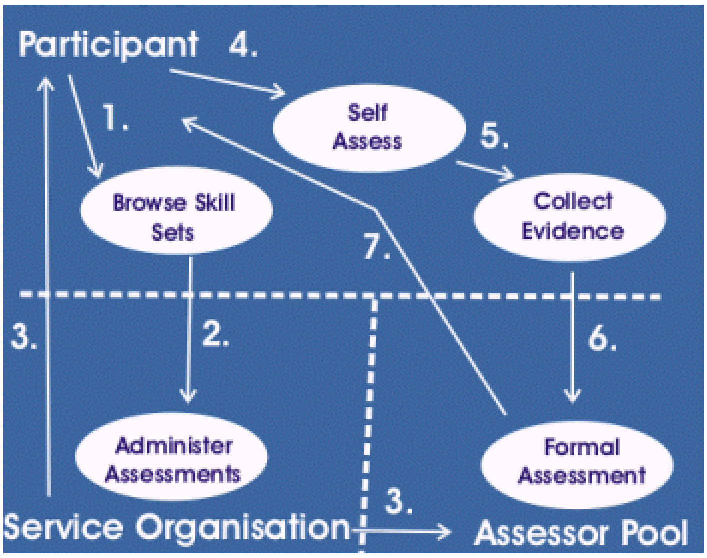


Figure 5: The Skills Assessment Process Model

Step 1 – Browse a Skills Set : You select a set of skills or competencies, which are required by your profession or job using national standards or your company standards. You browse different skills cards and select a job role you would like to achieve.

Step 2 – Register Interest with a Service Unit : This can be a service unit inside your own company (e.g. a personnel development department) or a skills card and assessment provider outside your company which offers skills assessment services.

Step 3 – Receive an Account for Self –Assessment and Evidence Collection : The service unit will provide you with an account and working space in which you can go through the steps of online self assessment and the collection of evidences to prove that you are capable of certain performance criteria.

Step 4 – Perform Self Assessment : You log into the system , browse through the skills required and self assess performance criteria, whole elements or whole units with a standard evaluation scale of non-applicable, not adequate, partially adequate, largely adequate, and fully adequate. A skills gaps profile can be generated and printed illustrating in which areas your self assessment shows improvement potentials.

Step 5 – Collect Evidences : Before you want to enter any formal assessment you need to prove your skills by evidences. Evidences can be any electronic files (sample documents, sample graphics, results of some analysis, etc.) or any references with details (e.g. a certificate received from a certain institution). Evidences you can then link to specific performance criteria or whole elements of skills units.

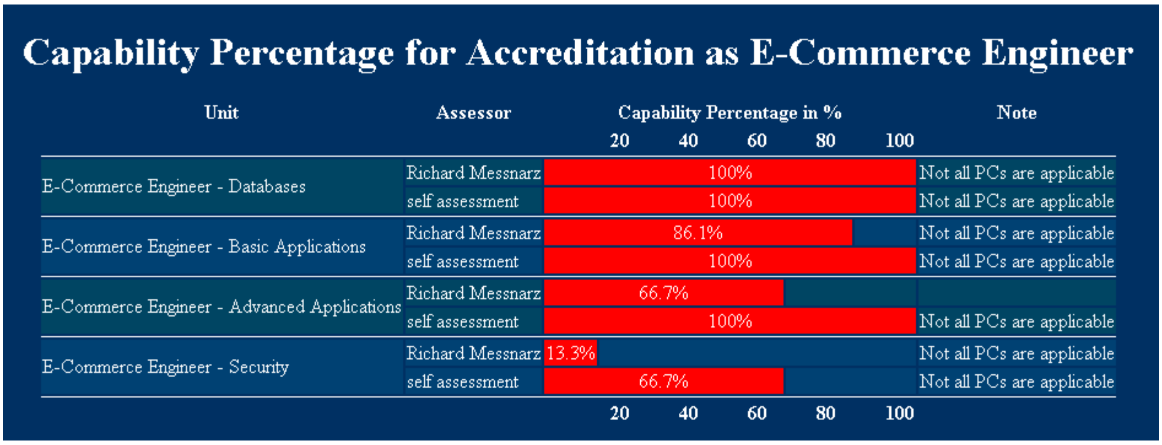
Testing of Skills (Addition to Step 5) – In traditional learning schemes people have always needed to go to a learning institution (university, accreditation body, pro-fessional body, etc.) to take exams and they received a certificate if they pass. This traditional approach however is insufficient when it comes to measuring experience and (soft) skills learned on the job and fails to give recognition to skills gathered on the job. The APL (Accreditation of Prior Learning) approach, by contrast, collects so called evidences. Evidences can be certificates obtained in the traditional way, but also references from previous employers, materials from previous projects in which the person took ownership of results (e.g. a test plan) to prove their capability, as well as any kind of proof of competence gathered on the job. The assessors will then evaluate the evidences provided and not only rely on certificates and exams.

Step 6 – Receive Formal Assessment : Formal assessors are assigned by the service unit to the skills assessment. Once formal assessors log into the system they automatically see all assigned assessments. They select the corresponding one and can see the uploaded evidences. They then formally assess the evidences and assess the formal fulfilment of performance criteria, whole elements or whole units with a standard evaluation scale of non-applicable, not adequate, partially adequate, largely adequate, and fully adequate. In case of missing competencies they enter improvement recommendations, a well as learning options.

Step 7 – Receive Advise on Learning / Improvement Options: After the formal assessment the participants log into the system and can see the formal assessment results from the assessors, can print skills gaps profiles based on the assessor results, and can receive and print the improvement recommendations and learning options. If required, the generation of learning options can also be automated through the system (independent from assessor advises).

2.3 Human Resource Selection Process

Managers have access to the skills database and can access skills profiles of staff. Staff members are assessed against required skills and skills gaps lead to learning recommendations. Selection of staff members would then depend on the job roles they need to satisfy and how they assess against the required skills of that specific job roles.



Capability Percentage for Accreditation as E-Commerce Engineer

Unit	Assessor	Capability Percentage in %	Note
E-Commerce Engineer - Databases	Richard Messnarz	100%	Not all PCs are applicable
	self assessment	100%	Not all PCs are applicable
E-Commerce Engineer - Basic Applications	Richard Messnarz	86.1%	Not all PCs are applicable
	self assessment	100%	Not all PCs are applicable
E-Commerce Engineer - Advanced Applications	Richard Messnarz	66.7%	
	self assessment	100%	Not all PCs are applicable
E-Commerce Engineer - Security	Richard Messnarz	13.3%	Not all PCs are applicable
	self assessment	66.7%	Not all PCs are applicable

Figure 6: A Typical Generated Skills Profile

3 The e-Commerce Jobs Practical Case Study

3.1 Motivation / Demand

In countries like Hungary, Austria, Ireland, and Spain national innovation centers identified a huge demand for e-business skills. In collaboration with chambers of commerce many hundred firms had been identified who would need e-business skills to prepare their company for the future required developments.

In late 2001 the e-Commerce Jobs Project was accepted which (Jan. 2002 – March 2004)

- Developed skills cards for e-commerce engineers, e-business managers, and e-strategy managers
- Fed the skills cards into the Capability Adviser System
- Provided skills profile and skills gaps analysis (including formal mentoring)
- Developed courses and exercises the gain the required skills

With this approach the following human resource management related scenarios can be implemented (typical implementation cases):

In collaboration with national training bodies the Capability Adviser Portal is made accessible and self assessments lead to learning paths, and the attendance of courses (recommended in the learning path) at the national training providers.

At large companies (training departments) the Capability Adviser Portal is made freely available (only during project period, later commercial) and self assessments lead to learning paths, and the potential to attend e-business related courses by an external training provider.

Skills placement and recruitment companies can use the Capability Adviser Portal to make decisions on recruitment (who has the best profile to match a certain job role).

Etc.

3.2 The Skills Sets

Define the Job Roles and Required Skills

When you start using the standard skills assessment and learning advise system, your first step (as published also in EuroSPI 2000 by Ericsson Italy) is to define the job roles which you can play within the team. For these job roles you need to identify the required skills.

In case of the e-commerce business manager for instance, the job role consisted of the following skills units, and learning elements illustrated in Figure 7.

Define Multiple Choice Questionnaire – For on line self test

For each learning element the model requires performance criteria which are used to check the capability of a person to perform certain tasks. For each performance criterion a set of multiple choice test questions can be configured.

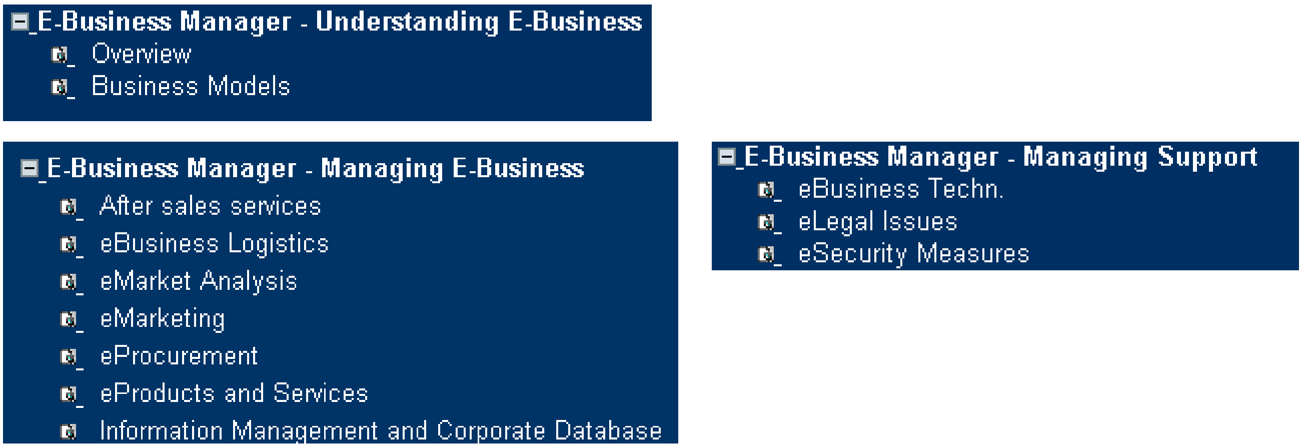
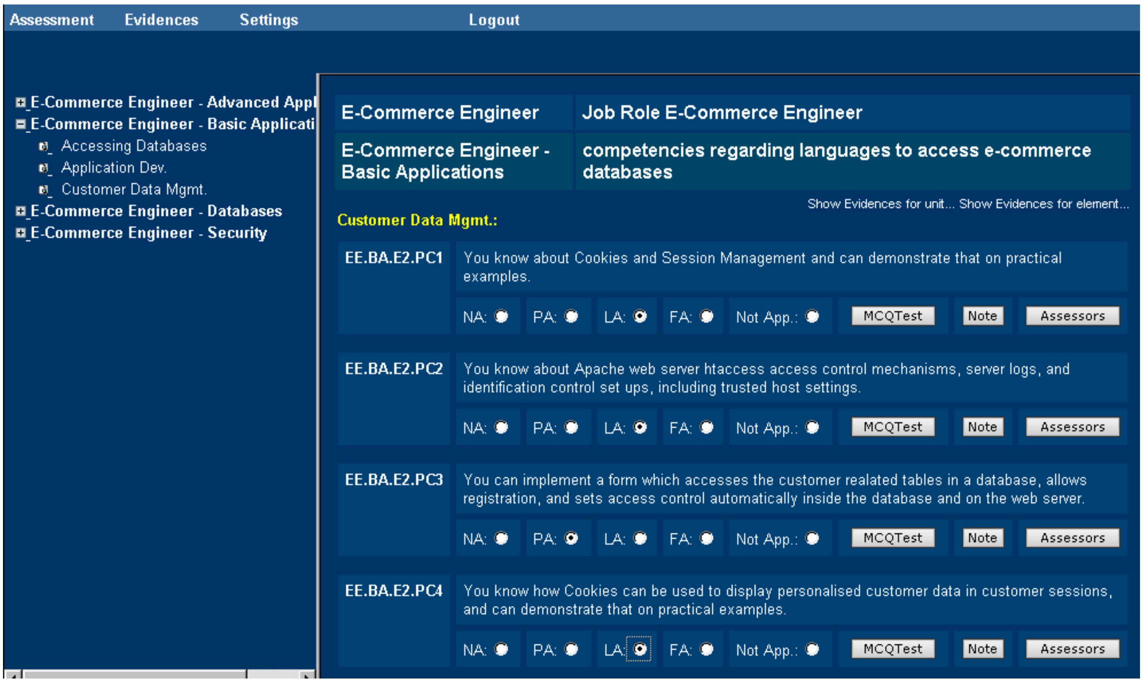


Figure 7: Skills Units and Learning Elements of an E-Business Manager [4]


Assessment
Evidences
Settings
Logout
E-Commerce Engineer - Advanced Appl
E-Commerce Engineer - Basic Applicati
Accessing Databases
Application Dev.
Customer Data Mgmt.
E-Commerce Engineer - Databases
E-Commerce Engineer - Security
Show Evidences for unit... Show Evidences for element...

E-Commerce Engineer	Job Role E-Commerce Engineer
E-Commerce Engineer - Basic Applications	competencies regarding languages to access e-commerce databases

Customer Data Mgmt.:

EE.BA.E2.PC1 You know about Cookies and Session Management and can demonstrate that on practical examples.
NA: ○ PA: ○ LA: ● FA: ○ Not App.: ○ MCQTest Note Assessors

EE.BA.E2.PC2 You know about Apache web server htaccess access control mechanisms, server logs, and identification control set ups, including trusted host settings.
NA: ○ PA: ○ LA: ● FA: ○ Not App.: ○ MCQTest Note Assessors

EE.BA.E2.PC3 You can implement a form which accesses the customer realated tables in a database, allows registration, and sets access control automatically inside the database and on the web server.
NA: ○ PA: ● LA: ○ FA: ○ Not App.: ○ MCQTest Note Assessors

EE.BA.E2.PC4 You know how Cookies can be used to display personalised customer data in customer sessions, and can demonstrate that on practical examples.
NA: ○ PA: ○ LA: ● FA: ○ Not App.: ○ MCQTest Note Assessors

Figure 8: Self Assessment and (MCQT Button) Multiple Choice Test for the E-Commerce Engineer Skills

3.3 The Evidence Collection

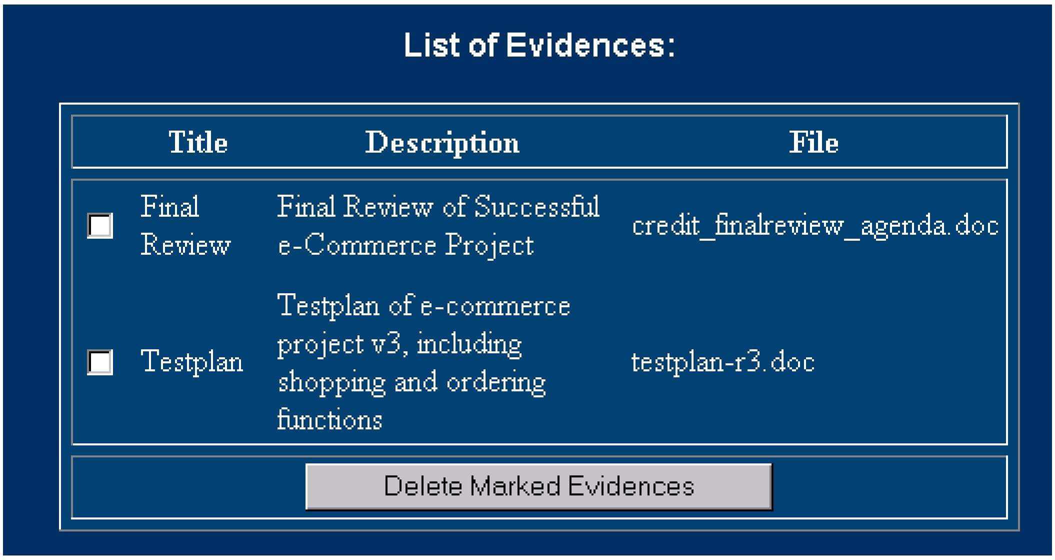
Provide Self Assessment Working Areas for Personalised Accounts

In a large company employees following the system supported skills assessment and human resource selection procedures would receive their personal assessment and learning account. A formal assessor pool (mentors) would be set up who log into the system and see all assessments assigned to them and give formal assessment and advise.

Collect an Evidence Portfolio to Prove Competencies

Employees then log into their user areas and do self assessments and collect an evidence portfolio.

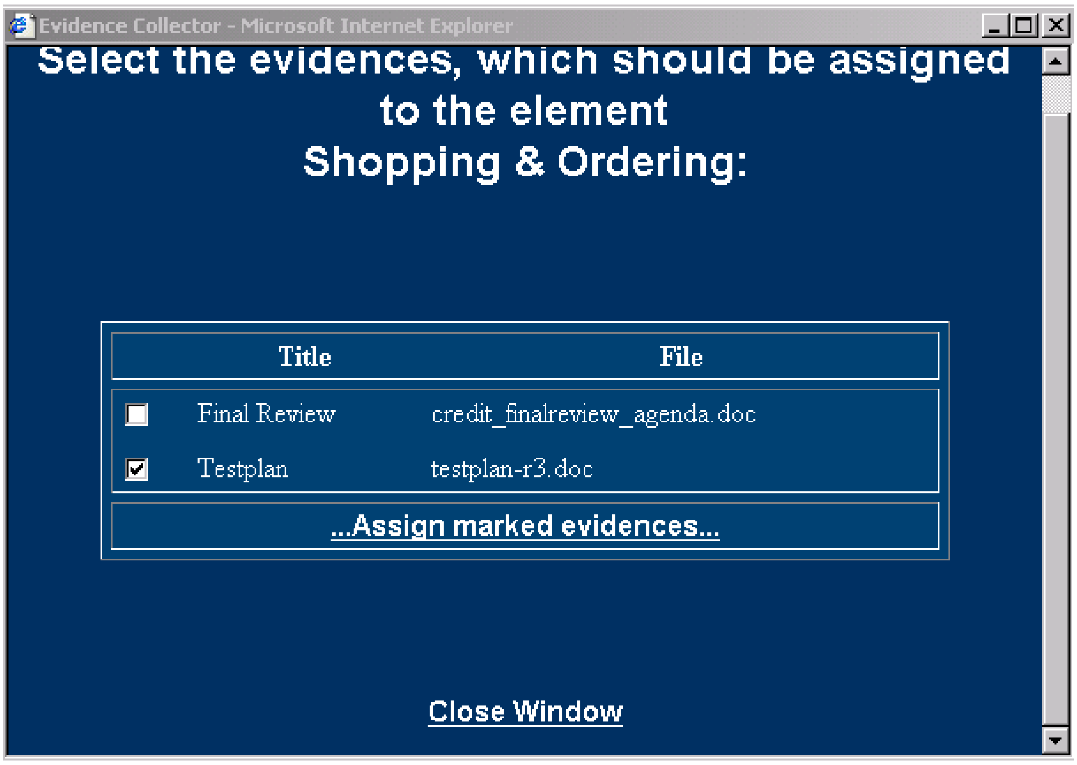
An evidence is a (experience on the job , or training certificate, or job reference, etc.) proof that a certain learning element is covered.



List of Evidences:

	Title	Description	File
□	Final Review	Final Review of Successful e-Commerce Project	credit_finalreview_agenda.doc
□	Testplan	Testplan of e-commerce project v3, including shopping and ordering functions	testplan-r3.doc

Figure 9: Collection of Evidence Portfolio in Personalised User Area



Evidence Collector - Microsoft Internet Explorer

Select the evidences, which should be assigned to the element Shopping & Ordering:

	Title	File
□	Final Review	credit_finalreview_agenda.doc
☑	Testplan	testplan-r3.doc

...Assign marked evidences...
Close Window

Figure 10: Evidence Assignment to Prove Elements of Required Skill Cards

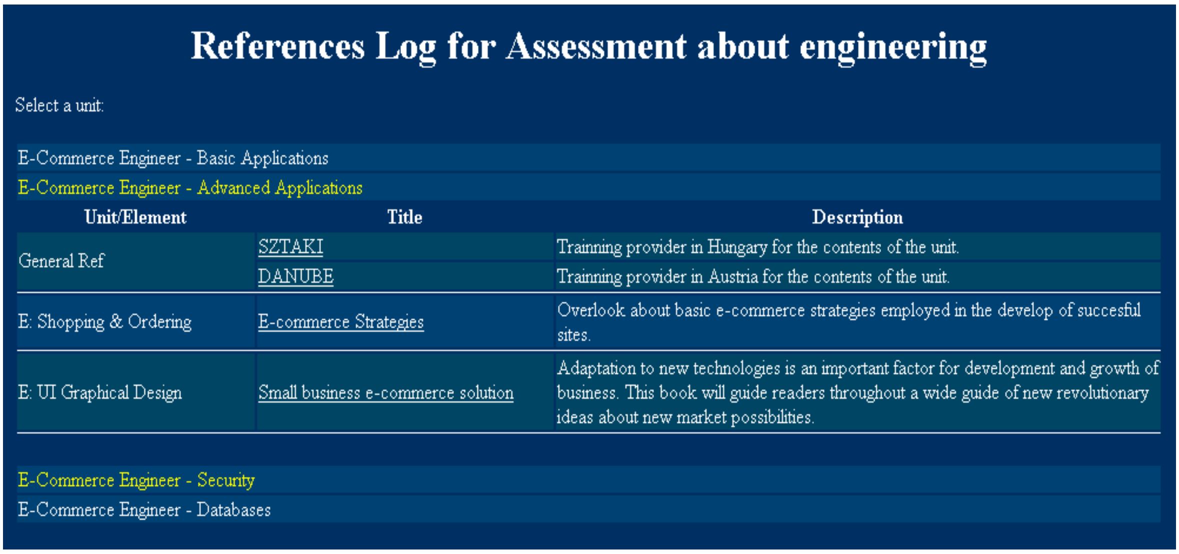
3.4 The Skills Profile and Learning Path

Formal Advise - Mentoring

After self assessments and evidence collection by the employees the mentors log into the system, look at the evidences, formally assess the skills, and provide learning recommendations. These are then online visible for the employee.

Learning Path Generation

The system also stores learning links and recommendations per element and automatically generates a learning plan based on which elements/units are not sufficiently satisfied by the employee.



References Log for Assessment about engineering

Select a unit:

E-Commerce Engineer - Basic Applications

E-Commerce Engineer - Advanced Applications

Unit/Element	Title	Description
General Ref	SZTAKI	Trainning provider in Hungary for the contents of the unit.
	DANUBE	Trainning provider in Austria for the contents of the unit.
E: Shopping & Ordering	E-commerce Strategies	Overlook about basic e-commerce strategies employed in the develop of succesful sites.
E: UI Graphical Design	Small business e-commerce solution	Adaptation to new technologies is an important factor for development and growth of business. This book will guide readers throughout a wide guide of new revolutionary ideas about new market possibilities.

E-Commerce Engineer - Security

E-Commerce Engineer - Databases

Figure 11: Learning References (Displayed in Yellow – Areas where you need to learn)

Objective Human Resource Selection

Decisions on job roles will then be available through a system (with online access by personnel managers and mentors and access y employees – respecting privacy of areas protected by passwords) and personnel decisions are made based on objective skills cards and skills gaps profiles.

See also Figure 6.

3.5 The e-Based Human Resource Management Process

The human resource management process is therefore a system based approach enabling

- Job role definition (based on standards)
- Skills assessment and skills gaps analysis
- Evidence collection and prove of competencies by employees
- Formal assessment and mentoring online
- Human resource selection based on skills profiles
- Human resource development based on generated learning paths

Especially in a situation where companies need to go global to stay competitive and need to merge with organisations at different locations this might be a future concept for managing human resources.

So far the participating partners (in e-Commerce Jobs) have applied process based assessment methods/models and established process based improvement plans. In future this process thinking will be enriched by a rigorous human resource based improvement planning.

4 Outlook

In e-Commerce Jobs [4] we have developed a set of skills cards (and courses) related with e-skills from the engineering level up to the management level. The system [14] is being tested for these job roles in national training organisations, and corporate networks, who need too upgrade their employees skills to match with the future demand in e-skills. The process of learning and human resource development will be supported by the Capability Adviser system.

Organisations interested in joining can contact the prime at eva.feuer@sztaki.hu.

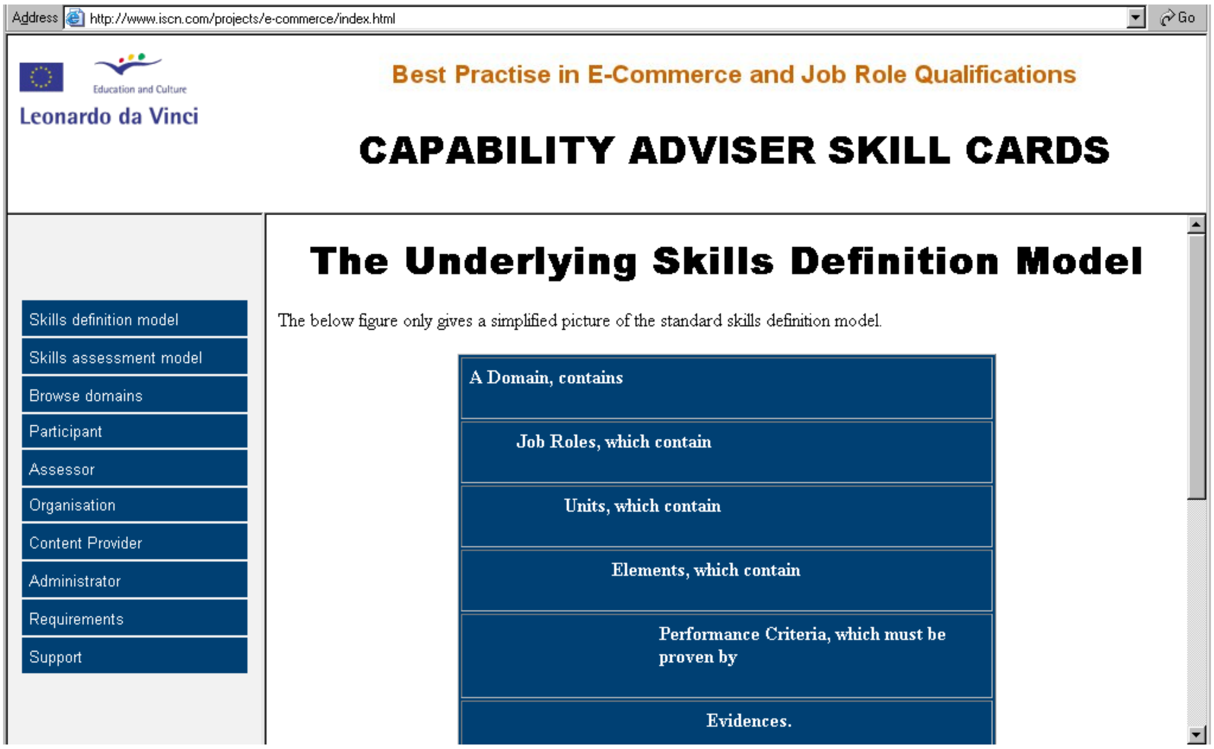
In general we presented a new way of virtual human resource management in this paper which shall be applied in more areas than just e-commerce skills in the future.

More information on the e-Commerce Jobs project can be found at:

http://www.sztaki.hu/e-commerce/

If you want to test your own e-business skills you can connect to e-commerce skills portal at

http://www.iscn.com/projects/e-commerce/index.html



Address http://www.iscn.com/projects/e-commerce/index.html

Best Practise in E-Commerce and Job Role Qualifications

CAPABILITY ADVISER SKILL CARDS

The Underlying Skills Definition Model

The below figure only gives a simplified picture of the standard skills definition model.

A Domain, contains

Job Roles, which contain

Units, which contain

Elements, which contain

Performance Criteria, which must be proven by

Evidences.

Figure 12: The E-Commerce Skills Portal

5 Acknowledgements

We would like to thank the European Commission for supporting the e-Commerce Jobs projects under Leonardo da Vinci.

We would like to thank Dr Bernd Hindel and Mr Robert Treffny from ASQF (http://www.asqf.de) to trial the assessment portal with 15 German companies under the Bavarian SOQRATES initiative. We would like to thank all partners in the e-commerce jobs project.

6 References

[1] The Capability Model for IT-enabled outsourcing service providers – Carnegie Mellon University, School of Computer Science, Vol. 1 CMU-CS-01-163 – Hyder E., Kumar B., Mahendra V., Siegel J., Gupta R., Mahaboob H., Subramanian P. http://itsqc.srv.cs.cmu.edu/escm

[2] CREDIT Project, Accreditation Model Definition, MM 1032 Project CREDIT, Version 2.0, University of Amsterdam, 15.2.99

[3] DTI - Department of Trade and Industry UK, British Standards for Occupational Qualification, National Vocational Qualification Standards and Levels

[4] Feuer E., Messnarz R., Best Practices in E-Commerce: Strategies, Skills, and Processes, in: Proceedings of the E2002 Conference, E-Business and E-Work, Novel solutions for a global networked economy, eds. Brian Stanford Smith, Enrica Chiozza, IOS Press, Amsterdam, Berlin, Oxford, Tokyo, Wash-ington, 2002

[5] Financial Times Management (1997). Manage people, Unit C15 Project: Dealing with poor performance. Manchester: The Open College.

[6] Financial Times Management (1997). Manage people, Unit C12 Project: Working with your team for results. Manchester: The Open College

[7] Gartner (2002), IT Services and Sourcing: Business Imperatives In The Face Of Commercial Un-certainty http://www4.gartner.com/2_events/conferences/asm3i/asm3i.jsp

[8] Rajesh Gupta., The eServices Capability Model (escm) - For Excellence in IT-enabled Outsour-cing Services, in Proceedings of the EuroSPI 2002 Conference, Nuremberg, Germany, 18-20 September 2002

[9] Information Society Commission, Benchmarking Ireland in the Information Society, Irish Government, 1997

[10] Klarus, R., & Blokhuis, F. (1997). Accreditation of prior learning in the Netherlands. 's Hertogenbosch: CINOP.

[11] Messnarz R., Stubenrauch R., Melcher M., Bernhard R., Network Based Quality Assurance, in: Proceedings of the 6th European Conference on Quality Assurance, 10-12 April 1999, Vienna , Austria

[12] M. Biro, R. Messnarz, Key Success Factors for Business Based Improvement, in Software Quality Professional, Volume Two, Issue two, American Society for Quality, March 2000

[13] Messnarz R., Biro M., Velasco G., O'Suilleabhain G., The Victory Project - A Virtual Enterprise for SPI, in Proceedings of the EuroSPI 2000 Conference, Copenhagen Business School, 7-9 November 2000

[14] Messnarz R., Experiences with the Combination of Skill Cards and Process Assessments in an Integrated Improvement Framework, in: (eds.) Nora Koch, Software Process Improvement – CMM & SPICE in Practice, Verlag Uni Druck, Munich, Germany, December 2002

7 Author CVs

Dr Richard Messnarz

Dr Richard Messnarz made his MSc at the University of Technology in Graz in Austria. In 1995 he received a PhD from the same university for the work on a QUES - "A Quantitative Quality Evaluation System".

He has participated in a number of European research and industry projects since 1990. He was also the project leader of the Comett expert exchange project ISCN in 1993 which led to the foundation of ISCN itself in 1994.

He is the Executive Director of ISCN. He was the technical co-ordinator in previous process improvement initiatives such as PICO (Process Improvement Combined apprOach), Victory (Virtual Enterprise for Process Improvement), and the co-ordinator of a software engineering group within the pilot project for building a prototype of a virtual university. He is the editor of a book published in 1999 from IEEE "Better Software Practice for Business Benefit - Principles and Experience". He is currently the technical director of a large e-working initiative TEAMWORK, and manages in the initiative MediaISF the technical development for an e-working solution for news agencies collaborating on Eastern European enhancement topics.

He is a chairman and main organiser of the EuroSPI conference series (the former ISCN series).

He is a SPICE and BOOTSTRAP lead assessor.

Eva Feuer

Éva FEUER is Head of Quality Management, senior research associate at the Computer and Automation Research Institute. She has more than 20 years of IT and project management experience.

M.Sc. in mathematics and computer science (Jozsef Attila University, Hungary). She is certified ISO 9000 TÜV CERT auditor and BOOTSTRAP software process assessor. She managed Hungarian participation in European multinational projects and organisations (European Software Institute, Bootstrap Institute, Idealist-East Leonardo projects, Pass ESSI Pie project, Vasie ESSI project). She is head of the Information Society Technologies Liaison Office for the European Union's 5th Framework Programme in Hungary. Her main R&D fields are: information systems, software development, quality management, software process assessment, process improvement and software product quality audit. She has publications in international scientific journals and conference proceedings in the field of software quality management.

Damjan Ekert MSc

Damjan Ekert is a Master student at the University of Technology in Graz, Austria. He is working with ISCN LTD / ISCN GesmbH Austria since 2001. He is a chief programmer of the Capability Adviser system and has also been integrated into the team developing the NQA e-working platform solutions.

Natalia Costas Lago MSc

Natalia Costas Lago studied at the University of Vigo, Spain and graduated as a MSC in computer science. In a work placement at ISCN LTD Ireland she helped developing modules of the assessment portal in relationship with the generation of learning recommendations. Her special interests web based developments, and new Internet languages.

Gonzalo Velasco del Barrio

Mr Velasco made a master degree in International Trade at Valladolid Chamber of Commerce (Spain) and graduated 1997. He also made a honours degree for bachelor of arts in business administration at the university of Coventry (United Kingdom).

Since 1997 he worked in Spain on the development and management of EU, national and regional projects at FGUVA (General foundation to further the integration of Spanish regions into Europe) training and innovation departments.
He is currently the person in charge of the development and the management of two networks: Innovation Relay Centre Gallaecia (IRC) and the Envirdis Network dealing about innovation issues both of them are framed in the Innovation programme. He has experience in Leonardo Projects and other transnational projects.

Eugene O'Leary

TecNet is a strategic network of thirteen Irish Institutes of Technology (technical university equivalents). The network was established by the Council of Directors of the Institutes of Technology and is jointly funded by Enterprise Ireland (Ireland’s enterprise development agency).
Eugene O’Leary is chief executive of TecNet. He has worked in third level education as well as in SME’s and has expertise in information technology and its application to business development. He is an experienced project manager having worked with over twenty European Projects including, ADAPT, Integra, Leonardo, Socrates, Euroform and others. He is currently co-ordinating an IST FP5 project. He also sits on a number of Consultative Committees in the Irish Department of Education.

Bruno Wöran

Mr Wöran made a master degree in economics and graduated from the University of Economics in Vienna in 1991. He also made a master of business administration degree at the University of Nebraska, Oklahoma, and graduated there in 1993. In 1993 he then joined Austrian strategic EU institutions to further the Austrian integration into Europe. Since 1996 he is the managing director of Danube. DANUBE is the regional service and information centre for the European vocational training Programme LEONARDO DA VINCI and the 5th Framework Programme for research and technological development. At the same time DANUBE is consortium partner in the INNOVATION Relay Centre Austria, a network of technology transfer and validation centres throughout Europe. DANUBE is the co-ordinator of several pool contracts for university-enterprise transnational mobilities of students and graduates for 4 universities and 8 polytechnics and partner institutions in former Eastern Europe.

Risk Analysis as a Prioritizing Mechanism in SPI

Tor Stålhane, PhD
Gunhild Sivertsen Sørvig, MSc
Department of Computer and Information Sciences, Norwegian University of Science and Technology, Trondheim, Norway.

Abstract

This paper describes some of the work done in order to improve the development of a safety and business critical system in a Norwegian company. The approach differs from many other SPI efforts in three respects: (1) increased focus on cooperation with the developers, (2) the goal is to improve quality and gain market shares – not to cut costs, and (3) the consequences of system failures are used as a prioritizing mechanism for the SPI activities.

Keywords
System's risk assessment, Software Process Improvement

1 Introduction

This paper describes the work done by the authors for a Norwegian company during the spring of 2003. The company – which for business reasons wants to remain anonymous – wanted to have a look at the competitive advantages SPI could offer them.

In most SPI efforts, the goal of the company is to cut costs. For our company, however, the main reason for the SPI work was to improve the product and, in this way, to increase the marked share. Cost cutting is an attempt to maintain status quo and is thus a defensive strategy, while product improvement is an approach that tries to put the company on the offensive. That this is a good strategy is not an original idea – see for instance [8]. "Our" company already has a considerable market share but sees new opportunities if they can provide a good risk analysis report for their product and reduce the system's risk and vulnerability.

The two approaches cost cutting and product improvement were first contrasted and discussed in details in [1] and are summed up in table 1 below. The points that are of main relevance for our company were that they

- Looked at quality improvement as a challenge, not as problem solving.
- Did not try to avoid complaints but wanted to elicit compliments.
- Prioritized quality management, the voice of the customer and the common responsibility for quality over any rigid management control regime such as statistical process control - SPC.

	Evolution			Revolution		
Quality view	**Problem to solve - a necessity**			**Ideal function to be achieved – an opportunity**		
Motive	**Eliminate complaints**			**Elicit compliments**		
Step	**Inspect**	**SPC**	**QA**	**Strategic quality management**		
Focus	**Discover**	**Control**	**Avoid mistakes**	**Listen to the voice of the customers**		
Method	**Measure**	**Statistical technique**	**Programs and systems**	**QFD**	**System approach**	**Robust design**
Responsible	**Inspector**	**Production**	**Production, development**	**Everyone**		
Result	**Earn right to sell** **Keep market shares**			**Command higher price** **Gain market share**		

Table 1: Reactive vs. proactive improvement

2 The approach

Based on previous SPI experience [10], especially from the two national projects SPIQ and PROFIT, we took as our starting point the two maxims:

- Make sure the company knows you. This is important in order to achieve commitment and good communication. This is discussed in the chapter “Getting a good start”.
- Know the company. This includes the company’s history – why they are the way they are – and their strong and weak points. This is important if we want to change the culture and is discussed in the chapter “About the company”

The selected approach was as follows: We first made sure that we knew the company, its competitive advantages and its employees. Only then did the real work start. Our first activity was to do a functional risk analysis. After this, we analyzed all errors reported in a seven-week period in order to get an overview of the vulnerabilities of the system. These two analyses were then combined in order to produce a “worry list” that was used as a starting point for fault reduction activities. The top items in the “worry list” were analyzed together with the company’s development process in order to identify improvement opportunities – how to reduce the number of dangerous errors in future versions of the system.

3 Getting a good start

The first job was to establish good contact between the researchers and the company’s employees. The job started with a discussion meeting with management. This was useful to make sure they agreed to our approach. Then we held a presentation for all the employees – management, developers, administrative personnel etc. We participated in one of the company’s luncheon meetings. Everybody in the company participates in these meetings and they were thus the ideal forums for our presentation.

- We made sure that everybody in the company knew who we were, what we should do and why. This created confidence and trust between the employees and the researchers. In addition, we got lots of interesting input, just because they knew why we were there.
- We got immediate feedback on our ideas and observations and were thus able to correct misunderstandings. This helped us to have a good relationship right from day one.
- The luncheon meetings were later also used as a forum for presentation of results and findings and as a vehicle for information exchange.

4 About the company

4.1 Company history

The company started as a software department within a larger organization, which also was their only customer. When the mother organization decided that development and maintenance of software was outside their business area, the software development activity was established as a separate company with approximately 30 employees. However, they still kept close contact with the mother organization though a reference group and the mother organization was still their only customer.

A lot changed when the company got more customers. The main change was that they did not have close cooperation with all their customers any more and thus they could not anticipate their customers' future needs.

As a new method to investigate customer needs, they introduced what they now call demo-ware. This is a version of the system, with a new set of functionality, which is developed based on the company's domain knowledge and their expectations about future customers' needs. Through interactions with prospective users, this functionality is later changed into the functionality that the customers ask for and will buy.

4.2 The company's competitive advantages

Demo-ware helps the developers to stay in front of development but causes a large amount of design changes further down the line. In order to stay in business, the change process must be quick. Any changes to the development process that takes away the quick responses to customer needs will take away an important part of the company's competitive advantages.

In addition to the change process, the company also provides service to their customers, mostly as fixes to local problems, tuning the system to a particular environment etc. Their customers have learnt to expect and rely on top service from the company's developers. That this is important is clearly shown in [7]. The two competitive advantages identified above need to be kept in mind when we later discuss possible process changes.

In order to improve their competitive edge, the company wants to perform a risk analysis of the system. By doing this, the company hopes to achieve the following:

- Increase the customers' trust in the system and confidence in the development company.
- Help the customers to identify their own risks and thus enable them to build risk handling into their day-to-day operational procedures.

Be able to grade the sub-systems according to their risk and use this as a prioritizing mechanism for system component and development process improvement.

The proposed improvement activities must be considered together with the company's competitive advantages. The company faces three threats to these competitive advantages:

- The product may cause damage to the company's customers and in this way open up for lawsuits or large compensation claims.
- The development organization petrifies, and loses its ability to give quick and efficient service to

the users.

- The product contains many errors, which will make the users frustrated or dissatisfied and give the product a bad reputation.

The national marketplace for this type of system is small and the company's current and potential customers meet on several occasions each year to exchange experiences, gossip, and war stories. Thus, it is important to keep the customer satisfied and listen to the voice of the customers.

5 Risk and error analysis

5.1 Risk analysis

Based on earlier experiences, we decided to use UML use cases [6] as the starting point of the risk analysis [3]. UML use cases are easy to make and easy to understand. Given the variety of personnel that participate in the risk analysis, we wanted the risk and hazard analysis process to be simple and intuitive. An example of the use cases that we developed as a starting point is shown in figure 1 below:

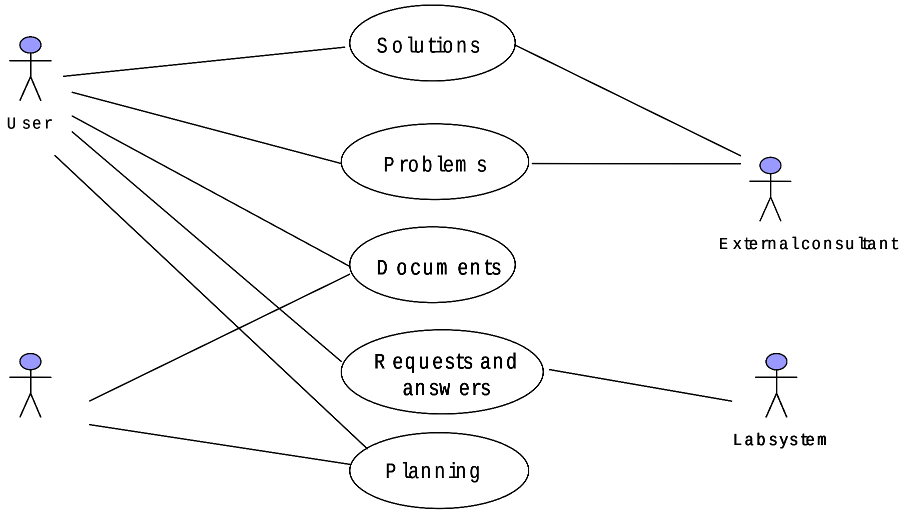


Figure 1: Use case diagram for an example subsystem

This use case diagram is a result of a 30 minutes brainstorming session with participation from developers and application experts. The use cases were documented on a whiteboard.

Since we had no study nodes, there was no point in applying a standard HazOp [2] based on guidewords. Instead, we chose to focus in the functionality offered by the system and performed a functional risk analysis. The process we used has the following steps:

1. Give the participants a short introduction to the idea of risk analysis and to the development and use of UML use cases
2. Develop uses cases for each subsystem of the system under consideration.
3. Perform a function-based HazOp for each use case. This was done by asking the participants two simple questions:
   - How can this function fail? This will give a list of failure modes.
   - Which consequences can this failure have for the stakeholders? This will give us a list of

hazards.

4. Document the results and obtain feedback. This was done by producing a failure mode versus risk table. An example is shown in table 2.
5. Assess the severity of each hazard.

Subsystem:			Risk		
Function	Failure mode description	Comments	User	Cust	Dev.

Table 2: Failure mode vs. hazard

The failure mode descriptions are supplied by developers and application experts. Different stakeholders may assign different severities to the same risk. Thus, we use three columns to identify the risks associated with each failure mode. In order to do this, we need a risk severity – stakeholder matrix. We have used three stakeholders: the development company, the service providers (the development company's customers) and the service receivers (the service providers' customers). An example of a risk vs. stakeholder table is shown below in table 3.

Risk level	Service receiver	Service provider	Developing company
High – 3	One person killed or several persons seriously injured	Several persons killed or many seriously injured	Bad press Lose customer
Medium – 2	One person seriously injured	User looks bad to his superior(s) or large sums lost	Dissatisfied customers or users
Low – 1	Minor irritant	Minor irritant, small amount of money lost or wasted	-

Table 3: Stakeholders' hazard assessment

In most cases, the risk score of the developing company will be the same as the risk score of the service provider. The reason for this is the high probability that unsatisfied service providers will leak the problem to the press in order to obtain maximum attention and compensation from the developing company in case of an accident.

In order to get a simple ranking of the subsystems, we combined the rankings in the following way: For each subsystem and stakeholder group we registered the number of functions that received each hazard value – a score form 0 to 3, where 3 denotes the most serious risk. We tested two ways to combine the function risk scores – weighted average and using the score given to the majority of the functions in the subsystem. In the latter case, the median was used if the majority criterion did not give a unique result. As it turned out, these two strategies gave results that were in close agreement and we thus selected the latter one since this was the simpler of the two.

The results from eight of the subsystems are shown in the tables below. Note that the real subsystem names have been replaced by the codes A1 – R2 in order to preserve the company's anonymity.

Subsys	Seriousness				Score	
	3	2	1	0	Avg.	Max.
D1	1	-	1	-	2.0	2
B1	-	-	3	2	0.6	1
O1	-	-	3	1	0.8	1
X1	1	3	2	1	1.6	2
I1	6	-	3	-	2.3	3
G1	2	1	1	-	2.3	3
M1	-	1	2	-	1.3	1
A2	1	1	2	-	1.8	1

Table 4: Grading of hazards from functional failures – service receiver view

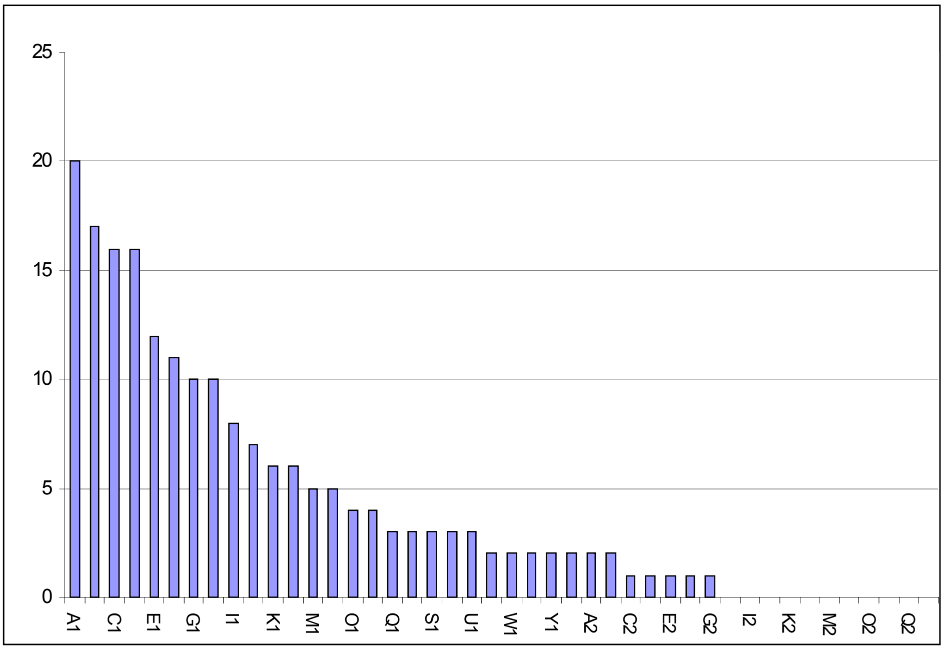
Subsys	Seriousness				Score	
	3	2	1	0	Avg.	Max.
D1	1	1	2	2	1.2	1
B1	-	4	3	-	1.6	2
O1	4	2	-	1	2.3	3
X1	1	4	2	-	1.6	2
I1	6	1	2	-	2.4	3
G1	2	1	1	-	2.3	3
M1	-	1	2	-	1.3	1
A2	1	1	2	-	1.8	1

Table 5 Grading of hazards from functional failures – service provider view

In order to combine the risk scores with the number of errors reported, we also need to analyze the reported errors. This is done in the next section.

5.2 Error analysis

We collected data for all errors reported over a seven weeks period. All reported errors were related to a subsystem – the largest unit in our hazard analysis. The first analysis of the error data – a Pareto analysis [4] – showed that 30% of the subsystems (13 subsystems) were responsible for more than 70% of all errors. The complete result from the Pareto analysis is shown in the diagram below.



.

Figure 1: Reported number of errors per subsystem

The plot shows that Pareto's rule of the "significant few and the insignificant many" holds also here. Many companies have used the Pareto plot alone to prioritize their improvement work. As mentioned before, our company decided to combine this information with the subsystem risk in order to get a better prioritizing mechanism. We show the effect of this choice in the next section.

5.3 Combining hazard and number of errors

To be correct, the risk should be computed by multiplying consequence – in our case hazard – with the hazard's probability. We did not, however, have data that could give us reasonable values for such probabilities. We thus decided to use the observed error rate as an indicator for failure probability.

From the analyses performed above, we can assign severity and number of errors found to each subsystem. The result of this is a table, which identifies the subsystem with the highest risk. Part of this table is shown below in table 6.

		Hazard		
		High (3)	Medium (2)	Low (1)
Number of errors registered	High 10	G1	B1	D1
	Medium 3 – 9	O1, I1		M1
	Low 0 - 2		X1	A2

Table 6: Subsystems' hazard and number of errors – service provider's viewpoint

From this table, we see that the subsystems O1, I1, G1 and B1 are our main concerns and that for instance D1 gets a low priority, even though it has more errors than the subsystems O1 and I1. This combination of number of errors and the risk, gives us a prioritized list of subsystems – our "worry list".

6 Improving the hazard analysis

The hazard analysis was considered useful by all participants, including the company's QA department and the application specialists. However, some of the feedback we received showed that there was room for improvement. In order to look into this, we performed a broad PMA [5] where all developers and application specialists participated. The top-level results of the KJ part of the PMA process are shown in the diagram below – positive experiences to the left, negative experiences to the right. Edges are used to connect boxes – topics – that influence one another.

The most important negative points raised were concerned with the preparation and planning activities. When we planned the sessions, we did not take into consideration that the participants had no previous experience in risk analysis. A good indicator for this problem was that they found it was difficult to grade the risks – everything became "very serious". In addition, they felt that a one hour session was too little time, given the newness of the concept. One of the subsystems did not fit into the chosen process and the developers responsible for this subsystem felt that the analysis process was too rigidly controlled by the coordinator. The main reason for the problems was that this subsystem did not have any user interaction and thus, a functional HazOp did not work. A traditional, guideword based HazOp would probably have worked better.

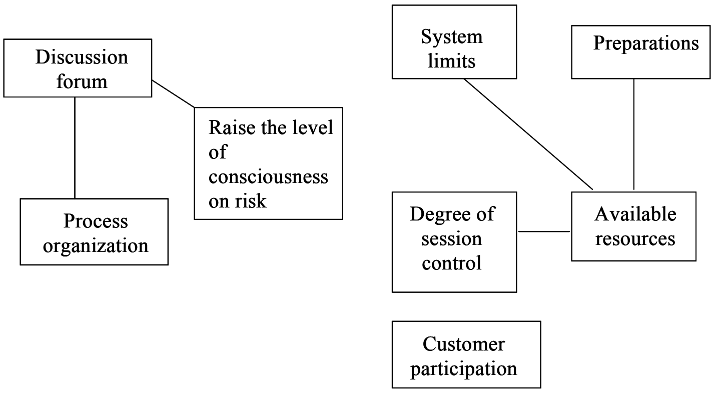


Figure 2: Top-level view of positive and negative points raised during the KJ process

We thought that the developers had complete knowledge of the functionality of the subsystems for which they were responsible and had a list of functions “ready in their heads”. This turned out to be a wrong assumption – people need time to sort out what they really know. Thus, some functionality was left out of the analyses, which again made it less credible, both for the developers and for the application specialists.

The most positive points were that the risk assessment process increased their level of consciousness of the risk inherited in using the system and that the process triggered important discussion between developers and application specialists. As a result of the points raised in the KJ process, we changed the process to the following:

1. Prepare use case diagram for the subsystems to be analyzed. This should be done based on the system’s function lists. Each developer should go through the use cases and familiarize himself with the subsystem’s functionality.
2. Start the risk assessment process with a warm-up exercise, for instance looking at the results from previous risk analysis. In addition, the risk assessment table (table 3) should be given a quick run-through.
3. Perform a function-based HazOp for each use case. This is done by asking the participants two simple questions:
    - How can this function fail? This will give a list of possible failure modes.
    - Which consequences can this failure have for the service receiver, the service provider and the development company? This will give us a list of possible hazards.
4. Document the results. Use the hazard assessment table shown in table 3 above for this purpose.
5. Assess the severity of each hazard.

This process will be used for the next risk analysis in our company.

7 Development improvement opportunities

As a first step in the development improvement process, we organized a brain storming session with all developers responsible for our top priority subsystems. The process started by considering the following question: “What are the most important reasons for errors in this subsystem?” Due to the rather limited topic area, we used a focused PMA [5]. The following were the most important reasons for errors:

1. Need to move the product from demo-ware to a system that fits the customer’s needs. Most design errors fell into this category.
2. Missing info or application knowledge.
3. Different requirements from different customers – even though they are in the same application area. Different sites have different needs, rules and regulations.
4. Missing initialization of variables
5. Missing updating of one or more status variables.

6. No systematic design or planning for the handling of errors through exceptions.
7. Misunderstandings or bad planning of graphical user interfaces.

Problems 1, 2 and 3 on the list are, to a large extent, due to the need to build demo-ware. The rest of the problems can – at least partly - be removed. We continued the improvement process by looking for answers to the question: "What can we do to reduce the probability of repeating these types of errors?"

Tempting as it might seem, the introduction of change control boards, formal change reviews etc. which are popular in the SPI literature, are not viable solutions in our case, since they will add overhead and thus remove an important part of the company's competitive advantages. The developers already feel quite strongly the squeeze between the need to test more and the need to give quick service to the customers. The solutions thus had to be found somewhere else. The following is a short summary of the proposed process changes:

- Code checklists, containing the most common coding errors.
- Some kind of "buddy programming". It is important that this activity does not take too much time. Experience from pair programming in XP shows that this is a sensible approach [9]. A formal review is not what they want – at least not at the present.
- Introduction of automatic testing tools.
- More team work – especially in the early phases of development. Working in pairs was mentioned also here.

These proposals will mostly concern problems 4, 5 and 7. Problem 6 will require innovative thinking. How the proposals stated above will be realized is not yet decided but an important action will be to employ more developers. This activity is already under way.

8 Conclusions

During our work, we have seen that:

- It is important to know the company's competitive advantages before doing process improvement in order not to do anything that will reduce this important asset.
- It is important to establish good communication with everybody in the development company right form the start. This also improves the flow of info and feedback.
- For products where a failure can have safety implications, it is important to use risk or hazard as a prioritizing mechanism for software process improvement.
- A simple way to get a risk analysis process running is to start with a standard version of the process and adapt it to the company's specific needs by using all or part of a PMA.

Our company has – without knowing about it – strived for the goals identified by H.L. Oh [1]. This has been done in the following way:

- Their motive is to improve the product's quality – "Elicit compliments"
- Everyone is responsible for improving the quality of the product. The company has a QA manager, but his job is to help the developers to achieve their quality goals, for instance by giving them extra resources. They do not have a quality department since they feel that this would just add administrative overhead and destroy their possibility to act quickly on customer requests.
- Their goal is to "Command higher prices and gain market shares". By using the customers' risk as a prioritizing mechanism, they have shown that they put the customers' interests ahead of their

own needs to save money. This is consistent with their goal: to make money by increasing their market share - not by cutting costs.

Our part has been to assist the company on their way to increasing the product quality in a way that the company feels is compatible with their culture and important to their customer. The fact that they got a new, large contract during our work indicates that their approach is right.

9 Literature

[1] Oh, H.L., "A Changing Paradigm of Quality", IEEE Transactions on Reliability, vol. 44, no. 2, June 1995

[2] MoD, Def. Standard 00-88: "HazOp studies on Systems Containing Programmable Electronics – Part 2", issue 2, UK MoD, May 19, 2000

[3] Kari Juul Wedde, Tor Stålhane and Eric Swane: "Designing safety critical systems using UML", SINTEF project 402856 memo, November 17, 1999.

[4] David Straker "A Toolbook for Quality Improvement and Problem Solving", Prentice Hall 1995, ISBN 0-13-746892-X

[5] Tor Stålhane et al., "Post Mortem – An Assessment of Two Approaches", EuroSPI 2001, October 10 – 12, Limerick, Ireland

[6] Martin Fowler and Kendall Scott, "UML Distilled – A Brief Guide to the Standard Object Modelling Language", Addison-Wesley Longman Inc., 2000, ISBN 0-201-65783-X

[7] Tor Stålhane, Per C. Borgersen and Kjell Arnesen "In Search of the Customer's Quality View", Proceedings of AQuIS 1996, Firenze, Italia

[8] Søren Lauesen and Otto Vinter, "Preventing Requirement Defects: An Experiment in Process Improvement", Requirements Engineering, vol. 6, 2001, Springer-Verlag, London

[9] Ken Auer and Roy Miller: "Extremer Programming. Playing to Win". Addison Wesley 2002, ISBN 0-201-61640-8

[10] Geir Kjetil Hanssen et al., "SPI – easy in theory, hard in practice", EuroSPI 2002, Nuremberg, Germany

10 Author CVs

Tor Stålhane

Tor Stålhane was born 1944 in Skein, Norway. He received his MsC. in Software Engineering from the Technical University of Norway – NTH - in 1969. The same year he started working for the software group at SINTEF where he worked in compiler development. In 1988 he received a PhD. in applied statistics. The title of the thesis was “A Load Dependency Model for Software Reliability”. He then went back to SINTEF where he worked in the areas of systems safety assessment and software process improvement. In 2000 he changed to the Norwegian University of Science and Technology where he became a full professor in Software Engineering. He is a member of IEEE and the Norwegian Society for Statisticians.

Common Criteria for IT Security Evaluation - SPI Analogies

Prof. Dr. Miklos Biro
Budapest University of Economic Sciences and Public Administration
miklos.biro@informatika.bke.hu

Abstract

Analogies between complex systems are useful for helping those familiar with one of them to better understand the other one, and for establishing cost and resource effective unified systems. The principle is applied in this paper to the ISO/IEC 15408 (Common Criteria) IT Security Evaluation standard, software quality evaluation standards and the Capability Maturity Model Integration (CMMI[®]).

Keywords

Security, Quality, Assessment, Evaluation, Capability, Maturity, Classification, Categorization

[®] CMMI is registered in the U.S. Patent & Trademark Office by Carnegie Mellon University

1 Introduction

Security is naturally present in all systems of software product quality criteria, and plays a significant role in the approporiate implementation of many software and systems engineering process areas. The development of the Information Society made this criterion of even higher significance, which resulted in the distinguished attention of international standardization bodies for example, resulting in the ISO/IEC 15408 (Common Criteria) standard.

Certification needs and the constraints of the standardization process led to the flexibility in both the product standards (ISO/IEC 9126, ISO/IEC 14598) and the process methodologies (CMMI, ISO/IEC 15504) which allows for evaluation modules based on a more elaborated background (ISO/IEC 15408, ISO/IEC 12207) as well as other modules based on simpler measurements.

Even if some of the underlying standards evolved independently of each-other, the discovery of analogies between their structure can contribute to the establishment of a cost and resource effective multiple certification process [Taylor, Alves-Foss, Rinker, 2002].

The combination of software process and product quality standards has already been studied in [Boegh, Régo, 2000]. In this paper we examine the analogies between the ISO/IEC 15408 (Common Criteria) standard and software quality and process capability evaluation methodologies.

2 The Common Criteria

The history of the ISO/IEC 15408 (Common Criteria~CC) standard goes back to the 80's with the following non-exhaustive list of milestones:

1980- **TCSEC:** Trusted Computer System Evaluation Criteria (USA)

1991 **ITSEC:** Information Technology Security Evaluation Criteria v 1.2
(France, Germany, the Netherlands, U.K.)

1993 **CTCPEC:** Canadian Trusted Computer Product Evaluation Criteria v 3.0

1993 **FC:** Federal Criteria for Information Technology Security v 1.0 (USA)

➔ **CC Editorial Board**

1996 **CC v 1.0 ➔ ISO Committee Draft (CD)**

1998 **CC v 2.0 ➔ ISO Committee Draft (CD)**

1999 **CC v 2.1 = ISO/IEC 15408**

CC v 2.1 consists of the following parts:

Part 1: Introduction and general model

Part 2: Security functional requirements

Part 3: Security assurance requirements

It is a common perception that understanding the Common Criteria (CC) evaluation process **requires painstakingly inspecting multiple documents and cross referencing innumerable concepts and definitions** [Prieto-Díaz, 2002]. The first challenge is the digestion of the abbreviations of which here is a brief extract for our immediate purposes:

TOE: Target of Evaluation — An IT product or system and its associated administrator and user guidance documentation that is the subject of an evaluation.

TSP: TOE Security Policy — A set of rules that regulate how assets are managed, protected and distributed within a TOE.

TSF: TOE Security Functions — A set consisting of all hardware, software, and firmware of the TOE that must be relied upon for the correct enforcement of the TSP.

PP: Protection Profile — An implementation-independent set of security requirements for a category of TOEs that meet specific consumer needs.

ST: Security Target — A set of security requirements and specifications to be used as the basis for evaluation of an identified TOE.

EAL: Evaluation Assurance Level — A package consisting of assurance components from Part 3 that represents a point on the CC predefined assurance scale.

Here is an illustrative list of the classes of security functional requirements discussed in Part 2 of the CC:

FAU Security audit

FCO Communication

FCS Cryptographic support

FDP User data protection

FIA Identification and authentication

FMT Security management

FPR Privacy

FPT Protection of the TOE security functions

FRU Resource utilisation

FTA TOE access

FTP Trusted path / channels

The following are classes of security assurance requirements discussed in Part 3:

ACM Configuration Management

ADO Delivery and Operation

ADV Development

AGD Guidance Documents

ALC Life Cycle Support

ATE Tests

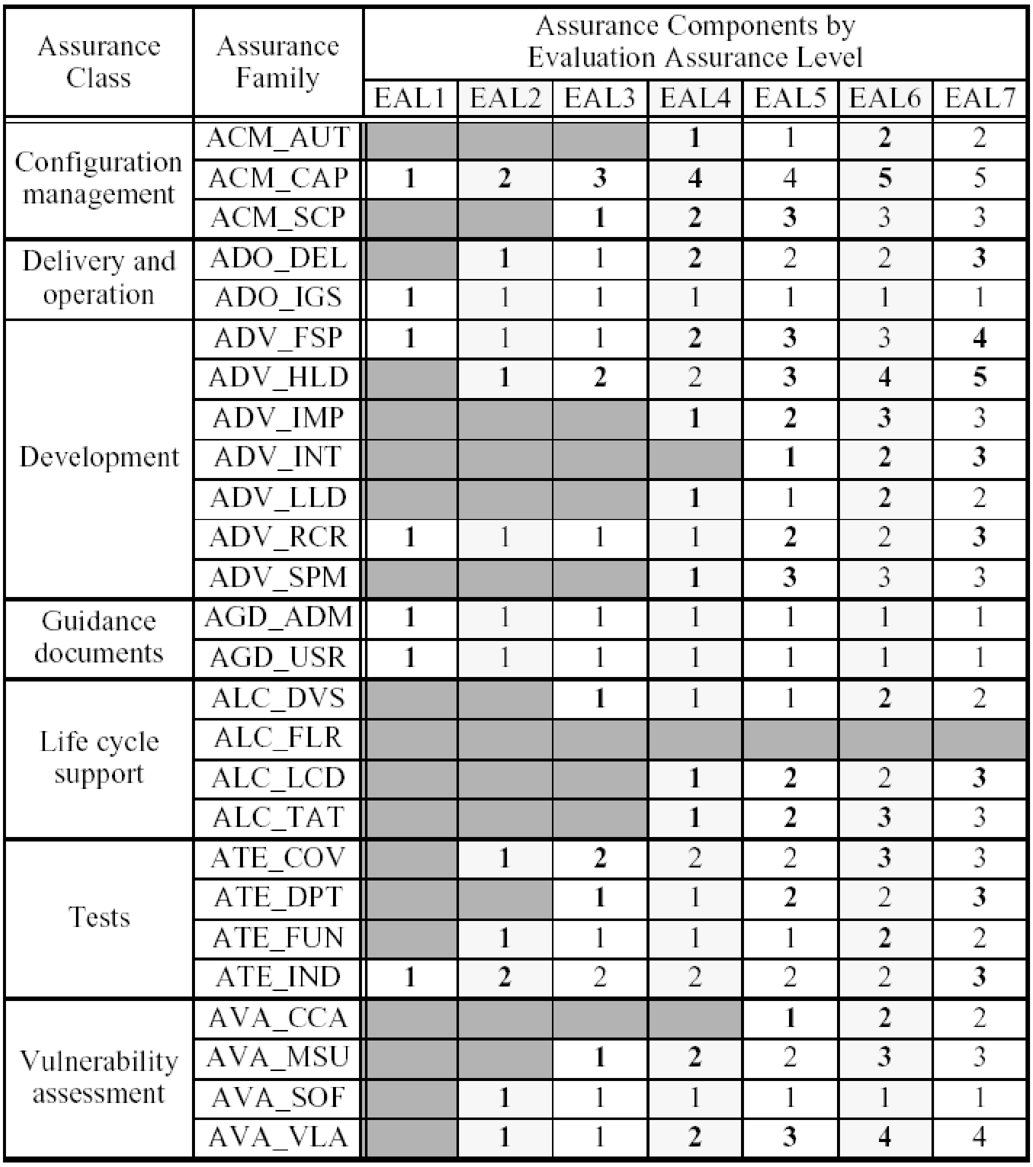
AVA Vulnerability Assessment

AMA Maintenance of Assurance

APE Protection Profile Evaluation

ASE Security Target Evaluation

And finally, table B.1 from Appendix B of Part 3 of CC v 2.1 which describes the relationship between the evaluation assurance levels and the assurance classes, families and components.



Assurance Class	Assurance Family	Assurance Components by Evaluation Assurance Level						
		EAL1	EAL2	EAL3	EAL4	EAL5	EAL6	EAL7
Configuration management	ACM_AUT				**1**	1	**2**	2
	ACM_CAP	**1**	**2**	**3**	**4**	4	**5**	5
	ACM_SCP			**1**	**2**	**3**	3	3
Delivery and operation	ADO_DEL		**1**	1	**2**	2	2	**3**
	ADO_IGS	**1**	1	1	1	1	1	1
Development	ADV_FSP	**1**	1	1	**2**	**3**	3	**4**
	ADV_HLD		**1**	**2**	2	**3**	**4**	**5**
	ADV_IMP				**1**	**2**	**3**	3
	ADV_INT					**1**	**2**	**3**
	ADV_LLD				**1**	1	**2**	2
	ADV_RCR	**1**	1	1	1	**2**	2	**3**
	ADV_SPM				**1**	**3**	3	3
Guidance documents	AGD_ADM	**1**	1	1	1	1	1	1
	AGD_USR	**1**	1	1	1	1	1	1
Life cycle support	ALC_DVS			**1**	1	1	**2**	2
	ALC_FLR							
	ALC_LCD				**1**	**2**	2	**3**
	ALC_TAT				**1**	**2**	**3**	3
Tests	ATE_COV		**1**	**2**	2	2	**3**	3
	ATE_DPT			**1**	1	**2**	2	**3**
	ATE_FUN		**1**	1	1	1	**2**	2
	ATE_IND	**1**	**2**	2	2	2	2	**3**
Vulnerability assessment	AVA_CCA					**1**	**2**	2
	AVA_MSU			**1**	**2**	2	**3**	3
	AVA_SOF		**1**	1	1	1	1	1
	AVA_VLA		**1**	1	**2**	**3**	**4**	4

Table 1

3 Enlightening Analogies

The above sample from the CC naturally raises a lot of questions whose answers would require the already mentioned inspection and cross referencing of multiple documents including hundreds of pages. As an introductory alternative approach, the analogies below offer a shortcut to those who already have a basic understanding of models of software quality and process capability.

CC certification is performed after the system is developed. In this sense, CC is closer to the software product quality evaluation standards ISO/IEC 9126, ISO/IEC 14598, and their follow-up being developed under the acronym SQUARE (ISO 25000 Software Quality Requirements and Evaluation).

As far as the ISO/IEC 9126 standard is concerned, the **classes of security functional requirements** and the **classes of security assurance requirements** are analogous to the **high-level quality characteristics**, while the **requirement families** to the **subcharacteristics**. **Evaluation Assurance Levels** (EAL) can be simply interpreted as **measurement results on an ordinal scale** analogously to measurements of subcharacteristics in ISO/IEC 9126.

A key concept of ISO/IEC 14598 is that of the **evaluation module**. "An evaluation module specifies the evaluation methods applicable to evaluate a quality characteristic and identifies the evidence it needs. It also defines the elementary evaluation procedure and the format for reporting the measurements resulting from the application of the techniques." It also defines its own scope of applicability. In other words, an ISO/IEC 14598 evaluation module defines a consistent set of requirements and procedures for evaluating a quality characteristic independently from the concrete product, but depending on its application environment. If we consider the concept of **Protection Profile** (PP) as an implementation-independent set of security requirements for a category of TOEs that meet specific consumer needs, as introduced above, we can immediately see the analogy.

Even-though CC certification is performed after the system is developed, its structure shows a striking analogy with the system of continuous and staged representation structures of the Capability Maturity Model Integration (CMMI[®]). In order to highlighting the analogy, let us consider the table in Appendix F of version 1.1 of the CMMI Continuous Representation showing the process area capability level (CL) target profiles of the Continuous Representation making an organization's maturity level equivalent to a maturity level (ML) defined in the Staged Representation.

[®] CMMI is registered in the U.S. Patent & Trademark Office by Carnegie Mellon University

Name	Abbr	ML	CL1	CL2	CL3	CL4	CL5
Requirements Management	**REQM**	**2**	Target Profile 2				
Measurement and Analysis	**MA**	**2**					
Project Monitoring and Control	**PMC**	**2**					
Project Planning	**PP**	**2**					
Process and Product Quality Assurance	**PPQA**	**2**					
Supplier Agreement Management	**SAM**	**2**					
Configuration Management	**CM**	**2**					
Decision Analysis and Resolution	**DAR**	**3**	Target Profile 3				
Product Integration	**PI**	**3**					
Requirements Development	**RD**	**3**					
Technical Solution	**TS**	**3**					
Validation	**VAL**	**3**					
Verification	**VER**	**3**					
Organizational Process Definition	**OPD**	**3**					
Organizational Process Focus	**OPF**	**3**					
Integrated Project Management (IPPD)	**IPM**	**3**					
Risk Management	**RSKM**	**3**					
Organizational Training	**OT**	**3**					
Integrated Teaming	**IT**	**3**					
Organizational Environment for Integration	**OEI**	**3**					
Organizational Process Performance	**OPP**	**4**	Target Profile 4				
Quantitative Project Management	**QPM**	**4**					
Organizational Innovation and Deployment	**OID**	**5**	Target Profile 5				
Causal Analysis and Resolution	**CAR**	**5**					

Table 2

Let us equivalently transform this table so that the last columns contain maturity levels instead of capability levels, and the cells underneath contain the capability level of the given process area necessary for achieving the given maturity level.

Name	Abbr	ML	ML1	ML2	ML3	ML4	ML5
Requirements Management	**REQM**	**2**	**-**	**2**	**3**	**3**	**3**
Measurement and Analysis	**MA**	**2**	**-**	**2**	**3**	**3**	**3**
Project Monitoring and Control	**PMC**	**2**	**-**	**2**	**3**	**3**	**3**
Project Planning	**PP**	**2**	**-**	**2**	**3**	**3**	**3**
Process and Product Quality Assurance	**PPQA**	**2**	**-**	**2**	**3**	**3**	**3**
Supplier Agreement Management	**SAM**	**2**	**-**	**2**	**3**	**3**	**3**
Configuration Management	**CM**	**2**	**-**	**2**	**3**	**3**	**3**
Decision Analysis and Resolution	**DAR**	**3**	**-**	**-**	**3**	**3**	**3**
Product Integration	**PI**	**3**	**-**	**-**	**3**	**3**	**3**
Requirements Development	**RD**	**3**	**-**	**-**	**3**	**3**	**3**
Technical Solution	**TS**	**3**	**-**	**-**	**3**	**3**	**3**
Validation	**VAL**	**3**	**-**	**-**	**3**	**3**	**3**
Verification	**VER**	**3**	**-**	**-**	**3**	**3**	**3**
Organizational Process Definition	**OPD**	**3**	**-**	**-**	**3**	**3**	**3**
Organizational Process Focus	**OPF**	**3**	**-**	**-**	**3**	**3**	**3**
Integrated Project Management (IPPD)	**IPM**	**3**	**-**	**-**	**3**	**3**	**3**
Risk Management	**RSKM**	**3**	**-**	**-**	**3**	**3**	**3**
Organizational Training	**OT**	**3**	**-**	**-**	**3**	**3**	**3**
Integrated Teaming	**IT**	**3**	**-**	**-**	**3**	**3**	**3**
Organizational Environment for Integration	**OEI**	**3**	**-**	**-**	**3**	**3**	**3**
Organizational Process Performance	**OPP**	**4**	**-**	**-**	**-**	**3**	**3**
Quantitative Project Management	**QPM**	**4**	**-**	**-**	**-**	**3**	**3**
Organizational Innovation and Deployment	**OID**	**5**	**-**	**-**	**-**	**-**	**3**
Causal Analysis and Resolution	**CAR**	**5**	**-**	**-**	**-**	**-**	**3**

Table 3

The analogy between table 1 and table 3 is immediately apparent if we consider the following equivalence of the concepts of the Common Criteria and of CMMI:

Common Criteria	CMMI
Assurance Family	Process Area
Evaluation Assurance Level (EAL)	Maturity Level
Assurance value	Capability Level
Classification of Security Requirements	Categorization of Process Areas

Table 4

This analogy not only helps those already familiar with CMMI to better understand the Common Criteria, but provides a new perspective on CMMI itself as well.

4 Conclusion

The analogies discovered between the complex standards and methodologies described in the paper help those familiar with one of the systems of concepts better understanding the other system of concepts on the one hand, contribute to the potential establishment of a cost and resource effective multiple certification process on the other hand.

5 Literature

Biró,M.; Tully,C. The Software Process in the Context of Business Goals and Performance. Chapter in the book entitled Better Software Practice for Business Benefit (ed. by R. Messnarz, C. Tully). (IEEE Computer Society Press, Washington, Brussels, Tokyo, 1999) (ISBN 0-7695-0049-8).

Biró,M.; Messnarz,R. Key Success Factors for Business Based Improvement. Software Quality Professional (ASQ, American Society for Quality) Vol.2, Issue 2 (March 2000) pp.20-31. (http://www.asq.org/pub/sqp/past/vol2_issue2/biro.html)

Boegh,J.; Rêgo,C.M. Combining software process and product quality standards. Presented at the 2nd World Conference on Software Quality, Japan, September 2000.

Prieto-Díaz,R. Understanding the Common Criteria Evaluation Process. Commonwealth Information Security Center Technical Report CISC-TR-2002-003, September 2002.

Taylor,C.; Alves-Foss,J.; Rinker,B. Merging Safety and Assurance: The Process of Dual Certification for Software. In Proc. Software Technology Conference, March 2002.

6 Author CV

Dr. Miklós Biró

Dr. Miklós BIRÓ (miklos.biro@bkae.hu) is a professor at the Department of Information Systems of the Budapest University of Economic Sciences and Public Administration.with 26 years of software engineering and university teaching (including professorship in the USA), and 16 years of management experience. He has a Ph.D. in mathematics (operations research) from the Loránd Eötvös University in Budapest, an Executive MBA (Master of Business Administration) degree from ESC Rouen, France, and a Master of Science in Management degree from Purdue University, USA. He is fluent in Hungarian, English, and French.

He is a Bootstrap and SPICE (Software Process Improvement and Capability dEtermination - ISO/IEC 15504) assessor. He gives Ph.D. courses and company training courses on software quality management, and on the Capability Maturity Model - Integrated (CMMI - service mark of Carnegie Mellon University, USA).

He initiated and managed the Hungarian participation in numerous European multinational projects and organisations committed to software process impovement (European Software Institute, Bootstrap Institute). He was the initiator and head of the Information Society Technologies Liaison Office in Hungary for the European Union's 5th Framework Programme. He is invited as expert consultant by Hungarian and international organizations (European Commission; Irish National Policy and Advisory Board for Enterprise, Trade, Science, Technology & Innovation~Forfás; Communications Authority of Hungary; Hungarian Committee for Technological Development; Investment and Trade Development Agency of Hungary; Hungarian Airlines; United Nations Industrial Development Organization~UNIDO; International Software Consulting Network;...).

He has numerous publications in international scientific and professional journals (Software Process Improvement and Practice, Software Quality Professional (1, 2), Software Process Newsletter, European Journal of Operational Research, Zeitschrift für Angewandte Mathematik und Mechanik, Optimization, Information Processing Letters, Discrete Mathematics, Journal of Advanced Transportation, Acta Cybernetica) and conference proceedings. He is the co-author of Hungarian and English language books on operations research models, software engineering, software process improvement and business motivations.

He is member of the Editorial board of the journal Software Process Improvement and Practice published by Jonh Wiley & Sons, and founding president of the professional division for Software Quality Management of the John von Neumann Computer Society. He is the Hungarian member of Technical Committee 2 (TC-2) Software: Theory and practice of the International Federation for Information Processing (IFIP) . He is member of several other professional bodies and societies.

A Framework For Risk Management For Software Projects

Rajesh Gupta, PMP , Gupta_Rajesh@satyam.com

ABSTRACT

Satyam, like many other organizations, find itself in stormy and uncharted water as the business environment has become more complex and competitive than ever before. Further, uncertainty in the environment has increased. Therefore, Satyam needs a comprehensive risk management model to deal with such challenges. This paper provides a framework – called **SCORE** (**S**atyam **CO**mprehensive **R**isk **E**valuation) framework that will enable Satyam software projects to not only reduce the impact of potential threats(risks) but also maximize the positive impact of opportunities (risks) that may come (or brought!) in the way of Satyam.

The SCORE framework suggests two levels of risk assessment: (a) High Level Risk Assessment, and (b) Function Level Risk Assessment. The framework also provides the (actionable) steps, sub-steps and activities that are needed to conduct a High Level Risk Management. For conducting a Function Level Assessment, the framework not only provides set of (actionable) steps/ sub steps and activities are provided, it also provides a set of questionnaires (Appendix –A) that can be used by software project teams to assess risks. The framework is an outcome of the author's experience in the field of software quality, discussion with peers at Satyam and elsewhere.

The next part of the paper provides a framework to deal with both types of risks – threats and opportunities. The last section is devoted to use of project management measurement. It provides an insight on how a project management measurement can be gainfully used to forecast a risk and makes one ready to deal with it. Due care has been taken to ensure that the framework suggested here is implement able and provides a road map for implementation in Satyam.

Key words

Risk management, SCORE framework, High Level Risk Assessment, Function Level Risk Assessment, project management measurement

1 Introduction: Risk Management Imperative

Heraclitus (540BC – 480 BC) once said, "*The only thing permanent is change and everything else is in a state of instability*." I believe, no statement other than this – made several thousand years ago - reflects the truth of today's business environment.

The business environment within which Satyam operates is becoming increasingly complex and certainly more competitive than ever before. The innovation cycle in industry is becoming shorter and shorter. During the course of a business, within a given time, many factors change like:

- The customer's requirements (additional features needed, changes in organizational environment)
- The market forces – demand, supply
- The underlying technology platform
- The system and operational environment
- Product changes due to problems
- The underlying software environment (new versions from Microsoft appear every six months)
- The management requirements (different initiatives, ISMS, Six Sigma)
- The development team changes
- Business factors (a developing company stops further development of a product because revenue can rather be made by other services or a substitute product)
- The politics (e.g. in the defense sector, the budget is reduced by half, consequently all the vendors are affected)

No doubt, many of these factors did exist even before. What has changed now is the rate of change, which is increasing by each passing day. The business environment is now changing rapidly and it poses a different type of challenges for an organization like Satyam in adapting itself continuously, and "creating higher value enduringly" – as Satyam's new initiative Orbit 5 aims at.

Now in a software project, key informations like customer requirements, our total efforts, end-results, total costs and many other such parameters become 'clear' only at the end of the project. The effect of rapidly changing business environment on our projects can be summarised in Table 1

Table 1: Changing Trend In SoftwareProjects

Project related Information	Moving from	Moving to
• Requirements	Clear	Ambiguous
• Rate of change	Gradual	Sudden
• Execution	Precise	Chaotic
• Project teams	Specialists	Generalists
• Project complexity	Low	High
• Customers	Demanding	More and more demanding

Executing a software project in today's business context is inherent with an increased level of uncertainty.

No wonder, risk management has become one of the most important and sought after project management knowledge areas for organizations like Satyam. Another reason to deal with risks effectively is derived from the insight that more and more software projects are reportedly failing in different organizations, generating high unplanned costs [1, 3,4]

Table 2: The Software Crisis Is Still On!

The Software Crisis Is Still On! [Source: 8]
• 75% of all large scale software systems fail • One -third of all IT projects is canceled before completion. • Over 1/2 of project budgets exceed 189% of original estimates • Average schedule overrun for projects in trouble is 222%. • Delivered product contains only 61% of originally specified features.

On the other hand, there are organizations that are making 'unplanned' profits also because they could utilize an unexpected opportunity successfully to their advantage**.**

For Satyam – a SEI CMM Level 5 company, any failure – even nearer to this magnitude - will spell only doom for the organization. Leave alone the additional effort due to re-work resulting in additional cost and dissatisfaction of its clients, the reputation of Satyam will be at stake. This is especially important when Satyam is planning to position itself as a truly world class organization that will be known for its leadership in cost, quality and commitment.

Besides, there are risks involved due to business volatility. Between 1990- 2001, out of 100 larger American corporations, 26 % lost their market capitalization by over 67%. None of them lost in the previous decade. (Source:http://www.concordens.com/Riskmgmtapp/Bus_volatility.htm). Appendix B provides a list of a few companies that have failed over past a few years. The reason: Poor risk management.

Organizations, large or small, suddenly have found themselves in stormy and uncharted waters. In other words, the risks in our project have increased. And Satyam is no exception. The risk management imperative has been born.

Satyam needs to not only deal with the negative consequences of an uncertainty but also has to maximize the positive outcome of opportunities that might be coming on its way. Thus, Satyam's definition of Project Risk Management includes a set of practices and support tools to identify, analyze and respond to project risks such that results of positive events (opportunities) are maximized and the consequences of negative events (i.e. threats) are minimized or eliminated. Risk management deals with both risk (the known distribution of potential outcomes) and uncertainty (the unknown distribution of potential outcomes).

The objective of an effective risk management program is not always to eliminate risk but to manage it. Satyam should pick and choose among the types of risks and degree of exposure taking on those where Satyam has a competitive advantage and laying off onto the capital markets or insurers those where you do not. One can also view this as a boundary question: What activities does Satyam do inside the firm versus what activities should be externalized to its customers and suppliers?

Risk management, like technology, R&D, distribution, brand, quality, cost position, and so forth, can be a source of either competitive advantage or disadvantage depending on how it is used.

Theodore Roosevelt once said: "Risk is like a fire - you can control it and use it to your advantage, if left uncontrolled it will rise up and destroy you...". Thomas Gilb [1] warned us by quoting: "If you don't actively attack your risks, they will attack you!"

This brings us to the question of how risk management can be implemented on a practical day-to-day usage basis, really helping project teams to establish an easy way of dealing with risks effectively.

2 Risk management in theory

There is plenty of literature available on risk management, of which [11, 12,13] are probably the most well known publications. Beyond this, there are interesting papers on this topic [5, 14] as well as number of magazine articles on risk management [10, 11].

Figure 2 shows a typical approach of how risk management is seen in the literature (a similar figure can also be found in [5]). While these kinds of figures give the impression that managing risks[1] is an easy to implement technique, it is a much too fuzzy picture when trying to get this work. It is simply not sufficient as a guideline for practitioner.

In [2], Sommerville and Sawyer introduce risk management within a short two-page section. At least they mention that "Assess Requirements Risks" is an advanced method causing" moderate costs.

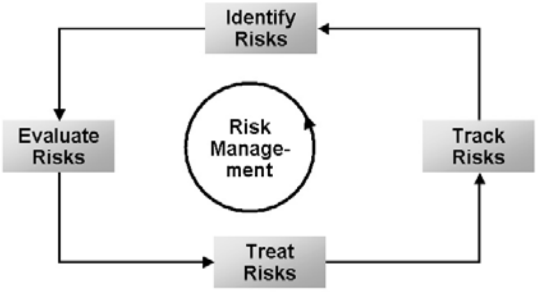


Figure 1: Typical Risk Cycle

[1] The term 'risk management' may be misleading because it implies that one can control events. On the contrary, risk management involves preparing for possible future events rather than reacting to events as they occur.

The requirements management standard work of Robertson and Robertson [3] picks up the risk topic as an integral part of requirements management as well, while stating that they only include a short reflection. Furthermore, they refer the reader to further (more complex) reading material.

The various publications from Software Engineering Institute USA (SEI) deal, amongst other things, with the human aspect of risk management and psychological aspects [5,14]. While team communication is the focus of a SEI-paper by Higuera and Gluch [5], there is another paper by Van Scoy [14] that refers to risks as challenges and opportunities rather than (negative) problems and defines three key elements of a successful risk management process (refer to Figure 2):

- **Identify**: Where and how can we find our project risks?
- **Communicate:** How do we communicate about risks?
- **Resolve:** How to solve detected risks?

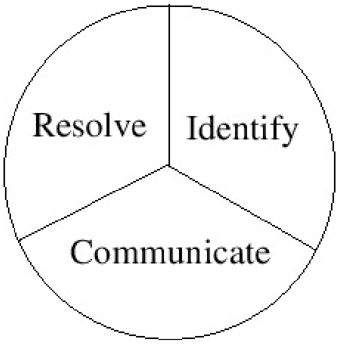


Figure 2: Elements of a successful risk management

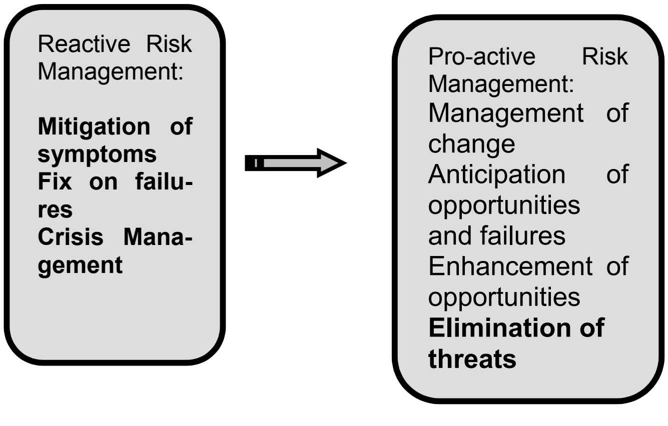
3 Accepting the practical challenge

While we, at Satyam, are aware of many such concepts on risk management, it has turned out, during several discussions with my peers in the organization, that most of these approaches can at best be categorised as theoretical. This indeed means that most people engaged in risk management questions lack of practical guidance. Nevertheless, the procedure on Risk Management (QP601RM1/4.0) and guidelines for Software Management (QWRMIW01/4.0) enumerated in Satyam's QUALIFY do provide some guidance. But, they have limited applicability in today's overall business context of Satyam.

The reasons for a practitioner's fears may be found in some words of Charette: "I feel that books (like my own books) can give no more than an overview of an approach, but less than the entire story. They don't enable the team to do the entire job. There is much more to it than could be put in a book." [13].

We must recognize the fact that the core goal of a software project in Satyam is to deliver a software product, not to carry out risk management. Therefore, as a supportive process, risk management should not introduce its own bureaucracy; nevertheless, it should be effective enough to safeguard us from any adverse consequences. Keeping this in mind, an attempt has been made here to present a framework for management of risks in software projects in Satyam, in the next part of the paper. We call this framework as **SCORE** (**S**atyam's **CO**mprehensive **R**isk **E**valuation) framework for risk management.

The framework for risk management presented here is based on my extensive research on this topic, discussion with my peers in Satyam and elsewhere, and my experience with my several clients whom I had interaction as part of my consulting assignments. The **SCORE** framework is based on a simple philosophy that Satyam now needs to move from a mindset of 'reactive' risk management towards 'pro-active' risk management. This is shown in figure 3.0



Reactive Risk Management:

Mitigation of symptoms
Fix on failures
Crisis Management

Pro-active Risk Management:
Management of change
Anticipation of opportunities and failures
Enhancement of opportunities
Elimination of threats

Figure 3**: Satyam now needs to move from a mindset of 'reactive' risk management towards 'proactive' risk management**

Operationally, a 'pro-active' risk management consists of two steps: (a) Risk assessment – that in cludes identification of risks and their evaluation, (b) Risk Response Development – defining enhancement steps for opportunities and responses to threats. [6]. A third step could be to deal with the risks that come as 'problems' over the course of the project; needless to add that this is a 'reactive' part of risk management. This is responding to a problem that was a risk earlier.

This paper provides a framework for handling risks on proactive basis. The framework excludes any reference to dealing with problems (i.e. reactive part of risk management) because handling problems is often a time specific and situation specific solution.

Figure 4 below [7] gives a comprehensive risk model, highlighting some 70 different risk categories.

In SCORE framework, we have NOT selected all the 70 categories but only a few of these that are most appropriate for a software project in Satyam (This information is based on discussing with several QRs in Satyam). Because most major risks cut across traditional functional or divisional lines, the SCORE Risk framework highlights many major risks that might be overlooked by ordinary "bottom-up" planning systems that are presently followed in many of the business units in Satyam.

3.1.1 Environmental risk Competitor Capital availability Financial markets	Sensitivity Catastrophic loss Regulatory	Shareholder relations Sovereign / Political Industry Legal
Process risk		
Operations risk Customer satisfaction Human resources Product development Efficiency Capacity Performance gap Cycle time Sourcing Obsolescence / shrinkage Compliance Business interruption Product/ service failure Environmental Health and safety Trademark / brand name erosion	**3.1.2 Empowerment risk** Leadership Authority / limit Outsourcing Performance incentives Change readiness Communications Information processing / technology risk Relevance Integrity Access Availability Infrastructure **3.1.3 Integrated risk** Management fraud Employee fraud Illegal acts Unauthorized use Reputation	**3.1.4 Financial risk** Price Interest rate Currency Equity Commodity Financial instrument Liquidity Cash flow Opportunity cost Concentration Credit Default Concentration Settlement Collateral
Information for making decision on risks		
3.1.5 Operations	**3.1.6 Financial**	**3.1.7 Strategic**
Pricing Contract commitment Performance measurement Alignment Completeness and accuracy Regulatory reporting	Budget and planning Completeness and accuracy Accounting information Financial reporting evaluation Taxation Investment evaluation Regulatory reporting	Environmental scan Business portfolio Valuation Performance measurement Organization structure Resource allocation Planning Life cycle

Figure 4**: gives comprehensive risk areas, highlighting some 70 different risk categories [Source: 7, Arthur Anderson]**

In SCORE framework, risk assessment - the first component of risk management, involves understanding the amount of risk present in a project. As one might expect, assessing risk could be labour intensive depending on how much effort one devotes to it. To avoid expending more efforts than necessary assessing risk, SCORE framework suggests for two levels of risk assessment, as under:

- High level risk assessment
- Function- level risk assessment

High-level risk assessment is a broad review of the project risks. It is the first step in the assessment process and is used to determine whether the risk associated with this project warrant a more detailed, function-level risk assessment. It does not involve formal quantification or extensive study.

Function-level assessment is a review of each project management function to determine the risk associated with each function. It is more detailed review than the high level assessments.

4 Framework for Conducting a High Level Assessment

A high-level risk assessment is always done at a project start-up and should also be done at other points in the life of the project whenever conditions change significantly. The purpose of conducting a high-level risk assessment is to identify project risks that need an in-depth examination. A high level assessment can be accomplished with a little time investment.

A high-level risk assessment involves examining four project attributes. These are: (a) project size (b) project structure (c) project technology (d) project impact. Three categories of assessment for risk are suggested: high, medium and low. These categories are clearly subjective and require additional descriptions suiting to each business unit in Satyam, to be useful.

A high – level assessment consists of four steps:

- Step 1: Review and assess project size.
- Step 2: Review and assess project structure.
- Step 3: Review and assess project technology.
- Step 4: Review and assess project impact.

Actions to achieve each of the above four steps are given below (Table 3 – 6):

Table – 3: Actions to achieve Step 1

	Step 1: Review and assess project size
Sub-step	**Action**
1.1	Estimate total project size in terms of the following: • Function points • Lines of codes • Pages of documentation
1.2	Compare project size to other projects previously accomplished by this business unit
1.3	Categorize the risk as high. Medium or low (H, M, or L) according to the relative size of the project compared with the team's experience. Note: The assessment considerations relate to the overall expertise level of the composite team that will do the work, however, do not ignore expertise that can be tapped from other parts of the organizations.

Table – 4: Actions to achieve Step 2

	Step 2: Review and assess project structure Project structure refers to the complexity of business processes and organizations associated with the project.
Sub-step	**Action**
2.1	Determine project structure by considering the following: • No. of customers and Satyam business units involved • Geographical locations of customers and Satyam business units • Scope of business functions ad

	dressed • Timing of project relative to other business or technology changes
2.2	Compare project structure to other projects previously accomplished by this business unit.
2.3	Categorize the risk as high. Medium or low (H,M, or L) according to the relative complexity of the project compared with the team's experience.

Table – 5: Actions to achieve Step 3

Step 3: Review and assess project technology It may appear to be premature to assess project technology when the design phase of the project is not yet completed. Each assessment, however, should be conducted with whatever information is known or can be presumed at the time. Delaying the assessment should be avoided until more data is collected.	
Sub-step	**Action**
3.1	Determine the hardware approach for the proposed system solution using the following considerations: • Previous use of this technology • Maturity of the technology • Vendor track record
3.2	Determine the software approach for the proposed system solution using the same considerations, as in sub step 3.1
3.3	Compare hardware/ software approaches to other projects previously accomplished by this business unit.
3.4	Categorize the risk as high. Medium or low (H,M, or L) according to the relative similarity (that is closeness of fit).

Table – 6: Actions to achieve Step 4

4.1.1 Step 4: Review and assess project impact	
Sub-step	**Action**
4.1	Determine the impact of this project in terms of strategic importance by answering the following questions: • How does this fit with the customer's strategies? • How does this project fit with Satyam strategies?
4.2	Determine the criticalness of the business functions addressed by the project (for example, real time electronic funds processing versus batch processing)
4.3	Determine the impact of the project on the customer organization in terms of changes in the way the business is run.
4.4	Determine the impact of the project failure on the customer organization (for example, will project failure put the customer organization out of business)
4.5	Categorize the risk as high. Medium or low (H, M, or L) according to the project impact.

Use of High Level Assessment:

After completing the high- level risk assessment procedure, the following guidelines could be used to determine whether further assessment is required:

- If all four assessments are low, one need not conduct further risk assessment at this point in the project life cycle. One should still plan, however, to reassess risk at key points in the life of a project.
- If any of the four risk assessments is medium or high, one should proceed with a function level risk assessment.

A high-level risk assessment shall be done at a project start-up and should also be done at other points in the life of the project whenever conditions change significantly – especially, when clients' requirements get altered drastically.

Table 7 explains which level of risk assessment to select for your project

Table 7

If	Then…
Any of the four assessments from the high-level risk assessments is medium or high	Proceed with a function-level risk assessment
Risk assessment is high from either the high level or function –level risk assessment, and further detail is required	Use available mathematical models. (Table 8)

If a more detailed risk assessment approach is required, there are mathematical models that can help quantify risk potential. Exploring these techniques in detail is outside the scope of this paper, but Table 8 displays some of the available tools and corresponding components, formulas, and references.

Table 8

Tools	Developer	Components (Examples)	Risk formula (Examples)	References
IEEE	Institute for Electrical and Electronics Engineer	Structure, size, technical	Weight x probability	Software Risk Management (only a sample)
DoD Mathematical Model	Department of Defense Systems Management College	Hardware and software maturity; complexity, and dependency on interfaces	(Consequences + probability) – (Consequences x probability)	Systems Engineer Management Guide
Taxonomy Mechanism	Software Engineering Institute	Requirements, processes, and environments	Calculate averages and point spread	SEI Software Risk Management Program

Example of High-Level Assessment Use

Table 9 provides an example of a high-level risk assessment worksheet. Nevertheless, one needs to establish one's own limit for each criterion.

Table - 9

Project Attributes	Risk Assessment Worksheet		
	Low	Medium	High
Size	Less than 10,000 lines of code	10,000 – 100,000 lines of codes	Greater than 100,000 lines of code
Structure	Immediate customer department and immediate Satyam team	Multiple customer departments; one to three additional business disciplines	Multiple customer departments plus multiple Satyam business units
Technology	Existing IBM mainframe platform	Client/serve approach using technology already in use on his account	New technology not previously used on this account
Impact	Existing, non-critical business functions	New business functions that are not critical	New, critical business functions

5 Framework for Conducting a Function- Level Assessment

The purpose of conducting a function level assessment is to obtain a more detailed evaluation of project risk than that provided by the high level assessment. By identifying risk for each project management function, one is able to pinpoint the appropriate risk management techniques that should be applied.

The primary vehicle for conducting the function-level assessment is the risk assessment job aid located in the appendix. One needs to use the steps in Table 10 to conduct a function level assessment.

Table 10: Steps to conduct Functional level assessment

Step	Action
1	Review the risk assessment questionnaire and, if necessary, tailor it to the project.
2	Using the questionnaire, review risk items for each function and categorize the degree of risk for each item
3	Group risk items according to high, medium or low risk
4	Document medium and high risk items for the project

Step 1: Review the function– level risk assessment questionnaire and, if necessary, tailor it to the project (The risk assessment questionnaire is given in the **Appendix – A – please refer Annexure A for better understanding of subsequent steps.**)

Objectively review the questionnaire to determine whether all questions are applicable. Modify questions to fit your project situation, being careful not to slant the questionnaire so that the results are distorted.

Step 2: Using the questionnaire, review risk items for each function and categorize the degree of risk for each item.

For each risk item, there are descriptions under the headings (high, medium, and low) to determine which description most closely matches the situation in the project.

Example: The following example addresses the commitment of senior customer management to the project solution:

--L Customer senior managers are enthusiastic and back up their words with actions. Customer managers are personally involved and actively work to ensure the right resources are available. A project champion is clearly identified.

-- M Customer senior management is adequate. Although customer senior managers are not personally in volved, they ensure that the right resources are available.

-- H Customer senior management is either unknown or reluctantly supportive. Appropriate resources are not always available when needed.

Step 3: Group risk items according to high, medium or low risk.

Grouping risk items helps us to focus on the items of greater concern. Generally, the items assessed low will not require further consideration.

Step 4: Document medium and high-risk items for the project.

For each item risk item, a brief statement may be enumerated describing the specific situation for this project. The phrasing in the questionnaire may be used to guide our thinking. The questionnaire statements associated with low risk (please refer example in step 2) could be interpreted as the ideal state, against which the medium and high-risk items may be contrasted.

Example: Table 11 shows how to document a risk item specific to the Customer Relationship function.

Table 11: An example on documenting a risk item

Function	Risk Degree	Rationale
Customer Relation-ship	Medium	• Only 2 of the 14 users in the target department are considered experts in the application and both are tied up with other work. • Three others have a good understanding of the application and are available • Most of the people in the department have previous project experience.

6 Use of Function- Level Assessment Results

The results of the function-level assessments are used as inputs in the development of risk handling plans. The function-level assessment will be used to determine the degree of emphasis required in risk handling.

High-Level Versus Function- Level Assessment

Table 12 provides the differences in descriptions and purposes of high level and function- level risk assessments.

Table 12:

Level of Risk	Purpose	Description
High Level	Identify those risks that need more in-depth examination	Broad review of project risk.
Function -Level	Identify risk for each project management function to help pinpoint appropriate risk management techniques that should be applied	Individual review of each project management function

7 Risk Response Development

This section of the paper, on risk response development, deals with developing plans to (a) minimize the impact of negative risks (threats) on the project, and (b) enhance the impact of opportunities, thereby enhancing the probability for project success.

The next section 7.0 offers a technique for establishing appropriate measurement that will provide early warning of future problems.

It is important to realize that the risk response development plan – prepared to handle each risk item – must be integrated in the overall project plan so the project manager has to deal with only one plan. Risk response development involves defining enhancement steps for opportunities and responses to threats. Responses to identified threats generally fall into one of three categories:

- Risk mitigation
- Risk transfer
- Risk acceptance

Let us call these three categories of responses as '**Risk Handling Techniques'**. Table 13 describes these three techniques for handling risks (threats).

Table 13

Risk (Threat) Handling Technique	**Definition**	**Example**
Mitigation	The act of revising the project's scope, schedule or budget (preferably without significant schedule or quality impact on the project's objectives) to reduce uncertainty.	Choosing an existing database management system over a new developed and relatively untried alternative
Transfer	The act of transferring all or part of risk to another party, usually by some form of contract	Setting a firm, fixed price for work subcontracted to a third party
Acceptance	The acknowledgement of the existence of risk but a decision to make no changes to the project plans and accepts the probability of risk occurring. Acceptance should include developing a contingency plan, which is a plan to be invoked only if the risk item occurs.	Committing to the scheduled project end date, even though planned ramp-up of personnel requires critical skill sets currently in short supply. A contingency plan might be identifying another Satyam support group to call upon if needed.

7.1 Framework for using Risk (Threat) Handling Techniques

Risk handling techniques help you evaluate the alternatives to address risk items. Too often a project manager in Satyam selects the first option that presents itself or assumes that all risks must be mitigated.

The alternative is considered based on any or all of the three risk techniques or combination of them. Following is a suggested framework to determine alternatives to use risk-handling techniques:

- Do not assume that all risks must be mitigated. If the probability of the risk occurring is sufficiently low, accepting the risk makes a good business sense.

- Consider all the three techniques to develop risk-handling alternatives for each risk item.
  - What are the ways to mitigate this risk?
  - Can transfer be used to reduce this risk?
  - What are the consequences of accepting this risk?

Example: The following scenario is an example of the use of multiple techniques in determining risk-handling alternatives.

This is a case of a Health Care application in Satyam that required a new hardware. A high risk was identified as: The hardware vendors might fail to deliver specified workstations in time to complete implementation of the new software.

Table 14 lists techniques to consider when developing each risk-handling alternative.

Table 14

Risk Handling Technique	**Alternatives**
Mitigation	• Choose a different configuration that uses more readily available hardware. • Chose a different hardware vendor with a more reliable de

	livery record. - Adjust the application development schedules to keep the hardware delivery off the critical path.
Transfer	- Include contractual performance incentives for developing hardware ahead of schedule. - Include contractual performance penalties for developing hardware late.
Acceptance	- Proceed with original plan but develop contingency plans to use alternative hardware in Satyam's possession in the event vendor fails to deliver on time. - Proceed with original plan and accept the consequences of late hardware delivery

After risk items are identified, determining and selecting an alternative is a fairly standard process. The key point to remember here is that the selected alternative needs to be in line with the project's objectives. The framework in table 15 helps us to determine and select risk-handling alternatives.

Table 15:

Step	Action	Techniques
	Determine the criteria for selecting alternatives	Review project objectives Review customer information acquired Determine measurable, objective criteria that will help differentiate alternatives (For example, "must meet planned end dates regardless of cost")
2	Examine each assessed risk item	Consider each risk item individually Consider combinations of risk items to determine whether there are common root causes.
3	Identify the alternatives to resolve each risk item	Use the risk handling techniques to identify alternatives Examine combinations of alternatives to see whether risk-handling solutions can be leveraged.
4	Evaluate the alternatives and select the best alternative	Weigh each alternative against the selection criteria Select the alternative having the closest fit with the selection criteria
5	Develop plans for the alternative selected	Plan the tasks needed to handle the assessed risk Validate plans against project objectives and adjust if necessary Incorporate the risk-handling plans into the overall project plans

7.2 Framework for using Risk (Opportunity) Handling Techniques

The risk handling techniques in section 6.1 have assumed risks as threats, whereas in reality, an uncertainty could be viewed as an opportunity. Satyam needs to proactively mould all such opportunities to its advantages. It is recommended to adopt '**6E**' framework to handle such risks (opportunities). These are:

1. **E**stimate the opportunity
   - Size of opportunity
   - Determine which events are likely to occur and are advantageous for Satyam.

2. **E**valuate the opportunity to determine if it is aligned with :
   - Satyam organizational objectives or project objectives, and
   - Objectives of other stakeholders (e.g Client's objectives)
3. **E**xplode the opportunity: If the identified opportunity meets the criteria of step 2, then:
   - Explore further and maximize the size of opportunity, say enhance number of

occurrence of the events or/and increase (positive) impact of each event
- Enlarge the business case

4. **E**stablish the process to use opportunity
   - Determine the process (people, methods and resources) to maximize the positive impact of the opportunity and reduce adverse effects of the event, if any.
   - If it is a case of 'repeat' opportunity, Satyam needs to look into the option to're-use' the process that was determined last time. Care has to be taken to incorporate modifications that were suggested as a part of 'lessons learnt' last time.

5. **E**xploit the opportunity
   - Satyam may decide to use the opportunity as it occurs, or exploit the 'explode' opportunity as determined in step 3, to its advantage.

6. **E**xtend the opportunity:
   - After having exploited the opportunity once, ensure that appropriate steps, wherever possible, are taken so that such opportunities are repeated in future.

7.3 Relationship Between Risk Handling Plans and the Project Plan

After developing plans for handling risks, immediately integrate these plans into the project plan. As new risk items are identified, risk-handling plans are continually integrated in the project plan throughout the life of the project. Effective management of risks is directly dependent on this integration, because execution of risk handling plans is accomplished as part of the process of executing the project plan.

The use of the suggested framework here in itself is an excellent mitigation technique. Incorporating the Satyam project metrics and project management methodologies in our project plan greatly reduces project risk (threat).

8 Use of Project Management Measurements To Forecast Risks

In addition to using risk assessment to develop risk-handling plans, we can use risk assessment to determine the degree of emphasis to place on the project measurement measurements. For risks that were assessed as having medium to high risk, especially where we have chosen acceptance technique, appropriate project management measurements provide an early warning system to alert us of emerging problems.

The purpose of relating risk management to project management measurements is to ensure that the level of measurement details is appropriate to the degree of risk. This maximizes the effectiveness of our project management measurements, ensuring that neither too little nor too much time and money are expanded on measurements. It must be recognized that one will never be able to track every aspect of the project in detail as the project progresses.

Table 16 describes and provides examples of some project management measurements. The measurements are annotated according to the following metrics categories:

- Business performance, including system engineering metrics
- Financials
- Customer Satisfaction
- Human Resources

Table 16:

Measurement	Category	Description	Examples
Effort	Business performance	Number of person-hours to complete a task	Person hours to complete Technical Design Phase Person Hours to develop the event/response model Person hours to unit test program XYZ
Size	Business	Function	Number of

	performance	points, lines of codes, or pages of documentation	lines of code in the claim processing sub system Number of pages in user guide
Cost	Financials	Satyam expenses incurred to complete a task	Cost ($000s) to develop the Business Design document Total travel costs Cost of capital equipment
Satisfaction index	Customer satisfaction	External Customer satisfaction Measurement (ECSM) score or any other survey score used as benchmark for future surveys	External Customer satisfaction Measurement (ECSM) survey
Training	Human resources	No. of hours of training expended	Hours of Jawa training per employee per year Hours of project management training during project start-up.

Granularity: A key point to remember in selecting measurements for a project is the amount of granularity needed. Granularity is the level of detail at which measurement data are collected. The levels of detail range from gross to fine. Measurements, which are taken at the summary level, or taken less frequently, would be considered of gross granularity. Measurements that are taken at the detail level, or taken more frequently, would be considered of fine granularity.

Selecting the right level of granularity for measuring a project is like deciding how often to check the fluid levels in our car. One of the factors influencing this decision might be the difficulty in performing the measurement. Another factor is the consequences of neglecting to check fluid levels regularly. Running out of engine oil may cause car engine to break down because of improper lubrication, resulting in costly repairs. On the other hand, running out of windshield washer fluid may present temporary difficulties, but a resolution is usually as close as the next service station.

Thus, our decision on frequency of measurement is influenced by the risk probability, on which we may have historical data to base our decision. If the car is new, one will check the oil frequently. After one has gained experience with the car and found that one need not add oil very frequently, the oil may be checked at less frequent interval.

8.1 Relationship Between Risk Management and Project Management Measurements

The risk assessment helps to determine the granularity of project management measurements. The higher the risk, the more important it is to invest time and money in project management measurement. It is important to reassess risk periodically to see whether some measurements needs to be collected at a high level or low level. The relationship between risk and project management measurement is portrayed in table 17

Table 17

Degree of Risk	Degree of project Management Measure-ment	Granularity	Example 1	Example 2
High	Complex	Fine	Task Level	Daily data
Medium	Moderate	Medium	Activity level	Weekly data
Low	Simple	Gross	Phase level	Monthly data

8.2 Using Risk Assessment

Table 15 provides a framework one can apply to monitor risks according to the relative risks within several example project management functions. A framework used for a given degree of risk and function should not necessarily be applicable for other degree of risk.

Also, if reassessment reveals changes in risk items, the granularity of the project measurement is likely to change too.

Table 18

Function	**Degree of Risk**	**Sample Technique**
Communication	Low	Managing by walking around (MBWA); monthly status meeting
	Medium	Associate delight survey, open forum
	High	Reverse open door; highly structured standards for meeting and memos; weekly project meetings
Schedule	Low	Bar charts; action item list
	Medium	Critical Path Method (CPM)
	High	CPM using resource leveling
Quality	Low	Processes in Satyam's QUALIFY
	Medium	Open defects count
	High	Mean time to failure
Financial	Low	Profit & Loss Variance report
	Medium	Cost data tied to measured effort
	High	Earned value analysis
Resources	Low	Time tracking in person months
	Medium	Time tracking in person hours at project level
	High	Time tracking in person days or person hours by milestone

Using risk management effectively ensures that one does not spend too much or too little time and money on project management measurements. Typically, the more complex the project measurements are, the greater their costs. The right amount of measurement will serve as an early warning of problems.

9 Conclusion

For those who anticipate and strategically manage business opportunities and problems, the gains can be enormous. But to realize these gains, Satyam needs to have [8]:

- Global perspective: Viewing software development within the context of the larger systems-level definition, design and development.

  Recognizing both the potential value of opportunity and the potential impact of adverse effects.

- Forward looking view: Thinking toward tomorrow, identifying uncertainties, anticipated potential outcomes

  Managing project resources and activities while anticipating uncertainties.

- Open communication: Encourage free- flowing information at and between all project levels

  Enabling formal, informal, and impromptu communication.

  Using processes that the value the individual voice (bringing unique knowledge and insight to identifying and managing risk)

- Integrated management: Making risk management an integral and vital part of project management

  Adapting risk management methods and tools to a project's infrastructure and culture.

- Continuous process: Sustaining constant vigilance

  Identifying and managing risks routinely through all phases of the project's life cycle.

- Shared product vision: mutual product vision based on common purpose, shared ownership, and collective communication

  Focusing on results.

- Team work: Working cooperatively to achieve common goal

  Pooling talents. Skills and knowledge

The SCORE framework for risk management, as suggested in this paper, is based on these principles. It would help software projects at Satyam to understand and manage the changes and uncertainties (from both internal and external sources) in a much more effective way. Thus, it will help Satyam to achieve its business goals and "create value enduringly' that Orbit 5 initiative aims it.

But be aware that just having a technical framework does not necessarily solve all problems in risk management!

One should remember the following:

The Need For Knowledge And Action

If we don't know, we can not act

If we can not act, the risk of loss is high

If we do know and act, the risk is managed

If we do know and do not act, we deserve the loss

Dr. Mikel J. Harry

References

[1] T. Gilb, Principles for Software Engineering Management, Addison-Wesley, 1988

[2] I. Sommerville, P. Sawyer, Requirements Engineering, Jon Wiley & Sons, New York, April 1997

[3] S. Robertson, J. Robertson, A Systematic Approach to Requirements Management, Addison-Wesley,

[4] D. Leffingwell, D. Widrig, Managing Software Requirements: A Unified Approach, Addison-Wesley, Oktober 1999

[5] R. Higuera, D. Gluch, A. Dorofee et. al.: An Introduction to Team Risk Management, Software Engineering Institute, Carnegie Mellon University, Special Report CMU/SEI-94-SR-1, May 1994

[6] A Guide to the Project Management Body of Knowledge (PMBOK® Guide) , PMI, 2000

[7] Anderson A, A Framework for Strategy in Risk andRegulation' , supplement to LSE Magazine, December 1999, Washington Brooking Institution 1999

[8] http://www.sei.cmu.edu/programs/sepm/risk /principles.html

[9] http://www.concordens.com/Riskmgmtapp /app_summary.htm

[10] http://www.willis.com/sfs/Downloads/ Volumetric_Hedge.pdf

[11] B. Boehm, Software Risk Management, IEEE Computer Society, 1989

[12] C. Jones, Assessment and Control of Software Risks, Yourdon Press, 1993

[13] R. Charette, Software Engineering Risk Analysis and Management, McGraw Hill, 1989

[14] R. L. van Scoy, Software Development Risk: Opportunity, Not Problem, Technical Report CMU/SEI-92-TR-30, ESC-TR-92-030, September 1992

About the author:

Rajesh Gupta *is Process Consultant (Quality) in Outsourcing Process Consulting Group.*

He is co-author of recently released capability model for IT enabled services, called eSourcing Capability Model (escm), by CarnegieMellon University, USA. (itsqc.srv.cs.cmu.edu/escm)

He is the recipient of 'Best of the Best' paper award for Asia Pacific SEPG 2002, arranged by QAI.

Maturing Measurements in SimCorp along with the Organisation's Maturity

Malene M. Krohn, SimCorp A/S

Abstract

The main points of this article are that:

- The ability to perform measurements is very much dependent on the maturity of the software development processes in the organisation.
- Working with measurements is a dynamic process. As the organisation matures its' software development processes it is important that the measurement program is matured and adjusted accordingly.

This article is a case story describing the experiences with implementation of a measurement program in SimCorp A/S. The experiences and findings presented in this article are based on several years of work with software process improvements including registration of work flow related data and measurements.

Keywords

Measurements, Metrics, Measurement program, Maturity, Software Process Improvements, Quality Goals

SimCorp A/S

SimCorp A/S is a Danish software company, which develops and globally markets integrated investment and treasury management systems. SimCorp A/S has approx. 550 employees (whereof approx. 300 are employed in the headquarters in Copenhagen and the rest in subsidiaries in Europe, New York and Sydney). The development of SimCorp's main software product – SimCorp Dimension – is centrally located in Copenhagen, where approx. 150 employees work with software development. The product SimCorp Dimension is a standard product and a new version is released every sixth months. For more information about SimCorp A/S please refer to www.simcorp.com.

SimCorp A/S has worked with software process improvements (SPI) for the past 5 years and achieved a CMM level 2 certificate in 2001. For the past 2 years SimCorp A/S has worked with a measurements program and this article outlines some of the lessons learned from this program.

1 Summary

Developing the Measurement program is closely related to the maturity of the organisation. In SimCorp A/S the measurement related activities and the achievements gained have to a high degree been dependent on the maturity of the development organisation. The illustration below gives an overview of the course of events:

	Maturity	Measurement related activities	Achievements
Getting started…	Before CMM	Create foundation for measurements	Culture for data registration established
	CMM Level 1	Ad hoc measurements carried out Measurements primarily established on project level	Project related measurements[1]: Planning & Tracking measurements implemented Product related measurements[1]: Product size and quality measurements Increased predictability Ability to make corrective actions
Progressing in Maturity	CMM level 2	Measurements on project level Begin to standardise measurements	Quality goals and measurements implemented Tool for Measurements developed (portal) Increased product quality
	Progressing	Measurements standardised and established on organisational level Analysis activities initiated Continuous adjustments and extensions of Measurement program	Implementation of Measurement program Process related measurements[1]: Effect of Quality Assurance activities on Product Quality Improved decision basis

Below a summary of the lessons learned during the period is presented. The summary and this article are composed of two major parts representing the chronological events related to measurements:

- Getting started with Measurements, and
- Progressing in Maturity

The findings, lessons learned etc., that will be elaborated throughout this article, will be related to these two periods.

1.1 Getting started with Measurements

Data registration – the foundation for measurements

It takes time to build the foundation for measurements, hence it is important to start registering data. It may seem senseless and frustrating in the beginning, as the quality of data varies, and processes are uncontrolled etc. Focus should be on the areas (projects, processes etc.) where registrations can facilitate improvements.

[1] Zahran divides measurements in 3 categories: **Process related** (e.g. process defects, cycle time measures, quality and productivity trends etc.), **Product related** (e.g. number of defects, system size, usability etc.) and **Project related** (e.g. work completed, effort, costs, resource usage etc.) [Zahran 1998].

Initial approach – Assessment and Metrics work group

Having an external evaluation (assessment) of the organisation's maturity can be an eye-opener for many people in the organisation. It can also be a kick starter for process improvement activities including working with measurements. It is important to accept that working with measurements is a learning process. Realise that measurements take time and effort.

Collect input from the organisation

Advices and best practices from other organisations are plentiful, but it is important to select measurements that make sense to your organisation. We found that project related measurements (e.g. progress of project plans) and product related (e.g. size of product and number of errors) made most sense when initiating measurements. Process related measurement were on the other hand very challenging, probably because of our immature processes.

Align measurements with the maturing processes

It is important to closely relate the measurements to the processes in focus in the organisation. There is a close relationship between improving a process and measuring the very same process: 1) When you start measuring a process, it naturally puts focus on the process and that affects the process performance for a period. 2) Improving a process does also to some extent enable the measurement of the process, e.g. a process is supported by registering data, defining input and output etc.

1.2 Progressing in Maturity

Align measurements with the organisation's processes

As the organisation matures it is important to constantly align the measurements with the processes. Measurements have to be adjusted and new requirements for measurements will be produced. Two process areas had an essential effect on our ability to perform measurements: A structured development lifecycle and establishment of quality goals.

Standardising measurements at the organisational level

Standardised measurements are enabled by the processes maturing. Standardisation ensures a common way of interpreting status and progress of the software development activities and provides a better basis for decision making. It is important to use the measurements already established in the projects, when defining standard measurements to be used at organisational level.

Implementation of the measurement program

The implementation of the measurement program should be carefully planned. It should be defined how to follow up on the measurements. Measurements will be questioned! Often when introducing measurements people will feel that they are being monitored, and communication about purpose, consequences etc. is therefore vital.

Analysis – taking measurements a step further

Having established a foundation for measurements and gained experience with measurements our next step has been to combine measurements into different analysis. Combining measurements in analysis improve decision basis and provide valuable input to which process areas to improve.

Looking back and evaluating the lessons learned the main conclusion is that **working with measurements is a dynamic process**. Developing a measurement program is not a task that ends. The measurements have to be adjusted as processes mature. The measurement program has to mature, and the same goes for the underlying data, the analysis of the data, the presentation of the measurements and the organisational processes using the implemented measurements.

The main lessons learned summarised above will be further elaborated in the rest of this article.

2 Background

The recommendations from external (SPI) consultants are clear from the first meeting: "*Start measuring…"*. In the beginning it seems a bit theoretical – discussing process-related measurements, but the arguments get more convincing with: *"Naturally, you need to have before- and after-measurements in order to see the progress and the result of your SPI-initiative"*, and *"people have short memories – in two years time they can't remember how it was to work in an immature organisation. You have to remind them with your measurements, otherwise they will keep questioning SPI".*

These arguments seem sensible and logic, and they are supported by SPI textbooks, where it is evident that measurements are a central point in software process improvements.

Internally within the organisation you will also meet challenging questions. Especially management tend to ask questions like *"what is the return on investment on SPI?"* and *"when can we see the (financial) benefit of this initiative?".* Not surprisingly, because managers are used to focus on a number of key figures when evaluating success or failure of businesses, projects etc. However, due to the individuality and creativity embedded in software development, universal standard key figures have been difficult to establish within the software industry. Rather, there are lots of recommendations and best practices dependent on your business, the industry you are in, and the models or processes you have chosen to follow.

That SPI is common sense is not enough – you need to prove the real value to your organisation, and the challenge remains: You have to find the right mix of measurements to support the organisation's improvements and make the progress visible to everyone. This is a dynamic process and as the organisation develops, the measurement program has to be adjusted and matured. As new goals for maturity or for the organisation's processes are defined, there will also be a need for new and often more sophisticated measurements. Furthermore using the measurements (to get a status, as basis for decisions etc.) will also generate new requirements for the measurements.

3 Getting started with Measurements

3.1 Data Registration – the Foundation for Measurements

In an immature organisation measurements or statistics are carried out on an ad hoc basis based on the immediate needs in the organisation. Some level of management reporting might be in place and used as basis for decisions. The data quality is most likely questionable. It is true that *"You can't measure chaos - Measuring the performance of a chaotic process will lead to inconsistent results. Organisations which start their improvement from a chaotic level do not have a baseline against which to compare the process improvements"* [Zahran 1998].

In SimCorp the initial basis for measurements were established back in 1995/96 when our workflow system 'SACA' was implemented.

The main purpose was:

- Registration of requests for new functionality
- Registration of errors
- Time registration (registration of working hours)

By registering the requests and the time spend on various activities we wanted to improve the planning processes, thereby improving the control of our software development. Slowly a culture for registering work flow related data arose. A culture that also today strongly exists, and is a very important pre-requisite for our software development processes and related measurements.

Data was registered in order to support the processes, and only to a limited extent with the purpose to measure process performance or to create measurements as basis for decision-making. At this time there was no thorough analysis of how to use all the registered data.

Along with the maturing of processes, increasing demands for quality, organisational and economical growth, additional registrations of data were added (note that this is before initiating SPI). The registrations were closely integrated with the development and test processes.

Example: In order to track changes the developers had to relate their code changes to an incident*. This was enforced in the development environment, so changes could not be saved without registering an incident number.

*An incident is an entity in SACA (our work flow system). A unique identification of either a request or an error including information like customer, categorisation, priority, owner, task flow, estimates etc.

All these registrations of data made a sound foundation for measurements. We developed a culture for registration of data and we learned by experience. This definitely benefited later measurement initiatives.

3.2 Initial approach – Assessment and Metrics Work Group

The SPI program was initiated in SimCorp in 1999 and this implied working with process improvements in a structured way. CMM was chosen as a framework for the SPI activities. In some way this was also the starting point for working with measurements, although it was not, at that point, that obvious.

3.2.1 Making the GAP explicit

In the beginning of year 2000 the first external evaluation of the software development processes (a CMM pre-assessment) was carried out. From this it was concluded that we had to do something about measurements:

> "Due to the lack of metrics regarding the efficiency of the current software development process, it is not possible to determine the actual return on investment at this time, beyond the industry-wide available documented and published results".
>
> "For this reason, the establishment of metrics and measurement of the current situation is one of the most urgent activities for all Process Activation Teams".
>
> **From the GAP-report, produced 2000**

3.2.2 Metrics Work Group

The GAP was documented and the need was evident: We had to start working with measurements in a structured way. In parallel with other SPI initiatives (moving from CMM level 1 to level 2) a Metrics work group was initiated and anchored in the newly established SPI Group.

One statement from the work group was made clear from the beginning: "*….it is fundamental for the whole measurement effort (Metrics Program) that our intention is focused entirely upon the process and the product - management information - and not upon the individual consultant." (from SPI Newsletter, August 2000).*

We chose to use the GQM method (Goals-Questions-Metrics) [Grady 1997] and started working with the process areas one by one, to define measurements giving indications about the processes' performance. A method was defined for the process improvement teams describing how – as part of defining the process – the measurements to monitor the process should also be defined.

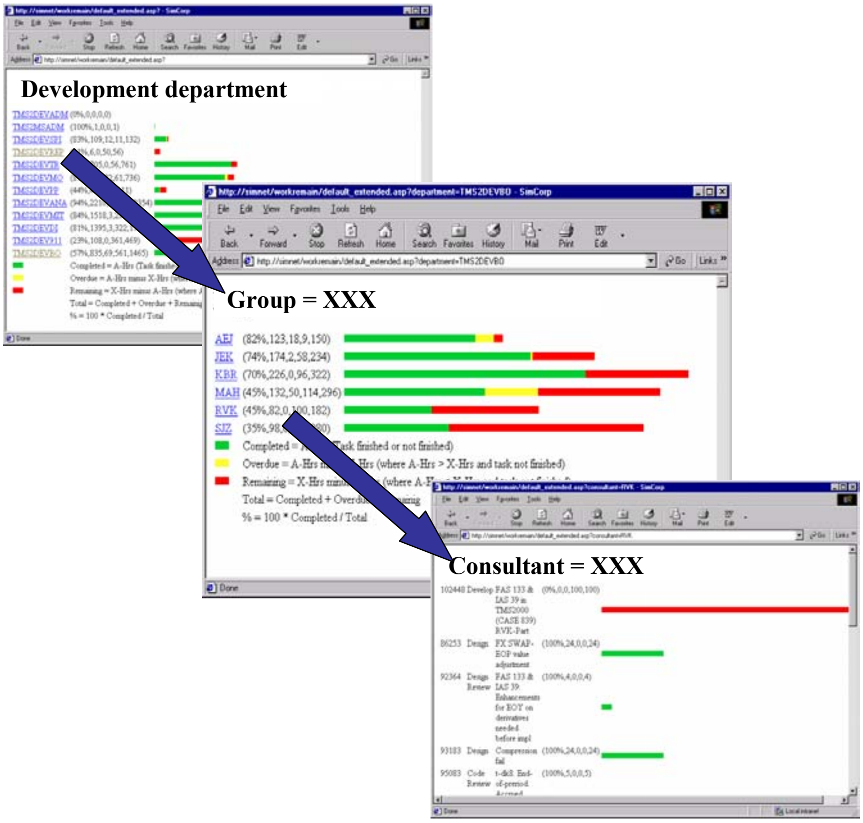
The Metrics work group was not a definite success and after approx. 4 months the work was stopped. The main lesson learned was that our primary focus was on process related measurements which by nature are difficult to work with – "the performance of processes" is not an easy implementable term. Working with goals and related measurements for process areas seemed to be too abstract at that time where the processes were not matured. We forgot to sufficiently align and integrate the measurements with the implemented processes, and the work ended up being too theoretical.

3.3 Collect input to Measurements from the Organisation

Although the Metrics work group was not a success, measurements did however arise from process areas, where process improvements were well on the way. The planning and tracking areas were still in focus and as the processes matured, the need for measuring progress, what time was spend on, observance of deadlines, numbers of requests, errors etc. also got stronger.

At that point we succeeded in implementing some project- and product-related measurements that were successfully adopted by the organisation.

Example 1: Project related measurement: A tool called "Baseline Progress" (Nov. 2000) enabling every developer and manager to follow the progress of the plans on department, group/project and individual level.



http://simnet/workremain/default_extended.asp? - SimCorp

Address: http://simnet/workremain/default_extended.asp?

Development department

TMS2DEVADM (0%,0,0,0,0)
TMS2MSADM (100%,1,0,0,1)
TMS2DEVSPI (83%,109,12,11,132)
TMS2DEVREP %,6,0,50,56)
TMS2DEVTR 05,0,56,761)
TMS2DEVMO 61,736)
TMS2DEVPP (44%
TMS2DEVANA (94%,221 354)
TMS2DEVMIT (84%,1518,3,2
TMS2DEVDI (81%,1395,3,322,1
TMS2DEV911 (23%,108,0,361,469)
TMS2DEVBO (57%,835,69,561,1465)
🟩 Completed = A-Hrs (Task finishe
🟨 Overdue = A-Hrs minus X-Hrs (
🟥 Remaining = X-Hrs minus A-Hrs (where A
Total = Completed + Overdue + Remainin
% = 100 * Completed / Total

http://simnet/workremain/default_extended.asp?department=TMS2DEVBO - SimCorp

Address: http://simnet/workremain/default_extended.asp?department=TMS2DEVBO

Group = XXX

AEJ (82%,123,18,9,150)
JEK (74%,174,2,58,234)
KBR (70%,226,0,96,322)
MAH (45%,132,50,114,296)
RVK (45%,82,0,100,182)
SJZ (35%,98, 80)
🟩 Completed = A Task finished or not finished)
🟨 Overdue = A-Hrs m Hrs (where A-Hrs > X-Hrs and task not finished)
🟥 Remaining = X-Hrs minu (where A-H
Total = Completed + Overdu aining
% = 100 * Completed / Total

http://simnet/workremain/default_extended.asp?consultant=RVK - SimCorp

Address: http://simnet/workremain/default_extended.asp?consultant=RVK

Consultant = XXX

102448	Develop	FAS 133 & IAS 39 in TMS2000 (CASE 839) RVK-Part	(0%,0,0,100,100)
86253	Design	FX SWAP-EOP value adjustment	(100%,24,0,0,24)
92364	Design Review	FAS 133 & IAS 39 Enhancements for EOY on derivatives needed before impl	(100%,4,0,0,4)
93183	Design	Compression fail	(100%,24,0,0,24)
95083	Code Review	t-dkS: End-of-period Accrued	(100%,5,0,0,5)

Figure 1: Planning & Tracking tool: Baseline Progress

An overall progress is presented at department level where the coloured bars indicate *Completed*, *Overdue* and *Remaining* work according to the established plans. From department level it is possible to drill down to group/project level and get the project's progress. From project level it is possible to drill down on individual level to view each developer's progress according to the plans. This measurement gives valuable input to make corrective actions during the development period.

Example 2: Product related measurement: Size of system, where 7 parameters were selected and measured for each version of the product to follow the development of the system's size.

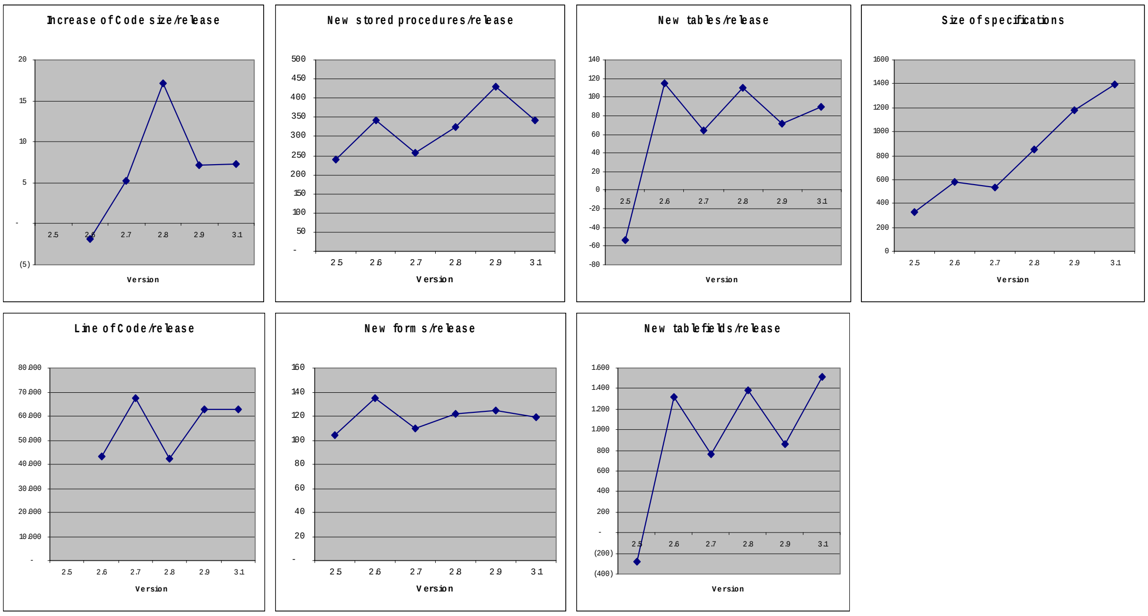


Figure 2: System size figures

The seven graphs show either the increase in the size, e.g. code size, tables, or the total size of the selected parameter, e.g. lines of code, specifications, (y-axis) per version of the system (x-axis).

4 Progressing in Maturity

Aligning the measurements to the maturing processes in the organisation was a lesson learned when initiating the measurement program, and this fact seemed to be strengthened during the following years where the SPI activities also were integrated in the organisation.

In 2001 SimCorp was officially assessed at CMM level 2 recognising that our software development processes had matured. Maturing the organisation also meant maturing the basis for measurements.

Especially two areas had an essential effect on our ability to perform measurements:

- Definition of an integrated and well-structured development lifecycle
- Definition of quality goals

4.1 Development Life cycle

The development life cycle in SimCorp is based on the *Time boxing* principle with fixed **deadlines**, i.e. we deliver a new version of our product, SimCorp Dimension, every sixth month. Also the **resources** allocated for development are relatively fixed due to the complexity of the product and the extensive training necessary for new employees. This leaves the **functionality** released in our product as the primary variable parameter in our software development.

We defined a development lifecycle with identical phases to be executed every sixth month. This structured lifecycle made measurements related to the development processes comparable and enabled continuous measurements.

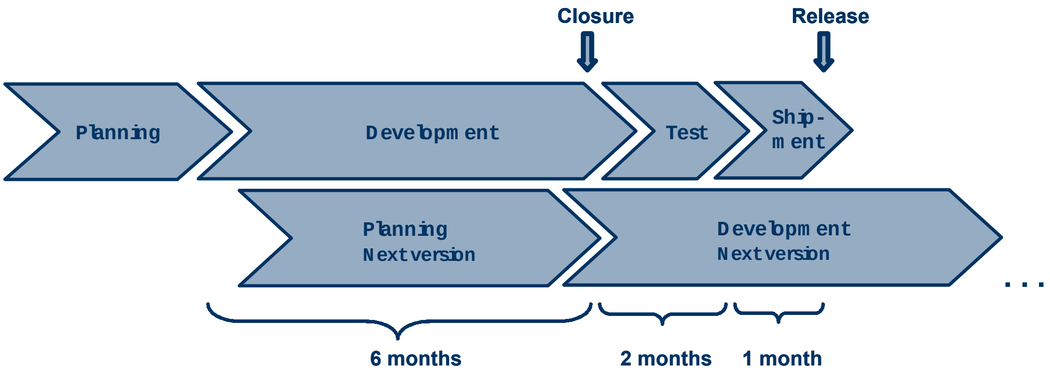


Figure 3: Development Life cycle

The repeatable phases and the amount of historical data collected through the life cycle made comparisons between releases possible. The measurements, e.g. effort spent, number of errors, were summed up per life cycle iteration (version of the product).

Historical data improved the estimations of future activities and the activities in the life cycle were measured to follow progress and degree of completion.

The predictability of our software development plans and processes improved considerably and this also improved our ability to make corrective actions throughout the life cycle.

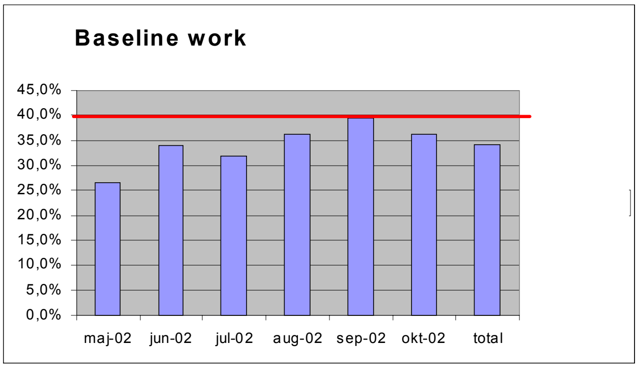
4.2 Quality Goals and Measurements

Focusing on increasing product and process quality we defined a set of Quality parameters and a number of Quality goals. The quality parameters covered both "hard" aspects such as:

- Few defects
- Predictability (example of quality measurement shown below in Figure 4 : "Baseline work", where the percentage of total work spent on Baseline work (work planned for next version of the system) is shown per month.

- and "soft" aspects such as:

- Respected partner (the Development department being respected both internally in SimCorp, i.e. towards the market units and externally towards our customers)
- Employee satisfaction (example of quality measurement shown below in Figure 4 "Resignations", where percentage of resignations is shown per half year)



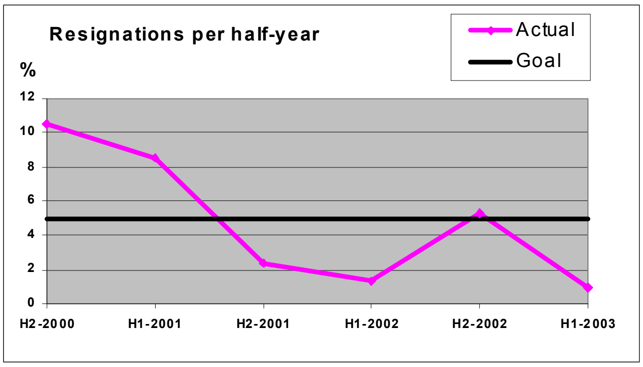


Figure 4: Examples of Quality measurements

The quality goals were institutionalised in the organisation by the SQA function performing regular reporting of the status of the quality goals followed by discussions by management.

4.3 Standardising Measurements at Organisational level

After the level 2 assessment in June 2001 additional effort was put into the measurement program. The foundation was (still) in place based on substantial data registration, and the SPI activities had further enlarged this foundation.

A new Measurement work group initiated their work in the fall 2001 by identifying all the data extractions, statistics, measurements etc. already used in the organisation. A surprisingly large number of measurements were identified supporting the fact that projects developed their own measurements and that measurements were developed on an ad hoc basis to support decisions.

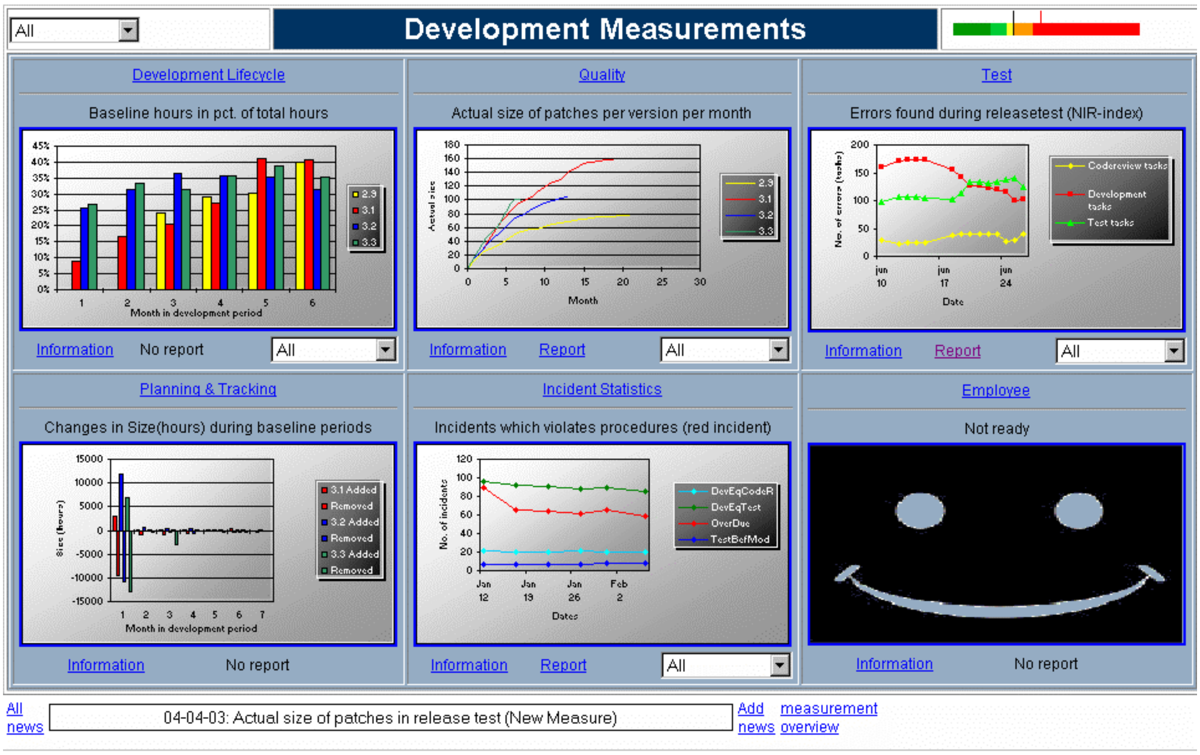
The main purpose of the Measurement work group was to define a set of standard measurements, i.e. selecting or defining the measurements to be integrated in a portal for measurements. The mission was defined as:

> *To develop a clear structure where measurements are easy accessible. The success criteria are to have a solution commonly known in the Development department, containing measurements that give a status of the development activities and product quality, and where measurements relatively easy can be added or removed.*

Main focus in the work group was

- Analyse the existing measurements, adjust them and define how to present them.
- Identify new measurements based on the needs from different process areas.
- Document the measurements to facilitate the implementation in the organisation and in order to ease maintenance.
- Automate the measurements so real time data is presented.

It was very important for the work that the existing measurements already used in the projects were used as basis for the standardised measurements. After 6 months of work and several iterations the first release of “Development Measurements” was ready and released:



04-04-03: Actual size of patches in release test (New Measure)

Figure 5: "The portal: Development Measurements"

“Development Measurements” is a portal with 6 categories for measurements. You can enter each category (e.g. Development Lifecycle) and see more measurements related to the category. The Information link gives you documentation of the measurements and access to the data in an Excel sheet and the Report link gives the possibility to drill down in the data. In the combo box it is possible to make selections (e.g. only data for project X). The measurements are updated every night in a batch job.

4.4 Implementation of the Measurement program

The next step was to implement “Development Measurements” in the organisation. It is important to carefully plan the introduction of the Measurement program, e.g. consider the following questions:

- What is the purpose, i.e. what are the measurements used for.
- What is expected by the practitioners and management based on the measurements.

It is also important to define how to follow up on the measurements. If possible the follow up should be related to the existing processes instead of defining a whole new set of processes that also have to be implemented. E.g.:

- The SQA function uses some of the measurements as part of their regular reporting already institutionalised in the organisation.
- Selected measurements are used at regular management meetings in order to evaluate progress, quality and status of development and test activities.

A measurements program will easily “sand up“, if no consequences or actions are defined as a result. Either you can set absolute goals (e.g. we want to reach 10% by 2005) or set relative goals (the trend should increase otherwise we will react). Please refer to Figure 4, where absolute goals are visible as part of the measurements.

We met the following challenges when implementing the Measurements program:

- Challenge of replacing existing measurements that are well institutionalised but run manually or adjusted to fit project specific needs.
- Getting acceptance of the selected measurements. A measurement or the underlying data will always be questioned when trying to explain why a negative trend is appearing. Thorough documentation will help this, but training in understanding the measurements and the underlying data might be required.
- The measurement – dilemma: Measurements have to be simple to be explained and communicated – but complex to represent real life/activities and dynamic in order to be adjusted to changes (that again needs to be explained and communicated…).

4.5 Analysis – taking Measurements a Step further

Having established various measurements on error levels, effort levels, progress etc., we have started to experiment with combining measurements into analyses. At this point process related measurements have started to make sense.

Example 1: Measuring the effect of quality assurance activities (such as effort on code review and module test) on the product quality (reverse error density):

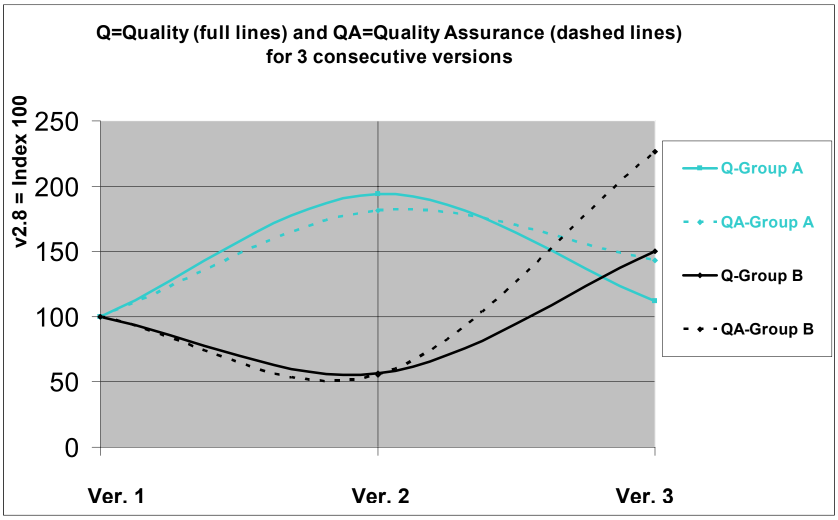


Figure 6: Example of analysis of the effect of quality assurance activities on product quality

The graph shows the correlation between quality assurance activities (dashed lines) and product quality (full line) for 2 selected development projects during 3 development periods (3 versions of the product). We found a high correlation between the effort spend on quality assurance activities and the product quality. Both increasing and decreasing levels of effort spend on quality assurance activities had a strong positive correlation to final quality of the product. This measurement was used to emphasise the effect of performing the quality assurance activities during the development phase.

Example 2: Application of system size. Various measurements of system size have been utilised for a number of analyses:

- Documentation of increasing quality (decreasing error levels in spite of increasing system size and increasing exposure, presented in Figure 7)
- Workload trends for non-coding groups (e.g. Test and Documentation)
- Early warning of possible approaching system limits

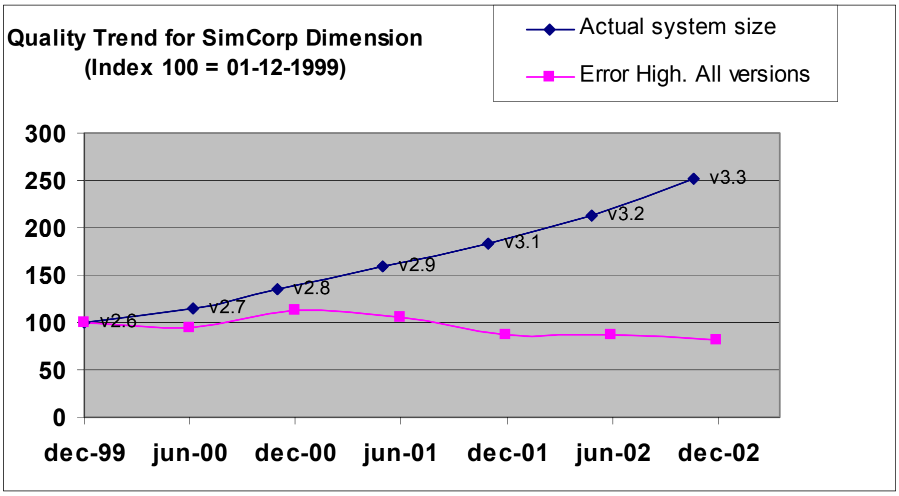


Figure 7: Example of application of system size documenting increasing quality

This measurement was used towards our customers supporting the fact that we were able to deliver quality even though our product had grown considerable in size and complexity.

We have also had very positive experiences supporting the value of the measurement program, when the SQA function uses the measurements to give early warnings or to raise red flags meaning that we have quality issues to be dealt with. During the activities in the development lifecycle SQA monitors the quality by monitoring a selected number of measurements, e.g.:

- Total number of errors
- Number of outstanding errors per developer and tester
- Stability of the system, e.g. patch levels in test version of the system
- Progress of test, e.g. actual effort compared to estimated effort

The issues are presented to management and used as basis for decisions, e.g. re-allocation of people, postponement of a release.

Besides from improving the basis for decisions these analyses provide valuable input to which processes we need to improve.

5 *Next steps*

The initiatives in the Measurement and Analysis area have taken us a long way and we have benefited from the work and gathered a lot of experiences. There is however still a lot of work to be done. To get input to future activities and to evaluate our current abilities in the area we carried out a Focused assessment in co-operation with Delta of the process area: Measurement & Analysis (from CMMI). The purpose of the Focused assessment was:

- To assess our current maturity status in the area of “Measurement and Analysis”
- To establish a GAP
- To pilot the focused assessment concept

The main findings from the focused assessment are reported below:

<table>
<tr><td colspan="2"><b>General observations</b>
• A high accept and culture for gathering data.
• Well established tool for managing measurements (gathering, storing and analysing data and presenting results).</td></tr>
<tr><td><b>Strong points</b></td><td><b>Weak points</b></td></tr>
<tr><td>• The existing SQA policy mirrors well to the measurement program.
• Data collection and storage procedures are well established and maintained.
• Selected data are gathered as specified.
• Analysis and interpretation of measurements.
• The development and maintenance of the measurement program is managed as a project with releases, incidents, etc.
• Assigned responsibility, allocated resources and trained people.</td><td>• General measurement objectives are only weakly stated.
• Variation in the specification of measurements and analysis results.
• Missing overall policy for measurements.
• No audit of the process.</td></tr>
</table>

The next step is to turn these findings into an action plan for the Measurement and Analysis process area.

The next period will be dedicated to further mature the measurement program and focus on the implementation in the organisation i.e. the processes using the various measurements.

6 Concluding remarks

This case story described the experiences in the Measurements & Analysis area at SimCorp A/S.

As described measurements are a vital part of improving the software development processes, thus the companies working with software process improvements (SPI) must also work with implementation of a measurement program.

Different approaches to measurements can be chosen but as our experiences show the ability to perform measurements are very dependent on the maturity of processes in the organisation.

Through the years our work with measurements have provided valuable input to our SPI activities.

We expect that measurements will play a more and more significant role regarding our software development, both as basis for decisions and as providers of input to which process areas to improve.

Literature

[Grady 1997] “Successful Software Process Improvement”, Robert B. Grady, Prentice Hall 1997

[Zahran 1998] “Software Process Improvement, Practical Guidelines for Business Success”, Sami Zahran, Addison-Wesley 1998

7 Author CVs

Malene M. Krohn

1998	Master in Business Administration and Computer Science, Copenhagen Business School
1998 – 1999	System Developer Consultant, SimCorp A/S
1999 –	SPI consultant, heading up the SPI group in the IMS Development department in SimCorp A/S

Learning as Software Process Improvement - An Improvement Program in a Medium-Sized Company

Tor Stålhane[1], Torgeir Dingsøyr[2]
[1]Norwegian University of Science and Technology, [2]SINTEF Telecom and Informatics

Abstract

This paper describes an improvement program in a medium-sized satellite software company, focusing on increasing learning. We conducted interviews and Post Mortem Analysis to capture experience and made a first version of a knowledge repository. We describe our experience with this approach, and reason on the limited success of the knowledge repository so far.

Keywords

Post Mortem Analysis, Knowledge Management, Experience Elicitation, Knowledge Repository.

1 Introduction

Many companies have been engaged in software process improvement projects, either focusing on standardising work practises or “improving” software development by innovative technology. There has recently been a lot of emphasis on improving software development through sharing experience and knowledge within a company, see for example [1, 2]. In this paper, we describe activities and results from an improvement program in a medium-sized company.

Kongsberg Spacetec AS is one of the leading producers of receiving stations for data from meteorological and Earth observation satellites. Through the PROFIT and SPIKE process improvement projects, we have been cooperating with Kongsberg Spacetec in an improvement initiative. The overall goal is to improve the precision of estimations by reusing experiences from earlier projects.

In order to increase experience sharing in the company, we first performed interviews, then several Post Mortem Analyses [3]. Finally, we designed a simple knowledge repository. We will first discuss the “lessons learned” from interviews and Post Mortem Analysis, and then we present the design of the knowledge repository and discuss its success.

But first, let us explain the motivation for using Post Mortem Analysis (PMA). We used PMA as part of the company’s project wrap-up process. The loose form and simple concept is one of the strengths of this method. It gives the company a method that is flexible, easy to learn and apply, while containing a minimum of formalities. In this way everybody in the organisation can participate and not just as information providers. This is crucial in order to support the Total Quality Management philosophy - that quality and quality improvement is everybody’s responsibility; it is not the domain of the quality assurance department alone.

2 Data Collection and Discussion of Results

As part of the improvement work at Spacetec, we conducted interviews with three persons in the company. The focus of these interviews was on identification of important estimation problems. The three persons were a technical manger (TM), a project manger (PM), and a person from the company’s marketing department (Market).

Even though each category has its own worries, some issues were shared by several categories, e.g. scope creep, personnel allocation, software reuse and the use of bid-to-win estimates – see fig. 1. When asked about possible improvements they focused on reuse of previous experience and an open discussion on why estimates fail. As a result of these results, we decided to perform post mortem analyses (PMAs) on three already finished projects.

We organised three PMAs in the company, where the goal was to collect experiences related to project estimation. As can bee seen from the table below and the diagram in Fig 1, some of the experiences collected were also related to other types of problems – e.g. code reviews.

The output from a PMA is a description of the problems in that particular project. Thus, the results show considerable variation. No issue is common for all three projects and the only issues that are common for two projects are problems related to the requirements, personnel allocation and detailed design.

The data shown in Fig. 1 can be organised into three groups of issues – administrative issues, development issues and issues common to development and administration.The KJ process (after Japanese ethnologist Jiro Kawakita) which collects and strucutures data from a group of people is only the first step in collecting experiences in a PMA. The KJ looks at what happened – not at why it happened and not on what we can do about it. This is, however, what we need to know if we want to translate the results from the KJ into reusable experience. For the PMA, this step is taken care of by the root cause analyses, which is documented by using Ishikawa diagrams.

Tab. 1. Issues raised during the PMAs

Issues raised	PMA1	PMA2	PMA3
Requirements related problems – e.g. scope creep		X	X
Personnel allocation	X	X	
Problems related to detailed design	X	X	
Software reuse	X		
Bid to win		X	
Project start-up phase		X	
Problems related to testing			X
Problems due to time pressure			X
Missing or bad code reviews			X
Unnecessary, time consuming meeting			X
Problems related to system's integration			X

Most of the results of the root cause analysis for our company turned out to be experiences such as "If we work more on the requirements before the project starts, we will get less requirement changes later". Only a few were concrete, like "We need a new template for activity X" or "Check that job Y is done".

When we consider the large variety in issues raised, there are some important conclusions that can be drawn:

- Each project is in some sense unique both in what should be developed and in the experiences that can be harvested.
- Over time, our methods and the environment will change. Collecting experience must therefore be a continuous activity.
- Given the variety of results, it is important that all roles in a company are given the opportunity to contribute their experiences.
- Making experience available for reuse is not something that will just happen. We need to have a separate activity where the experiences are collected, organised and made available for reuse.

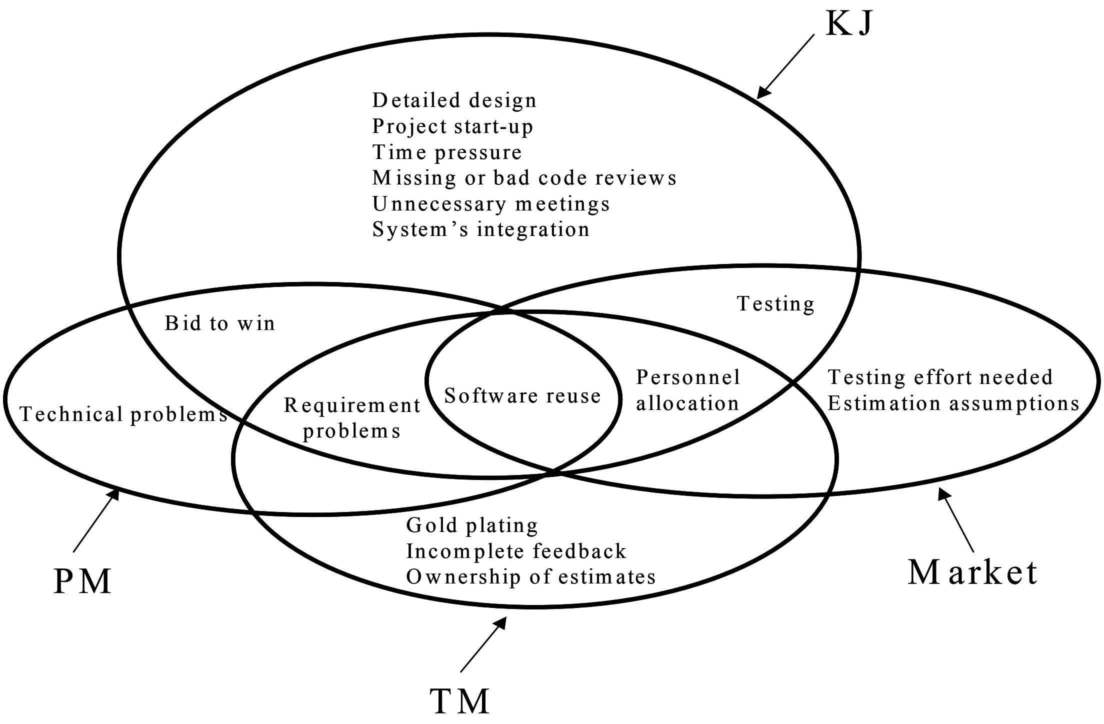


Fig. 1. Where was each problem identified?

The KJ circle contains lessons learned from the Post Mortem Analysis, the PM, TM, and Market issues were found in interviews with a Project Manager, Technical Manager and one person from the Market department.

Since the PMAs were done without management participation, we will consider these results to be the voice of the developers. The items not mentioned by the developers are mostly concerned with project management, such as:

- Gold plating
- Estimators do not want to discuss their estimates, for instance the assumptions made
- Testing and project management are underestimated
- Incomplete feedback from developers on task status

The developers and management – inclusive market - seem to have rather different views on the estimation problem.

- Developers – why don't we meet the current estimate? The estimate is considered to be a fixed value.
- Management – why did we make the wrong estimate? The focus is on the underlying assumptions and status control.

Both of these views are important and help us to highlight the dual problem of project cost estimation – getting a reasonable estimate and keeping the project under control in order to keep within this estimate. It also highlights the fact that when the assumptions are no longer valid, neither is the cost estimate.

3 PMA – the company on its own

After the researchers had conducted three PMAs, the company decided to perform PMAs themselves. So far, they have done one PMA on a project that lasted for nine months, and required approximately

4.000 person-hours of work.

The following positive experience was found from this project: The participants learned about a new domain, there was a good working environment in the group, the customer was satisfied and the project was in general carried out well. Negative experience was the large workload, lack of knowledge of the product and the customer, poor pre-contract preparation and also poor status control during the project.

In addition to identifying these positive and negative experiences, the company also did a root-cause analysis, and found corrective measures for some of the main causes. For example, one of the causes of the “large workload” was found to be “lacking design/analysis”, and corrective actions were defined as “better separation between pure deliverance and design projects”, “use more time for design and analysis”, and “do work estimations more often”. Some of these problems were also identified in the previous PMA – e.g. poor status control and the need for re-estimation when the underlying assumptions are no longer valid. This illustrates two things:

- Management experience should be included in the experiences from a project.
- If the repository is not in place, we will end up doing the same mistakes and collecting the same information again and again, without getting any process improvement.

4 Learning by Experience – Designing a Knowledge Repository

Spacetec’s goal was that ”all relevant, reusable experience should be retrievable for projects in the simplest possible way, which is best done through a knowledge repository available on the company Intranet”. The company Intranet already had a project web containing an overview of all projects that are running and completed (including a project” cover page” with key information about the project). Further, the Intranet contained web pages with the quality system, acronym lists, maintenance documentation, a “frequently asked questions” about executive work, and also mailing lists for projects and a searchable repository of project mail.”

In order to design a knowledge repository, we gathered five managers and developers in a workshop consisting of four parts:

1. Finding types of experience that will be useful in a repository on a long-term basis. We used the KJ technique described in [3].

2. Finding experience that is useful on a short-term basis, again using the KJ technique. We also brainstormed to identify factors that make experience from one project relevant for other projects.

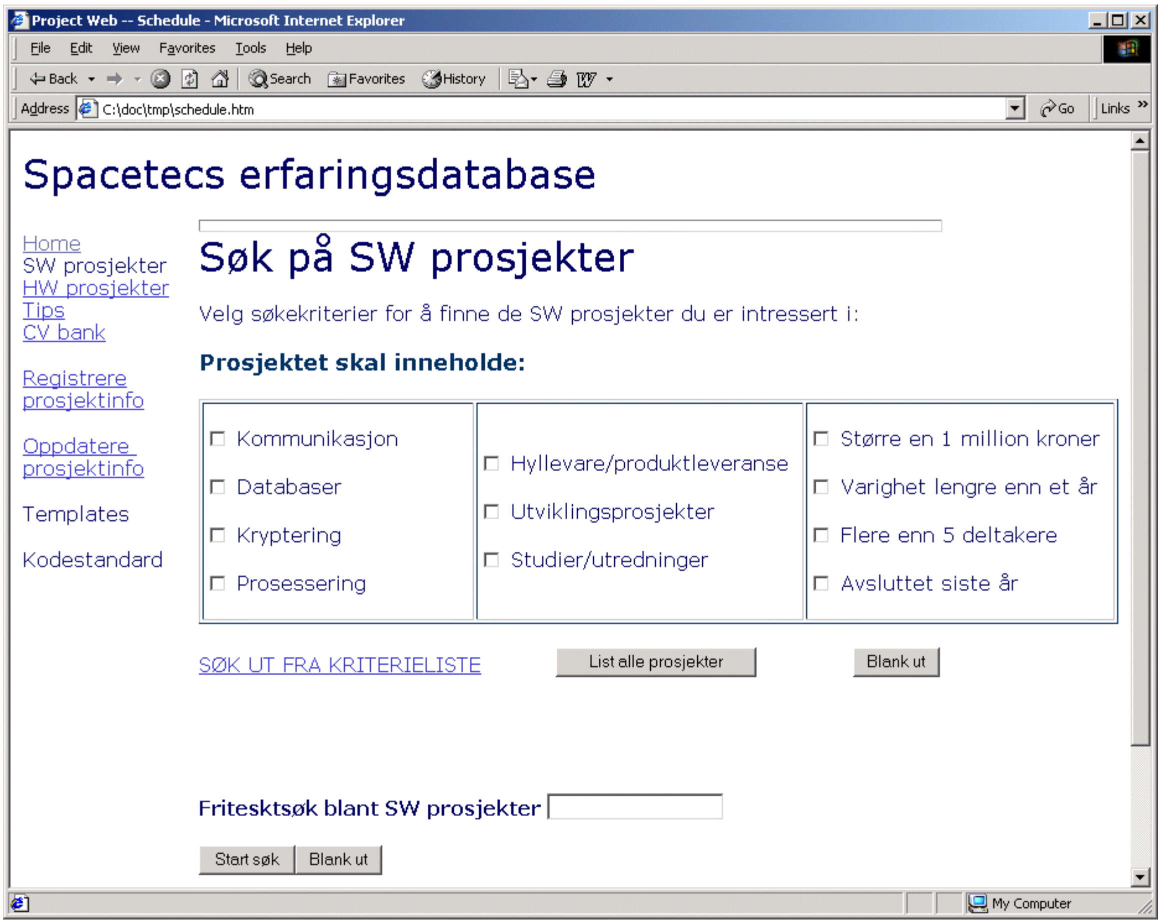
3. Designing the user interface – a group exercise used to design possible user interfaces for the knowledge repository.

4. Brainstorm on how to gather the information – based on the experience that will be useful and the design for the user interface.

The conclusion of the workshop was that the repository should have:

- A front page with the possibility to search for software projects, grouped according to project type.
- Hits should be shown with descriptions, project number, and keywords for the project. This page should also contain links to acronyms that are used.
- A detailed display of each project should contain a short description, a project summary with start and stop information, number of participants and price, a link to the project web, a statement of work, project management plan, final report, results from post-mortem reviews, and the project cover.

Based on these requirements, a prototype mock up knowledge repository was created, and tested on the workshop participants. This test clarified some of the requirements for the knowledge repository. The further development of the repository was to be done by the person in the company who was responsible for the company Intranet.


Project Web -- Schedule - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address C:\doc\tmp\schedule.htm
Go Links
My Computer

Spacetecs erfaringsdatabase

Home
SW prosjekter
HW prosjekter
Tips
CV bank

Registrere prosjektinfo

Oppdatere prosjektinfo

Templates

Kodestandard

Søk på SW prosjekter

Velg søkekriterier for å finne de SW prosjekter du er intressert i:

Prosjektet skal inneholde:

□ Kommunikasjon
□ Databaser
□ Kryptering
□ Prosessering

□ Hyllevare/produktleveranse
□ Utviklingsprosjekter
□ Studier/utredninger

□ Større en 1 million kroner
□ Varighet lengre enn et år
□ Flere enn 5 deltakere
□ Avsluttet siste år

SØK UT FRA KRITERIELISTE

Friteksøk blant SW prosjekter

Fig. 2. A prototype mock up of the knowledge repository – where you can search for projects containing "communication", "databases", "cryptography" or "processing", after project type or project size.

The mock-up contains an interface for search for projects where you can select search for projects that contain communication, databases, cryptography, processing, product release, development project, studies, projects larger than 1 million NOK, projects lasting more than a year, with more than five developers, completed last year.

The knowledge repository has not yet been implemented fully, even though it is now eighteen months since we organised the workshop. The company assigned a person to implement the knowledge repository, who has been overloaded with other work in the company, and says that this is the reason why it has taken such a long time. However, we do not think implementing the knowledge repository as we designed it would require many man-hours of work for someone who is used to designing Intranet solutions. It might be that the repository has not been implemented because there has not been a sufficient demand from managment or from the main users. It might also have been that the development would have been done faster if the person responsible for developing the repository had been more involved in the design, and also in the improvement project in the company.

Another option is to get someone from outside the company to develop the repository. We did not see this as a good solution, because we think the maintenance competence should be kept within the company. Also, we have found [4] that these kinds of tools require a “champion” within the company to promote the usage of it – that would not be present if it was delivered form elsewhere.

There is, however, a possibility that the knowledge repository will be implemented as a part of another improvement initiative. The company is working on documenting the main existing processes, and making process descriptions available in a process guide on the Intranet. A knowledge repository is then seen as an ideal way to communicate additional feedback that are relevant to process descriptions, but are not suggestions for changes of the description. That is, material that gives users a more detailed understanding of certain processes. This work is still ongoing.

5 Some conclusions

From performing interviews, PMAs and designing a simple knowledge repository in the satellite software company, we have learned that:

- It was a good idea to “kick start” experience sharing with PMAs, which meant that the company did not have to wait for the knowledge repository implementation.
- Running PMAs without any form of experience repository – supported by a computer or not – will remove some of the advantages with PMAs.
- The company has involved many people in designing the knowledge repository, which shows that the company is focusing on experience transfer (something people in the company have requested).
- The implementation of the knowledge repository depended on a person that was ad-hoc-allocated to the project after initiation. We should have clarified that the implementation job had got resources before initiating the knowledge repository project.
- We still hope that the knowledge repository is implemented, and recent work on documenting processes in an Intranet-based process guide gives the company a new opportunity to do so.

6 Acknowledgement

We are deeply grateful to Nils Brede Moe at SINTEF Telecom and Informatics for participating in collecting the data for this paper, and in designing the mock-up knowledge repository. We are also grateful to our contact persons at Kongsberg Spacetec.

7 Literature

[1] Torgeir Dingsøyr, Emil Røyrvik, "An Empirical Study of an Informal Knowledge Repository in a Medium-Sized Software Consulting Company," 25th International Conference on Software Engineering (ICSE). Portland, Oregon, 2003.

[2] Kurt Schneider, Jan-Peter Von Hunnius, "Effective Experience Repositories for Software Engineering," 25th International Conference on Software Engineering (ICSE). Portland, Oregon, 2003.

[3] Andreas Birk, Torgeir Dingsøyr, and Tor Stålhane, “Postmortem: Never leave a project without it,” IEEE Software, special issue on knowledge management in software engineering, vol. 19, pp. 43 - 45, 2002.

[4] Torgeir Dingsøyr, “Knowledge Management in Medium-Sized Software Consulting Companies,” doctoral thesis, Department of Computer and Information Science, Norwegian University of Science and Technology, Trondheim, 2002, pp. 206, ISBN 82-7477-107-9.

8 Author CVs

Tor Stålhane

was born in 1944 and became a M.Sc. at the Norwegian Institute of Technology, University of Trondheim (NTNU) in 1969. During 1969 to 1985 he worked at SINTEF - RUNIT, department for languages and compilers. From 1985 he worked on his Ph.D. studies and finished his thesis on software reliability in 1988. From 1988 he was back at SINTEF where he mainly worked with quality assurance and software safety and reliability. In 1997 he became professor in Computer Science at the Stavanger Polytechnic. In 2000 he became professor at the Norwegian University of Technology and Science in Trondheim where he today works full-time. During the latest decade he has been mainly been working with safety analyses of software intensive systems and measurement based process improvement.

Torgeir Dingsøyr

is a research scientist at the Department of Computer Science at SINTEF Trondheim, Norway. He wrote his doctoral thesis on “Knowledge Management in Medium-Sized Software Consulting Companies” at the Department of Computer and Information Science, Norwegian University of Science and Technology. He has published papers on knowledge management in software engineering, case-based reasoning and on software engineering education. In 2001, he spent half a year as a guest researcher at the Fraunhofer Institute for Experimental Software Engineering in Kaiserslautern, Germany.

Questionnaire-based Process Evaluation in NOVE-IT

Daniel Keller, Ann Cass, Philipp Sutter

Abstract

The goal of the 150 million € project NOVE-IT [NIT] of the Swiss Federal Strategy Unit for Information Technology (FSUIT) is to standardise the provision of IT services across the 7 departments of the Swiss central government over approximately 5 years. A process model for IT procurement, operation, and service provision (also called NOVE-IT) was developed in the first phases of the project. The defined processes are being implemented by more than 70 governmental agencies of the Swiss Federal Government.

Within the Swiss Federal government, process responsibility across organisational borders is new and unfamiliar. Experiences gained so far show that the cultural change from working in a functional way to process-orientation is the most difficult challenge of the programme.

The performance of process assessments by external parties is a key part of the strategy of independent quality assurance that has been established for the NOVE-IT project. A SPiCE-like assessment method has been developed and 4 process assessments of federal organisations have been performed to date. The benefits of these process assessments have been undeniable but the assessments are viewed as costly as well.

A less costly form of process evaluation has been subsequently developed to provide an overview of process implementation across all the federal departments. This method, called Process Assessment Light, is based on the completion of questionnaires by organisational staff in a one-day session moderated by two external assessors.

A first trial of Process Assessment Light has been conducted in June 2003. The full evaluation of all processes in all departments of the Federal Government has been conducted at the end of September 2003.

Keywords

NOVE-IT, questionnaire-based process evaluation, ISO 15504 process assessment, SPiCE, quality assurance, IT processes

1 A Composite of Processes in NOVE-IT

1.1 NOVE-IT Programme Overview

The goal of the 150 million € project NOVE-IT [NIT] of the Swiss Federal Strategy Unit for Information Technology (FSUIT) is to standardise the provision of IT services across the 7 departments of the Swiss central government over approximately 5 years. A process model for IT procurement, operation, and service provision (also called NOVE-IT) was developed in the first phases of the project. The defined processes are being implemented by more than 70 governmental agencies of the Swiss Federal Government.

This IT process model serves as a framework of guidelines for the homogenous but decentralized implementation and the provision of standardized tools. The process documentation is structured uniformly and simply. It is typically produced using normal office tools and published in the Intranet using a Lotus Notes application. Experiences gained so far show that the cultural change from working in a functional way to process-orientation is the most difficult challenge of the programme.

Through long years the Swiss Federal Government had become a traditional, strongly functional and hierarchical organisation. The management culture was shaped by federalism and a relatively high independence of departments and organisational units. Working together in the sense of a united enterprise named “government” is still uncommon. Process responsibility across organisational borders is new and unfamiliar. The performance of process assessments therefore has to overcome big resistances. Experience shows that without the assignment and reporting attached to the top-level management it is not possible to introduce and assess new processes successfully on an enterprise-wide scale.

1.2 Process Comprehension and Modelling

Process concepts and the process methodology have been developed during the programme NOVE-IT in a pragmatic way. The following concepts and best practices have formed the methodology used in NOVE-IT:

- Osterloh’s [OSTR00] process concepts provided a common understanding in the beginning. Osterloh’s “Triage” idea was used to formulate “scenarios” in several processes.
- Process responsibility was defined following Becker [BCK02] and Schmelzer [SCHM01] and by applying the expert knowledge of external consultants working in the NOVE-IT programme.
- The IT process Model of IBM [ITPM] was used as the main reference model for the infrastructure and support processes. Principles and best practices from the IT Infrastructure Library [ITIL] had an influence on the service management processes.
- An off-the-shelf Lotus Notes application was used and slightly modified for the process documentation.

For the modelling of the processes, common office tools of the government were used from the beginning. The process groups worked simultaneously on the definitions of their processes, reconciling their results via e-mail. This resulted in a rapid growth of the documents and their versions which proved difficult to control. Improvements were realised by using the Lotus Notes application for the management of the process documentation.

To verify the consistency of the process interfaces, a process model (base model) in IDEFØ-Notation [IDEF] was developed in addition to the process definition using standard Office tools. This base model is available to interested parties, however it is only used to ensure consistency.

1.3 Process Map

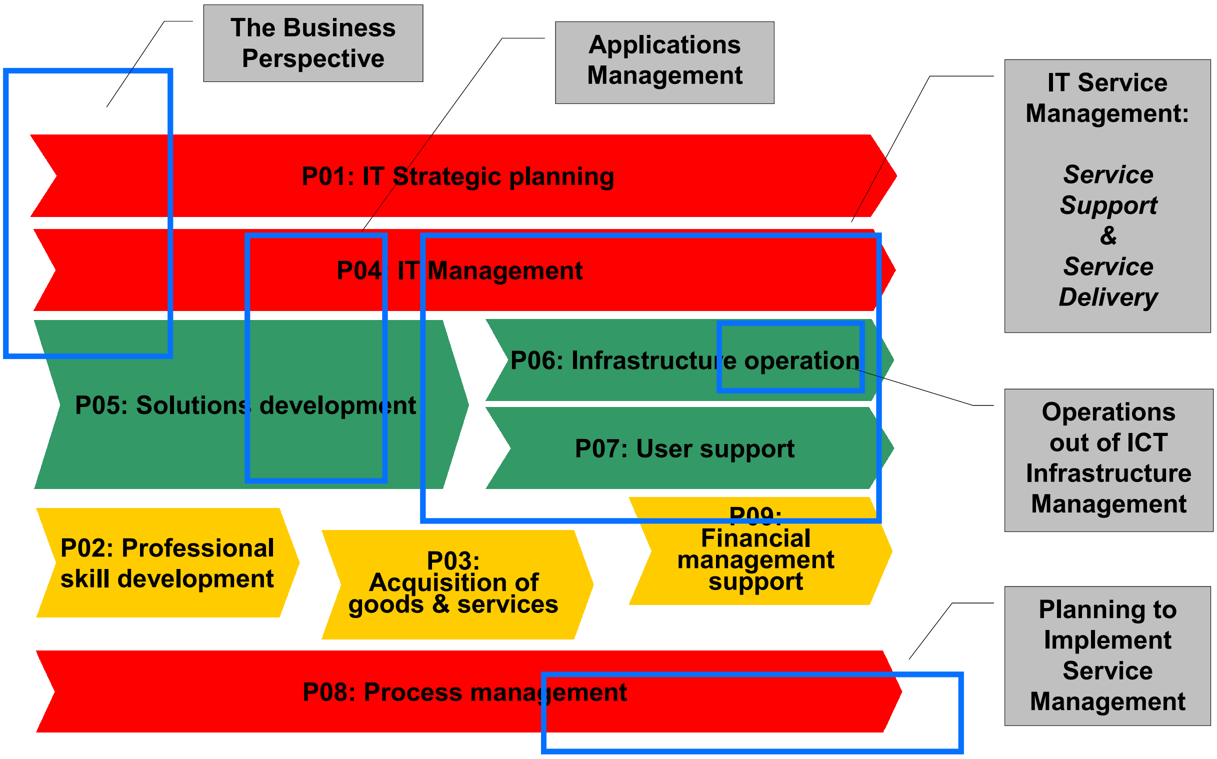

The Business Perspective
Applications Management
IT Service Management:
Service Support & Service Delivery
P01: IT Strategic planning
P04: IT Management
P05: Solutions development
P06: Infrastructure operation
P07: User support
Operations out of ICT Infrastructure Management
P02: Professional skill development
P03: Acquisition of goods & services
P09: Financial management support
Planning to Implement Service Management
P08: Process management

Figure 1: Process Map and visual comparison with ITIL

IT service consumption and IT service provision in the Swiss Government are defined through 9 IT processes (see Figure 1 and Figure 6 (detailed)). They link customers (service consumers), internal service providers and external suppliers into common activity chains. The performance of the core processes (P05, P06, P07) creates the added value for the service provider. The support processes (P02, P03, P09) are necessary to execute the core processes. The management processes (P01, P04, P08) define guidelines, co-ordinate and secure the process and product quality. The service flows in bold (see Figure 6) are the most important ones from the point of view of the service consumer.

A coarse comparison of the NOVE-IT processes with the current version of the IT Infrastructure Library (ITIL) shows the following differences and similarities (see Figure 1 for the common process areas):

- The NOVE-IT processes are broader in scope than those described by ITIL, e.g. in the areas of strategic planning (including architecture development und standardisation), financial management, and acquisition which are covered by specific NOVE-IT processes.
- ITIL contains only a few clearly defined roles compared to approximately 50 roles in NOVE-IT. Role definitions are a mandatory part of every process definition in NOVE-IT, whereas they are not consistently defined throughout ITIL.
- In some areas ITIL shows detailed task sequences in the form of best practices (e.g. change management), while in other areas it just provides general concepts and guidance (e.g. service level management). NOVE-IT has a documentation structure where task sequences (here called activity lists) consistently appear at level 2 through all processes (see also Figure 2: Structure of Process Documentation).
- The ITIL processes have not been taken over 1:1 but have only been used as a reference for activities and work products. NOVE-IT processes are defined as end-to-end (from producer to consumer) processes, whereas ITIL processes are defined as a network of functional areas.

The NOVE-IT processes have reached the status of a systematic framework [BCK02] in the Swiss Federal administration, which manages the provision of process instruments and the continuous improvement of the processes. The work of the special interest group "IT Process Improvement Network" [ITPIN] of the SwissICT (the Swiss association of IT and telecommunication) has since then lead to the creation of a similarly structured IT process map. ITPIN states as its mission "to advance to a centre of competency of reference in the areas of assessment, design, implementation and improvement of IT processes".

1.4 Process Documentation

The documentation of the IT processes is stored in a Lotus Notes application that is accessible to all users on the Intranet as a "process portal". From the main page, there are links to the homepages of all processes, each structured in the same way and giving a direct access to all relevant process documents.

The process documentation is structured into several logical levels. A normative process core – spanning all departments – is separated from the rest of the documents. The processes are defined at a sufficiently high level so that they apply to all organisations within all federal departments; the documentation of the organisational management system is not included in the processes.

Levels 0 and 1 comprise the process design and definition, which may include the process description, a graphical diagram (one per process and sub process), role descriptions, and process measurements.

The activity lists of level 2 form the execution-oriented part of the process documentation. They serve as simple and easily understandable descriptions of sequences of actions. They define what each principal role has to do to produce the requested result.

Level 3 defines the "How" and is focussed on the details of software solutions and process instruments implemented locally. Therefore, level 3 is available in the central documentation only in small parts, mainly represented through forms and service procedures.

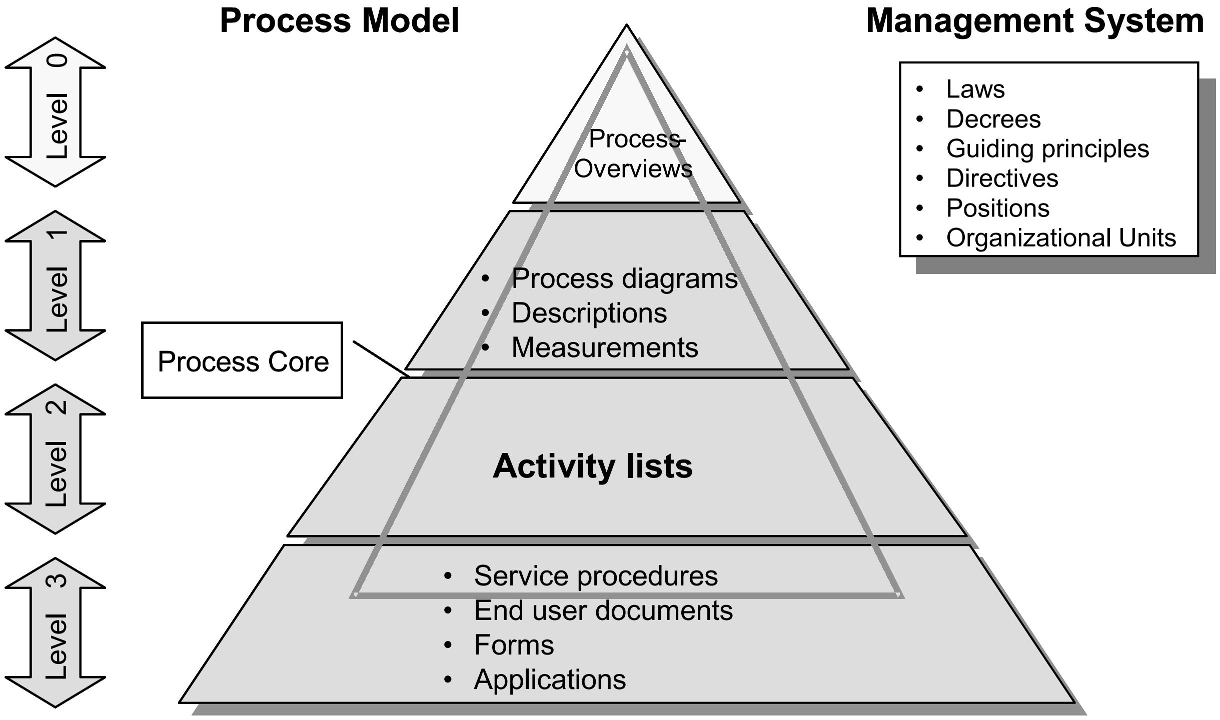


Figure 2: Structure of Process Documentation

1.5 Continuous Process Management

Based on experiences, new requirements or proposals, the processes are continuously improved following the process “Process Maintenance” (P08). P08 comprises of measurements, evaluations and continuous process improvement by the process responsibles.

Responsibility on the federal level for each process is assigned to a so-called “process group” with the process responsibles from all departments. Each department appoints a process responsible per process. A process committee insures inter-process co-ordination.

Improvement suggestions are assessed by the process group, implemented in the process definition and released for departmental introduction in no more than two releases per year. This guarantees a useable balance between stability and change.

The process responsibles bear the responsibility for the correct usage and improvement of the processes. Line management is responsible for the process performance of their employees and supports the process responsibles in the process improvement. Critical success factors for the continuous process management are:

- A good collaboration between the process and line organisations
- Regular feedback through internal and external process assessments

2 Types of Audits and Assessments

Several types of assessments, reviews and audits are performed either as part of NOVE-IT or as part of other organisational activities. To gain acceptance from the organisation they have to complement the existing audits and reviews.

2.1 Audits and Reviews

Audits of financial statements are a longstanding part of the organisational culture. External audits and management reviews are less common. Figure 3 compares the 3 methods.

2.1.1 Internal Audit

Internal audits have their traditional roots in the examination of financial accounts. The bodies in charge have extended their original scope of their investigations into checks of the cost effectiveness and other evaluations. For IT audits the internationally recognized COBIT reference model [COBIT] is used.

2.1.2 External Audit

External audits are also known as third party audits. They provide evidence about the fulfilment of requirements. Well-known examples are the certification of organisational units according to ISO 9000 or ISO 14000.

2.1.3 Management Review

Management reviews stem from the idea of TQM. They are also required by ISO 9001:2000. Management reviews are part of the management culture of those organisational units that commit themselves to excellence following the EFQM [EFQM] model or who have certified their management system according to ISO 9000:2000.

	Internal Audit	**External Audit**	**Management Review**
Rating Model	COBIT reference model and other evaluation standards	Defined requirements	Objectives, metrics, checklists
Size	Several days	Several days	1-4 h
Methods	Audit methods	Audit methods	Review
Roles	Audit team	Audit team	Management team
Results	Report (fulfilment of audit criteria)	Report (fulfilment of audit criteria)	Measures for achievement of objectives
Objectives	Evaluation of effectiveness and efficiency; compliance and legality	Compliance with defined requirements, confidence of stakeholders	Determination of current position, continual improvement; achieving excellence

Figure 3: Review and Audit types

2.2 Process Assessments

Process assessments are a part of the NOVE-IT process P08 “Process Management”. For all assessments detailed procedures have been elaborated. Figure 4 compares the 3 assessment types.

2.2.1 Individual Process Assessment

Individual process assessments (IPAs) provide a means for the line manager to discuss the processes and possible improvements in an open meeting with each employee. IPAs must be clearly separated from employee performance reviews.

2.2.2 Periodic Process Assessment

With the periodic process assessment, the maturity of the process is derived in a structured self-assessment by the process responsibles. The result of the evaluation is a quantitative measurement of process maturity. Through direct involvement of an executive manager, decisions on improvement plans can be made as part of the assessment.

2.2.3 External Process Assessment

The performance of process assessments by external parties is a key part of the strategy of independent quality assurance that has been established for the realisation phase of the NOVE-IT project. SPiCE-like assessments are used as an external means of evaluating both the NOVE-IT processes themselves (as defined in the NOVE-IT process model and supported by various templates and tools) as well as their degree of implementation in the assessed organisational units. The main focus of the assessments is on capability determination. The results however will later on be taken as input into the continuous process improvement cycle that is defined in the NOVE-IT process P08, Process management.

	Individual Process Assessment	**Periodic Process Assessment**	**External Process Assessment**
Rating Model	Open checklist	Checklists, criteria catalogue, process index	ISO TR15504, NOVE-IT process model
Size	Approx. 1h	½ day	3-5 days, 8-10 group interviews
Methods	Interview	Structured self-assessment (review)	Assessment method (group interviews)
Roles	Line manager, individual	Process owner, executive management, process team members	Assessor team (internal or external), process users
Results	Improvement proposals	Index, measures	Capability level, report
Objective	Process improvement	Continuous improvement, fulfilment of middle term objectives, process maturity	Cultural change, determination process maturity, process improvement

Figure 4: Assessment types

2.2.4 Assessment method

During the preparation of the first assessments a custom assessment model was derived from the NOVE-IT process model. The model adopts the structure of the exemplar model in ISO 15504, for example regarding the dimensions of the model, its elements and indicators. It consists of a two-dimensional system of processes and ratings.

The process dimension was taken over more or less unchanged from the NOVE-IT process model. Processes and sub-processes become the main elements of this dimension. Indicators are formed as the equivalent of base practices and work products from similar parts of the NOVE-IT process model. Interfaces between the processes in NOVE-IT for example could be used in many cases to represent work products; sub-sub-processes, lower level activity steps or groups of steps formed the base practices.

The capability dimension was taken over from ISO 15504 unchanged. The model therefore has the capability levels 0 to 5 with the 9 attributes as defined in ISO 15504.

NOVE-IT assessments follow the seven classic steps of Initiation, Planning, Briefing, Data Collection, Process Rating and Reporting. Objective evidence of process performance and capability is collected through interviews with organisational staff and the examination of project documents.

Although originally the project planned to establish the NOVE-IT assessments as compliant with ISO 15504, the complex relationship between the NOVE-IT processes and the ISO 15504 processes in Part Two made establishing an algorithm for converting ratings difficult and time-consuming. As formal conformity as specified in ISO 15504 is not a primary requirement for the project, it was decided to abandon this effort. Therefore although clearly the NOVE-IT assessment method follows the requirements of ISO 15504 "in spirit", no formal proof of conformity was established for the assessment model.

2.2.5 Assessment performance

The performance of external process assessments has proved difficult, because, among other reasons, the effort needed to perform the full evaluation can be high, both on the side of the organisational unit assessed and for the assessment team. Approximately 25 to 30 assessor days per process have been reported in NOVE-IT. The external process assessments can therefore not be used to evaluate the full extent of all processes in all departments in the Swiss Government; instead, external process assessments are mainly used in focussed areas for an in-depth analysis of specifically chosen parts of processes and organisational units.

The large amount of effort required for NOVE-IT assessments may seem surprising at first glance, particularly in comparison with other published results. For example, in a series of ISO 15504 conformant assessments of space software suppliers sponsored by the European Space Agency, an average effort of 4.5 person-days per process has been reported [ESA]. However it must be stressed that these assessments were of individual software development projects. In NOVE-IT, the processes describe organisational activities that in most cases are not project-based. A larger number of interviews with different process roles are necessary to collect sufficient evidence for judging the process performance and capability of an organisation. Also the granularity of what NOVE-IT calls a "process" is 3 to 10 times larger than the typical SPiCE process. NOVE-IT covers a broader area of activities with 9 processes compared to SPiCE with its 40 processes. META Group came to the conclusion that the efforts for NOVE-IT assessments meet or even undercut their benchmark data [ISB-SPICE].

2.3 Benefits and Cost Effectiveness of Process Assessments

The feedback from assessed organisations regarding the benefits and cost effectiveness of external process assessments therefore is mixed. It is however sure that without comparable and standardized assessments the status of implemented processes is not objectively measurable. The performance of assessments that openly evaluate and rate existing processes is a new element in the organisational culture. The effort in relation to the benefits has been specifically analyzed for external process assessments. The FSUIT commissioned the consultancy META Group to conduct a study comparing the experiences of performed NOVE-IT external assessments with benchmark data from other organisations [ISB-SPICE]. The results of this study can be summarized as follows:

- The basic requirements of the FSUIT are fulfilled through a transparent and pragmatic utilization of the standard ISO/IEC TR 15504 (SPiCE).
- The cost-value ratio of IT process assessments is directly defined through the productivity gain reached. There are examples in the IT area where substantial yearly increases in productivity have been reported.
- First-time assessments produce higher costs, however costs can be cut down on follow-up assessments through several measures. Proposals include the build-up of internal assessor know-how and the use of mixed assessment teams.
- The study proposes an evaluation over the full area of all NOVE-IT processes based on a simplified evaluation model. The "Process Assessment Light" has been developed specifically for this task. This first evaluation shall then be followed over a period of 1 to 4 years by follow-up external process assessments.

3 Process Assessment Light (PAL)

3.1 Purpose

The NOVE-IT programme management needs an overview of the current state of all processes as they have been implemented locally in all departments of the Swiss Government. To provide a means that allows for efficient full-scale coverage, the so-called "Process Assessment Light" or PAL has been developed. The PAL consists of two parts:

- A tool, an MS EXCEL-based questionnaire
- A procedure that defines how the PAL is performed

The “Process Assessment Light” is based on this assessment model of the external process assessment. However to make for a quicker evaluation and to keep the effort needed on both the side of the assessment team and the assessed organisation low, the “Process Assessment Light” focuses on the requirements of Capability Level 1 of the assessment model, the process performance as evidenced by its executed base practices and the created work products. In addition to that, the existence of the primary roles as required in the NOVE-IT process model is evaluated. Roles must be allocated to persons who have the necessary skills and time to perform their tasks. In this aspect, the PAL also evaluates practices associated with Capability Level 3 of ISO 15504. Because of the way NOVE-IT processes are defined, there are also some aspects of planning and measurement, which are checked with the PAL.

Some NOVE-IT processes also define so-called scenarios, paths through the process that are typical or that define variants of how the specific process may be instantiated. The existence and use of the scenarios in these cases is also part of the evaluation with a “Process Assessment Light”.

3.2 PAL Forms

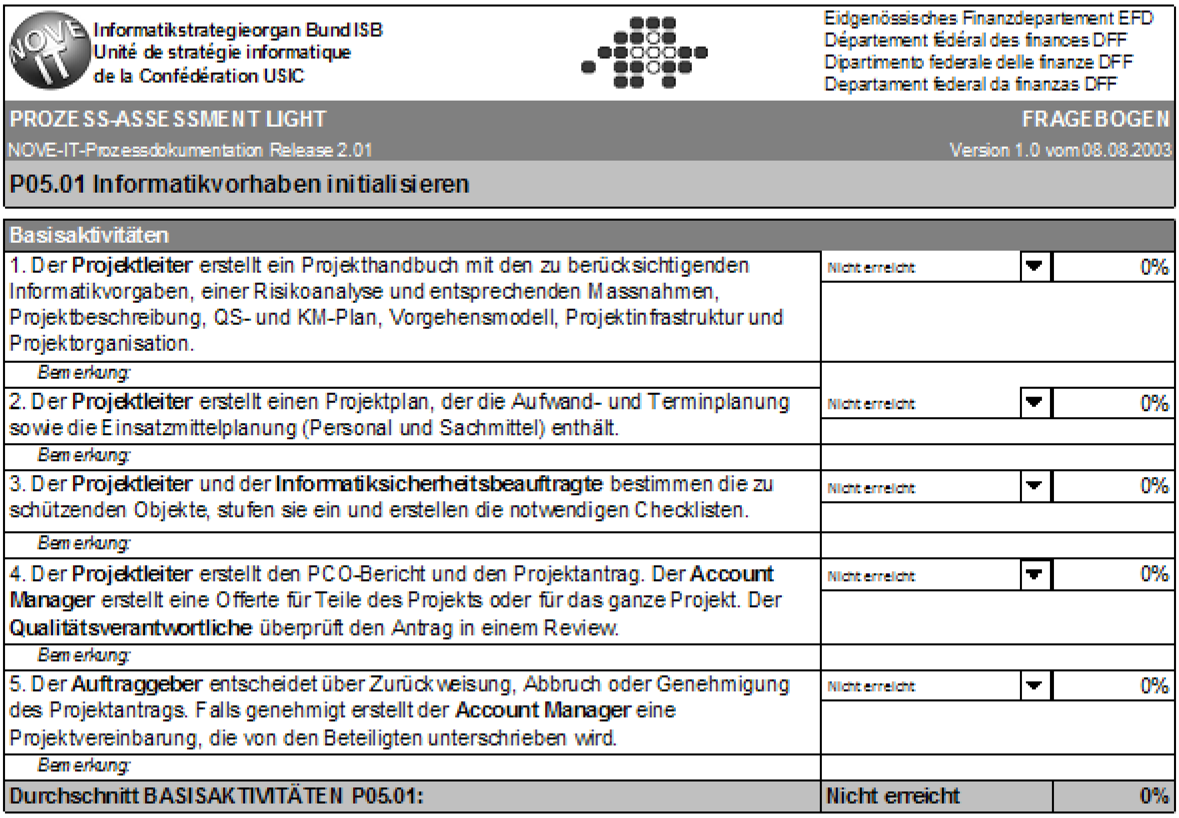
The “Process Assessment Light” is based on forms that help the assessment team during the evaluation. For each of the 9 NOVE-IT processes a form consisting of several pages has been developed. The forms are implemented in MS Excel that helps to perform some necessary calculations and allows for an electronic delivery of all questions to the assessed organisation before the assessment, as well as the results collected in the evaluation meeting. Detailed information regarding the facts that are evaluated (e.g. a specific work product) is available in tool tips that appear automatically if the mouse is moved over the question. Answers typically are given using the scale as defined in ISO 15504

- Not achieved (0 – 15%)
- Partially achieved (16 – 50%)
- Largely achieved (51 – 85%)
- Fully achieved (86 – 100%)

The forms have the following structure:

- The front page contains a summary of the information collected for all sub-processes; it also contains information about the conduct of the evaluation meeting (participants, date etc.). The front page also allows tailoring of the rest of the questionnaire. If specific sub-processes will not be evaluated this is noted on the front page, in which case the pages for these sub-processes will automatically become hidden.
- One page for the process itself contains questions regarding
    - Roles
    - Process instruments
    - Sub-processes or scenarios
    - Interfaces (input and output work products)
- Each scenario or sub-process is evaluated on a separate page. Scenarios are evaluated for the performance of the specified chain of base activities for each scenario
- Sub-processes are evaluated on the level of base activities
- At the end of the questionnaire a special page gives all participants the possibility to collect problems found and remarks, both regarding the process and the PAL.

Whenever the questionnaire asks for work products, the tool allows one to provide a rating in the usual NPLF scale, and it also allows the assessment team to specify if individual examples of a specific work product have been seen.



Informatikstrategieorgan Bund ISB
Unité de stratégie informatique
de la Confédération USIC

Eidgenössisches Finanzdepartement EFD
Département fédéral des finances DFF
Dipartimento federale delle finanze DFF
Departament federal da finanzas DFF

PROZESS-ASSESSMENT LIGHT | FRAGEBOGEN

NOVE-IT-Prozessdokumentation Release 2.01 | Version 1.0 vom 08.08.2003

P05.01 Informatikvorhaben initialisieren

Basisaktivitäten		
1. Der **Projektleiter** erstellt ein Projekthandbuch mit den zu berücksichtigenden Informatikvorgaben, einer Risikoanalyse und entsprechenden Massnahmen, Projektbeschreibung, QS- und KM-Plan, Vorgehensmodell, Projektinfrastruktur und Projektorganisation.	Nicht erreicht ▼	0%
Bemerkung:		
2. Der **Projektleiter** erstellt einen Projektplan, der die Aufwand- und Terminplanung sowie die Einsatzmittelplanung (Personal und Sachmittel) enthält.	Nicht erreicht ▼	0%
Bemerkung:		
3. Der **Projektleiter** und der **Informatiksicherheitsbeauftragte** bestimmen die zu schützenden Objekte, stufen sie ein und erstellen die notwendigen Checklisten.	Nicht erreicht ▼	0%
Bemerkung:		
4. Der **Projektleiter** erstellt den PCO-Bericht und den Projektantrag. Der **Account Manager** erstellt eine Offerte für Teile des Projekts oder für das ganze Projekt. Der **Qualitätsverantwortliche** überprüft den Antrag in einem Review.	Nicht erreicht ▼	0%
Bemerkung:		
5. Der **Auftraggeber** entscheidet über Zurückweisung, Abbruch oder Genehmigung des Projektantrags. Falls genehmigt erstellt der **Account Manager** eine Projektvereinbarung, die von den Beteiligten unterschrieben wird.	Nicht erreicht ▼	0%
Bemerkung:		
Durchschnitt BASISAKTIVITÄTEN P05.01:	**Nicht erreicht**	**0%**

Figure 5: Example of a PAL Questionnaire, sub-process P05.01

3.3 PAL Procedure

When the work on the “Process Assessment Light” started it became obvious from the beginning that simply implementing a form as the basis for the evaluation would not be sufficient. It was also discussed if the form should be distributed for a self-evaluation in the assessed organisational unit, or if it should be filled out in a guided interview.

To standardize the process of performing a “Process Assessment Light” a PAL procedure was defined. Key points of this procedure are:

- The PAL is performed as a guided self-assessment with a duration of approximately 2 hours per process.
- Several key points for the planning of a PAL are specified.
- The performance of the PAL requires several roles with identified responsibilities. Some of the roles come from the process assessed. The procedure also defines the role of a moderator who is guiding the interview session so that it follows the rules set up for the performance of the PAL. The moderator will also help to provide a uniform interpretation of the evaluation questions and the ratings.
- It is defined how and where the results of a PAL are used.

3.4 Experiences and Benefits

Both the PAL questionnaires and the procedure are available and used for the full evaluation of all processes in all departments of the Federal Government conducted in September 2003. A first trial has been conducted in June, where the results of a normal External Process Assessment (EPA) were compared to the results of a Process Assessment Light (PAL) for the same process within the same department. The ratings therefore should have been identical for those levels of the scale that are covered by both types of assessments. However the two types of assessments rated several sub-processes differently. When there were differences, most often the PAL gave a rating that was higher than the full assessment, however in some cases the opposite resulted. One possible reason that explains a lot of the differences is that the information that is acquired in a PAL is not directly collected at the source as was the case for the EPA assessment; for the sake of limiting the effort, the PAL refrains from questioning multiple individual roles but collects the information from the person responsible for the process. This person of course has a good theoretical understanding of the process itself, but may be somewhat detached from the actual daily execution of it.

It became clear during the first PALs that it is difficult to focus the rating on the *existence* of the process for a whole organisation and separate this from rating the *quality* of the process execution. In addition it has been found that the rating for a whole organisational unit tends to be too high. The aggregation of individual results from areas where a process has been only partially implemented gives the impression that evidence for every aspect of the process has been found, even if only fragments of the process may exist in each implementation of it.

Although the above-mentioned problems exist, the main benefit that is expected from the PAL is the full overview of the level of implementation of all processes across all departments of the Federal Government. The PAL will be the means to collect the basic information about the current status, which can the be used for several purposes like

- To encourage the discussion between line management and process owners about processes and process interdependencies
- To clarify the current implementation status by department
- To find possible barriers that block specific parts of the process model from being implemented
- To find and define critical areas that will be examined in more detail using the standard external assessment.

4 Conclusions

The NOVE-IT project uses a variety of methods to evaluate the implementation of the NOVE-IT process model in the Swiss Federal Government. SPiCE-like external process assessments provide much information but are too time-consuming for regular monitoring. A form-based procedure, the Process Assessment Light, has been developed to provide a full overview of the level of implementation of all processes across all departments of the Federal Government, however without completely replacing the full external assessment of a process in the accuracy of the results provided. The Process Assessment Light has been used for the first time in September 2003.

5 Literature and References

5.1 Literature

[BCK02] J. Becker, M. Kugeler, M. Rosemann: Prozessmanagement. Ein Leitfa-den zur prozessorientierten Organisationsgestaltung. Springer-Verlag, Heidelberg, 2002.

[ESA] Cass, A., Völcker C., et. al., "SPiCE for SPACE: A Process Assessment and Improvement Method for Space Software Development", *ESA Bulletin Number 107*, August 2001.

[ISB-SPICE] Beurteilung der externen Prozess-Assessments (nach ISO 15504) des Informatikstrategieorgnas Bund ISB durch die META Group Schweiz AG, part of the information pages of the FSUIT [ISB]

[ISO15504] ISO/IEC TR 15504-1 bis -9:1998 (E): Information technology — Soft-ware process assessment — Part 1 bis Part 9. Der ISO TR, auf welchem die angewandte Assessmentmethode basiert (http://www.iso.ch/).

[SCHM01] H.J. Schmelzer, W. Sesselmann: Geschäftsprozessmanagement in der Praxis. Carl Hanser Verlag, München, 2001.

[OSTR00] M. Osterloh, J. Frost: Prozessmanagement als Kernkompetenz. Gabler, Wiesbaden, 2000.

5.2 Internet Links

[BPG] Federal employment law, Bundespersonalgesetz vom 24. März 2000
http://www.admin.ch/ (Systematische Rechtssammlung)
http://www.personal.admin.ch/ (Thema Bundespersonalgesetz)

[COBIT] Official COBIT website: http://www.isaca.org/cobit.htm

[EXPA] External process assessments based on ISO/TR 15504, conducted by SynSpace AG in the framework of the mandate in NOVE-IT. Links: http://www.synspace.com/, specifically the pages http://www.synspace.com/D/Products/nice.html

[EFQM] European Foundation for Quality Management, http://www.efqm.org/

[IDEF] Modelling of the interfaces and processes using IEDFØ was done with the tool “Workflow Modeler” from Meta Software Corporation. Link: http://www.metasoftware.com/

[ISB] Information pages of the Federal Strategy Unit for Information Technology, with additional links to HERMES and NOVE-IT: http://www.isb.admin.ch/

[ITIL] Official ITIL website: http://www.itil.co.uk/

[ITPIN] Special Interest Group IT Process Improvement Network (ITPIN) of the SwissICT: Downloadable IT process map that takes into account the basics of ITIL.
http://www.swissict.ch/ag/itpin/

[ITPM] Process reference model ITPM, which was used as a basis for NOVE-IT within the framework of the mandate of IBM Business Consulting Services in NOVE-IT.
Links: http://www.ibm.com/, insbesondere die Seite http://www.ibm.com/services/whitepapers/

[NIT] Information pages about the programme NOVE-IT within the Swiss Federal Administration, http://www.nove-it.admin.ch/

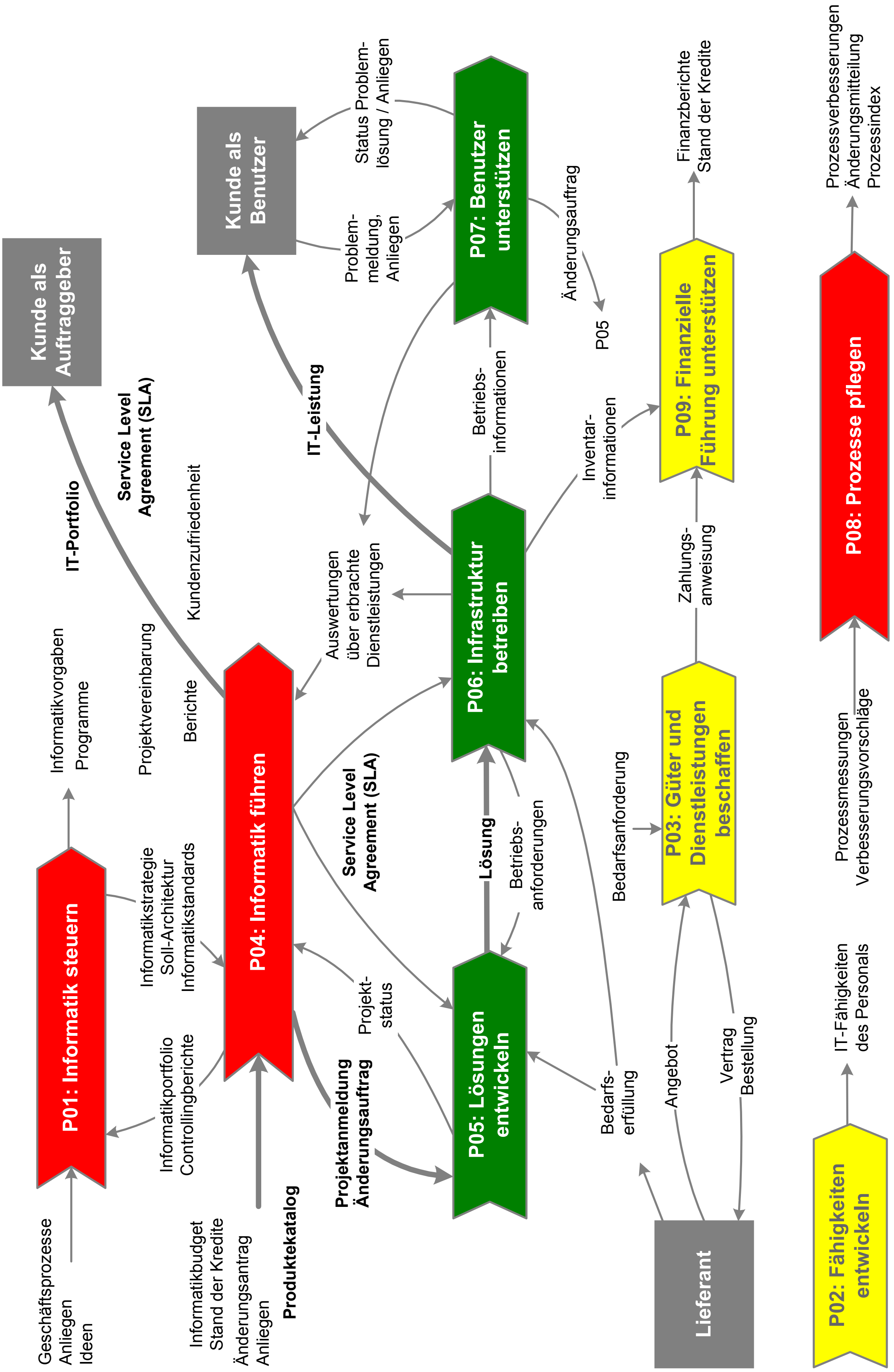


Figure 6: Process Map, Detailed

Author CVs

Daniel Keller, Swiss Federal Strategy Unit for Information Technology

Daniel Keller graduated from the Institute of Organic Chemistry in 1983 with a doctoral degree. Since 1991 he is working for the Swiss Federal Government, and since 2000 he is a Process Manager in the Federal Strategy Unit for Information Technology (FSUIT). The main focus of his work is the development, maintenance and implementation of the IT processes in the Swiss Federal Government.

Ann Cass, SYNSPACE SA

Ann Cass graduated from the University of California, San Diego, with a Ph.D. in experimental plasma physics in June 1998.

Since 1999, she has been a consultant at SYNSPACE AG, specialising in coaching space and commercial clients in Software Process Improvement. Ann is a co-author of SPiCE for SPACE (S4S), an ISO 15504-based method of software process assessment for the European Space Agency (ESA). She has performed over 10 assessments either as Lead or Support Assessor.

Philipp Sutter, SYNSPACE AG

Before joining SYNSPACE, Philipp worked 10 years for a large healthcare company. He became familiar with all aspects of development, implementation, release, training and maintenance of chemical plants, their control systems, and the related FDA conformant validation documentation.

At SYNSPACE, Philipp performs SPiCE Assessments and works on Software Process Improvement. He is the project leader of SYNSPACE'S part of the NOVE-IT project.

Identification of Improvement Issues Using a Lightweight Triangulation Approach

Tony Gorschek (tgo@bth.se) and Claes Wohlin (*claes.wohlin@bth.se*)
Department of Software Engineering and Computer Science,
Blekinge Institute of Technology, PO Box 520, SE-372 25 Ronneby, Sweden
Phone: +46 457 385000

Abstract

One of the challenges in requirements engineering is the ability to improve the process and establish one that is "good-enough". The objective of this paper is to present a lightweight approach to identify process improvement issues. The approach is developed to capture both the views of different stakeholders and different sources of information. An industrial investigation from a small company is presented. In the investigation both projects and the line organization have been interviewed and documentation from them has been studied to capture key issues for improvement. The issues identified from one source are checked against other sources. The dependencies between the issues have been studied. In total nine issues for improvement of the requirements engineering work at the company were identified. It is concluded that the approach is effective in capturing issues, and that the approach helps different stakeholders to get their view represented in the process improvement work.

Keywords

Requirements Engineering, Software Process Improvement, Data Triangulation, Process Evaluation.

1 Introduction

The area of requirements engineering (RE) is often underestimated in value in the area of software engineering, in spite of the fact that requirements problems are the largest cause in project failure. [1, 2].

This paper presents an investigation conducted in industry aimed at introducing a way to identify RE *improvement issues* in small and medium sized enterprises (SME) [3]. The lightweight approach presented uses data point *triangulation* [4] as a means to identify possible improvement issues. The data points consist of two major parts, i.e. a *line study*, and a *project study*. This is depicted in Figure 1. The project study is comprised of three exploratory case studies [4], and each case study is in turn comprised of project specific interviews and documentation (see section 3.2.).

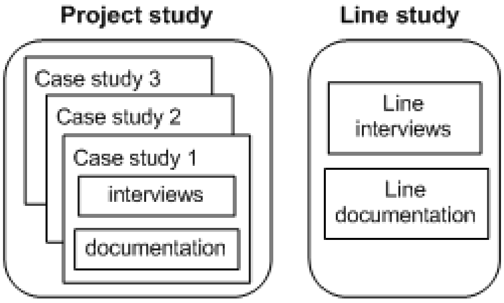


Figure 1. Investigation description.

The line study is comprised of line interviews and documentation (see section 3.3.). The main objective of the investigation was to ascertain state-of-practice at Danaher Motion Särö (DHR) (see section 2.) and to identify improvement points in their RE process. There are approaches for RE process improvement targeted at e.g. SME. Sommerville and Sawyer [5] for instance present a series of guidelines as well as a guide for process improvement as a product of the REAIMS project [6]. The PERE (Process Evaluation in Requirements Engineering) method used for analysis of the requirements engineering process from two different viewpoints in REAIMS is basically a systematic approach to understanding processes, and a human factors analysis of the process. The lightweight approach presented in this paper uses data point triangulation with *four* main data sources (see section 3.1), In addition to this the emphasis of the approach presented in this paper is on fast and fairly low-cost evaluation.

In addition to RE specific improvement strategies there are some more general, targeted at quality assurance in software projects, such as *The TickIT Guide* (ISO 9001:2000) [7]. These process improvement and quality assurance approaches do not offer any lightweight identification of how to establish what needs to be improved in an RE process, i.e. how to identify improvement issues.

The main objectives (contributions) of the paper is to present (i) the lightweight triangulation approach used for the identification of the improvement issues (presented in section 3), (ii) the improvement points identified and triangulated (presented in section 4), and (iii) the dependencies between the improvement points (also presented in section 4). In addition to this there is an overview of state-of-the-art in relation to each improvement point (see section 5). Section 6 contains the conclusions drawn in the paper.

2 Investigation Context

Danaher Motion Särö develops and sells software and hardware equipment for navigation, control, fleet management and service for Automated Guided Vehicle (AGV) systems. More than 50 AGV system suppliers worldwide are using DHR technologies and know-how together with their own products in effective transport and logistic solutions to various markets worldwide. The headquarters and R & D Centre is located in Särö, south of Gothenburg, Sweden. DHR has approximately 100 employees. DHR is certified according to SS-EN ISO 9001:1994, but is uncertified according to CMM and CMMI.

DHR has a wide product portfolio, as the ability to offer partners and customers a wide selection of general variants of hardware and supporting software is regarded as important. Tailoring and especially lighter customer adaptation often follows the procurement and subsequent installation of a sys tem. This in addition to development of new software and hardware makes it a necessity to plan, execute and manage a wide range of projects.

Requirements are one factor that binds all of the projects together. It is not necessarily so that requirements breach project boundaries (although this is known to happen) but rather that most projects involve elicitation, analysis and negotiation, management and documentation of requirements. In addition to this the stakeholders (or groups of stakeholders) are at least as divergent and diversified as the projects themselves. Requirements from general sources like *the market* have to be taken into consideration as well as the ones from *industry-partners*, *end-customers* and *internal sources* such as developers and management. All of these factors contribute to a need for an RE process that is good enough (at present) and has the potential to meet growing demands and complexity (in the future). The ability to prepare for the future, and improve current practices, is the underlying motivator for the work conducted in the partnership between DHR and Blekinge Institute of Technology.

The first step in this partnership was to map the RE process at DHR (establish a baseline), and to determine what the main improvements issues were.

3 Investigation Design

In this section the overall design of our *multi method investigation* is described, how the data was elicited for each data source (project study and line study), compiled and analyzed. A validity evaluation is also presented below.

3.1 Multi Method

By looking at several data sources instead of relying on a single one, e.g. solely a case study, a higher level of validity can be achieved [8]. Below four major data sources are described, (A) data from the case study interviews, (B) data from project documentation, (C) data from line interviews, and (D) data gathered from line documentation. (A) and (B) together comprise the project study, i.e. interviews and project documentation from three specific projects (see section 3.2.). (C) and (D) are interviews and documentation from the line, and together comprise the line study. The idea is not to use all of the data sources solely for the purpose of getting more data, but rather to have a confirmation (validation) of the individual issues identified. This is achieved through triangulation, illustrated in Figure 2. One issue is identified and specified, then checked against the other data sources for confirmation. An additional benefit is that this usage of different sources enables several perspectives, decreasing the possibility of missing crucial information.

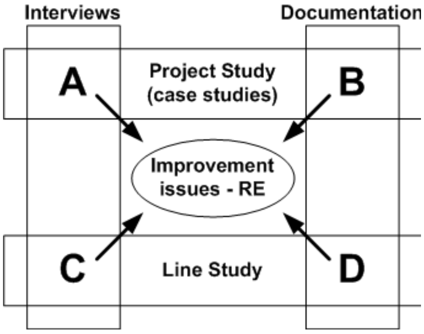


Figure 2. Triangulation using several data sources.

In addition to getting project specific data (A) and (B), data is elicited from the line (C) and (D), i.e. the development support and production parts of the organization (see section 3.3.).

In addition to the horizontal division of the data collected there is a vertical distinction to be made. Data gathered from the interviews, (A) and (C), are complemented and/or verified (occasionally contradicted) by the data gathered from the documentation, (B) and (D). When conducting the triangulation two leading data sources were selected, namely the interviews (A) conducted in the case studies and the interviews conducted in the line study (C). The reason for selecting leading data sources was the need for points of reference that could be compared (triangulated). If no leading source was selected every single potential data point had to be considered and triangulated. This would have been possible but very time and resource consuming, and the final improvement issues identified would be the same.

It was however not a foregone conclusion that specifically the interviews were to be chosen as the leading data sources. The main reason for the decision was the fact that the interviews reflected the views of the project and line team members. In addition to this a piece of documentation (whether it was from the project study or the line study) was primarily used for confirmation of the issues identified during the interviews. If the documentation was used for the identification of issues themselves all things not present in the documents could be considered potential issues. This became a problem if the contents of the documents were compared to state-of-the-art, in which case the absence of things in DHR's documentation could yield any number of issues depending on what sources in state-of-the-art were used for comparison. The idea was not to identify all things not performed and turn each these in to an issue (basically getting a maximum sized RE process), but rather to identify what needed to be changed/added to DHR's RE process based on their own perception and experiences.

Three other reasons existed for making the interviews the leading data source. The case study interviews were the source of pre-verified quantitative data (the pre-verification is described in section 3.2.1.), the case studies were the source of the most up-to-date data (three current projects were studied), and last the case studies yielded the lion share of the total data collected.

3.2 Project Study (Case Study) Design

In order to get an overview of the RE process no single case could be studied that was representative for most of the projects conducted at DHR, rather a block of projects had to be identified [9]. Several projects had to be considered and three were finally chosen. The main criteria were getting a cross-section of typical projects conducted, and to get a wide representation of the developers and managers involved in the projects, i.e. avoid asking the same people questions about different projects. Company representatives with an overview of DHR's domain, project portfolio and current practices picked three projects. The projects chosen were of three types:

- ***Project Alpha*** Externally initiated software development project aimed at an end customer. Basically an addition of customer specific features to an existing piece of software.
- ***Project Beta*** Externally initiated software development project aimed at an industry partner. Basically development of a new service tool to be used by an industrial partner in their work with end-customers.
- ***Project Gamma*** Internally initiated software and hardware project aimed at updating a certain central product. Basically a generational shift to newer technology for mainly production aspects.

The division of the projects can be seen as two steps, first there is the obvious distinction between the projects from the point of what sort of product they yield, bespoke (alpha and beta) or generic (gamma). As described by Sommerville [10] bespoke products are aimed at specific customers and the projects are initiated by external sources, generic products however are rather aimed at a market and initiated internally by the producer. Second there was a need for a further granulation of the "bespoke project type". This is due to the fact that project alpha was aimed at an end-customer, while beta was aimed at an industrial partner. The preconditions of a partner project differ from that of an end-customer. Industrial partners often have more domain specific and technical knowledge.

Subsequent to project division and selection, interviews were booked with representatives for all *roles* in the projects at hand. Four major roles were identified that were typical for all of the projects in question, *Orderer*, *Project Manager*, *System Engineer* and *Developer*. These official roles can briefly be described in the following manner:

1. The *Orderer* has the task of being the internal owner of a certain project, i.e. has the customer role and if applicable the official contact with an external customer and/or partner. This party is responsible for the official signing-off when it comes to the requirements, i.e. he places an order.
2. The *Project Manager* has the traditional role of managing the project, resources, planning and follow-up. As far as requirements are concerned the Project Manager is responsible for that the requirements engineering is performed, the requirements specification is written and signed off by the System Engineer.

3. The *System Engineer* is the technical responsible for a project. It is also important to recognize that the System Engineer has the official responsibility for the requirements specification in a project.
4. The *Developer* is a representative for the developers (e.g. programmers) in a project, the ones actually implementing the requirements. The developers use the requirements specification.

The researchers in cooperation with representatives for DHR identified these roles. In total 11 people were interviewed during the case studies, four each from projects Alpha and Gamma, but only three from project Beta. The Orderer from project Beta was not accessible at the time of the investigation.

3.2.1 Case Study Interview Questions

Semi-structured interviews [4] were used for the case studies. There were 31 questions in total, two warm-up questions where the subjects stated name, title and responsibilities etc. Following warm-up the questions were divided into four main parts, i.e. *requirements elicitation*, *requirements analysis and negotiation*, *requirements management* and a *general part*. The questions posed in the first three parts were aimed at finding out what tasks were performed in the context of RE for the project in question, who performed them, when and how. An example of such a questions could be “was there any classification of the requirements into classes, groups or types during the elicitation process”, and the follow-up questions “how and by whom was this done”, and “what were the criteria for the classification…”. The subjects were asked to elaborate on their answers, especially if the answer was a simple “no”. The intention is to get an idea of why a certain task was not performed. The structure of the questionnaire and the tasks that the questions were based upon were inspired primarily by the work of Sommerville & Sawyer [5] and Gorschek et al. [11]. The questions posed in the first three parts were of a qualitative nature.

The general part (fourth) contained three questions (from a total of four) of a quantitative nature. Each of these questions was comprised of several sub-questions, and two of them were *open*, i.e. up to the subject to come up with the contents and not merely answer if a certain task was performed. An example of such a question was “list three things that work best with respect to RE at you company” and the follow-up “…three things that have the best improvement potential with respect to RE…”. These open questions were beneficial from at least three aspects, i.e. the data gathered during preceding questions could substantiate the answers given here, items missed by earlier questions could be observed, and last the subjects were asked for their opinion without being led in a certain direction.

When the interviews were completed the results were compiled and the quantitative results were used as a primary source of data, and the qualitative results were used as a secondary source as well as to validate the quantitative data. Figure 3 shows the procedure, the reasoning behind this was to be able to see patterns in the quantitative data that could be substantiated (validation) and augmented by the data gathered from the qualitative part. The confirmed data was then used as one of the sources (A) in the triangulation (see Figure 2).

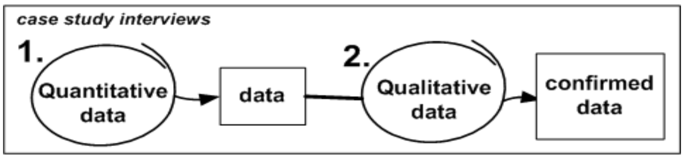


Figure 3. Pre-validation of case study data.

The subjects were given the questions on location. Each interview took about one to one and a half hours. All of the interviews were conducted over a period of five consecutive days. In addition to general information given to the subjects they were assured that all answers would be anonymous. The questionnaire used can be obtained through the authors. A digital recorder was used during the interviews in addition to the notes taken, this to enable further clarification and analysis after the fact.

3.2.2 Project Documentation

The study of documentation was a substantial part of the overall investigation. The documentation in question was of two types, first there were documents pertaining to the projects investigated in the case studies (B), and second there was line documentation (D) (see Figure 2).

Project documentation included pre-study documents, project specifications, project plans, requirements specification, subsequent follow-up documentation, budget and time reports and meeting minutes.

3.3 Line Study

In addition to the 11 case study interviews conducted three line interviews were performed. This was done to elicit data from three additional roles not present in any of the block projects. The roles interviewed consisted of representatives for *System Test*, *Production* and *Application Developer.* The descriptions of the roles below are according to the official line documentation.

- The System Test role can be described as the traditional role of application and feature test. This role is officially present during initial project meetings and is a part of the verification of the requirements specification.
- The Production role also has a presence in projects. This representation consists of verifying that the productional aspects are met, i.e. that production is taken into consideration at an early stage.
- The Application Developer role represents installation and adaptation aimed at industry partners and/or end customers. Adaptation here translates to tailoring, development of some light customer features and some support.

The interviews conducted here were unstructured [4], i.e. some directional questions were used, but for the most part informal conversation dominated. The conversational interview's goal was to elicit data about the lines role in the RE process, the representatives opinions and experience. The structure observed during the case study interviews (see section 3.2.1) was present to a certain degree, i.e. the interviewer wanted answers to some of the same questions, but no questionnaire was formally used. The reason for using a more unstructured approach during the line interviews was due to that they were somewhat more general in nature, i.e. did not refer to a specific project. The same general information and assurance of anonymity as before was issued.

3.3.1 Line Documentation

Line documentation (D) (see Figure 2) included general process descriptions, development process descriptions, process improvement strategies, documentation pertaining to roles and responsibilities, and the requirements engineering process description. Generally one could say that the line documentation was a collection of documents in which the official processes and responsibilities were specified.

3.4 Validity Evaluation

In this section we discuss the threats to this investigation. We base this on the discussion of validity and threats to research projects presented in Wohlin et al. [12]. One type of threats mentioned in [12] is not relevant, since the investigation is conducted in an industrial environment. The threat not considered is construct validity, which mainly is concerned with the mapping from the real world to the laboratory. The investigation presented here is however conducted in the real world. The validity threats considered are: conclusion, internal and external validity threats respectively.

3.4.1 Conclusion validity

The questionnaire used for the structured interviews was validated through preliminary testing and proofreading by several independent parties, this to avoid factors like poor question wording and erroneous formulation.

Each case study interview was conducted without any break. Thus the answers were not influenced by internal discussions about the questions during e.g. coffee breaks.

The sampling technique used for the case studies, i.e. projects selected and interview subjects for every project, can pose a threat to the validity of the investigation. The projects may not be totally representative for the projects conducted at DHR, in a similar manner the interview subjects may also not be representative for the role they represent. Three things alleviate these potential threats. First the fact that three cases was studied, and this also gives us three representatives interviewed for each role identified. The exception to this is the Orderer in project B. Third is the triangulation effect, i.e. data from different data sources is used for validation.

3.4.2 Internal Validity

The use of a digital recorder during the interviews could be considered as a problem due to the fact that certain people may feel constrained to answer differently if an interview is recorded. This potential problem was alleviated by the guarantee of anonymity as to all information divulged during the interview, and that the recordings are only to be used by the researchers.

3.4.3 External Validity

The external validity is concerned with the ability to generalize the results. This is not a main threat in the investigation, since the objective is not to generalize the improvement issues to another environment. The important generalization here is whether the applied approach for identifying improvement issues is possible to apply in other environments. There is nothing in the approach that makes it tailored to the specific setting ad hence the approach should be useful at other small and medium sized enterprises that would like to identify improvement issues in the requirements engineering process.

4 Results

4.1 Project Study

4.1.1 Case Study Interviews (A)

Data from the case study interviews resulted in nine general improvements issues summarized in Table 1. In the first column there is a unique id for each improvement issue, the second column holds the name of each issue, and the last column houses the support number of each issue. The support number is the total number of times an issue was brought up during the interviews, i.e. issue 1 has a support number of 5, meaning that five out of eleven people interviewed brought up this issue during the open questions (see section 3.2.1.).

Issue Id	Improvement Issue	Support number
1	Abstraction level & Contents of requirements	5
2	Requirements overview in projects & over project boundaries	4
3	Requirements prioritization	4
4	Requirements upkeep during & post project	4
5	Requirements responsible/owner with overview of the requirements	3
6	Roles and responsibilities RE process	2
7	System test performed against requirements	2
8	Customer relations during RE	2
9	RE process/methods	3

Table 1. Improvement issues case study.

Issue-1: A central issue for improvement is how the requirements are specified, contents of each requirement and to establish a detailed enough level of description. This relates to usability and understandability of the requirements (how well a specified requirement depicts what it is intended to), and to comparability between requirements.

Issue-2: Requirements overview in projects and over project boundaries is an issue viewed as important. In order to facilitate this overview three main other issues have to be satisfied, i.e. Issue-1, Issue-4 and Issue-5. Figure 4 illustrates this, where the relations between the nine issues can be viewed. Issue-2 is *dependent on* Issue-1, Issue-4 and Issue-5 (denoted by the arrows). The relations in Figure 4 can be several levels deep, e.g. Issue-2 is dependent on Issue-4 which in turn is dependent on Issue-5 and so on. Below only the direct relations are addressed during the description of each issue.

Issue-3: In order to meet the demands posed on projects by resource and time limitations, requirements should be prioritized. The level of prioritization should reflect the need on a project basis. Prerequisites for this issue are that the requirements are specified in an adequate way and at a comparable level of abstraction (Issue-1), and that there is an overview of the requirements in question (Issue-2).

Issue-4: The requirements (requirements specification) should be kept up-to-date during and post project. The point of keeping the requirements 'alive' during the project, and not discard them after a certain point in the project, was made.

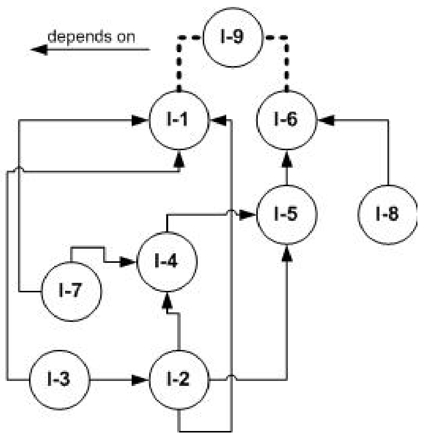


Figure 4. Dependency diagram.

Issue-5: There should be a person(s) responsible for the RE as a whole that has an overview of the requirements. This role should further be the owner of the requirements, i.e. be up-to-date on the requirements and have executive powers and responsibilities pertaining to change to and addition of new requirements. This issue has an obvious relation to Issue-6.

Issue-6: The roles and responsibilities of project members and other stakeholders, e.g. customer, should be clearly and unambiguously defined pertaining to RE.

Issue-7: System tests should be performed against the requirements. Prerequisites for this are that the requirements specification is kept up to date during the project (and post project in some cases) (Issue-4), and that the requirements are specified adequately and at a specified level of abstraction.

Issue-8: The relations and particularly the flow of information between the customer (a group name used for all executive stakeholders, i.e. stakeholders representing the customer in a formal and executive capacity [11]) and the project team vary with project. There is a general view that the information from the customer is passed through too many filters before it reaches the project team. In addition to this the customer is often not accessible (at least not directly) for elicitation and clarification purposes during a project. A prerequisite identified here for this issue is that roles and responsibilities during RE are defined (Issue-6) (see Figure 4), i.e. all persons involved in the RE process (including the customers) are stakeholders, thus they are a part of the responsibility paradigm as well as the members of a project team.

Issue-9: This issue refers to the view that there is a lack of a formalized and/or structured RE process, or methods supporting certain tasks performed during RE at DHR. Issue-9 is to some extent outside of the dependency diagram (see Figure 4) as far as the level of abstraction. All issues except Issue-9 can be described as more or less specific tasks to be performed or rules to be upheld, while Issue-9 rather is an 'umbrella' issue under which the rest can be sorted. Many of the other issues contributes to Issue-9, this is illustrated in Figure 4.

4.1.2 Case Study Project Documentation (B)

Looking at the project documentation in the case study projects several of the improvement issues described in section 4.1.1. are substantiated. Table 2 shows each issue (sorted by issue id), the cells in the B-column (B for triangulation point B, see Figure 2) with a ticked box denote the substantiation, and the note-column offers a short account for it. Four issues could not be directly substantiated by the project documentation, i.e. Issue-2, Issue-5, Issue-6 and Issue-8.

4.2 Line Study

4.2.1 Line Study Interviews (C)

Looking at the line interviews several of the issues described in 4.1.1. are substantiated (denoted by the ticked boxes in column C of Table 3, C standing for triangulation point C, see Figure 2). Four issues could not be directly substantiated by the line interviews, i.e. Issue-2, Issue-3, Issue-5 and Issue-8. It is important to notice that three new improvement issues are identified during the line interviews, i.e. Issue-10 to Issue-12.

Issue-10: There are no mechanisms or practices implemented for requirements reuse. There is reuse on the design/code/function/solution levels however.

Issue-11: No traceability policies are implemented for requirements. In some cases it is possible to trace requirements to the initiating party (backward-from traceability) [5, 11], and to partially trace requirements through design documents to implemented components (forward-from traceability) [5, 11]. These traceability possibilities are however dependent on individual knowledge by project team members and they are not documented.

B	Issue Id	Note
☑	1	The abstraction level and contents of the requirements tend to vary. This is seen within projects, and over project boundaries, i.e. requirements are not comparable.
	2	
☑	3	No requirements prioritization is documented, and no priority grouping can be observed.
☑	4	The update of the requirements specifications (RS) varies, however no RS was ever updated after the first 50% of the project calendar-time had elapsed.
	5	
	6	
☑	7	The problems with Issue-1 and Issue-4 make system test based on requirements difficult.
	8	
☑	9	The RE process/methods seems not to be good enough. This is based primarily on issues substantiated above, but the fact that the level of RE and the methods used varies with project, also indicates this.

Table 2. Improvement issues documentation.

C	Issue Id	Note
☑	1	Abstraction level & Contents of requirements varies and may be inadequate for tasks such as system test.
	2	
	3	
☑	4	Outdated req. specifications make it difficult to use the req. document.
	5	
☑	6	The roles of the interviewed as to their roles and responsibilities in RE are not clear.
☑	7	System test are seldom performed against requirements due to Issue-1 and Issue-4.
	8	
☑	9	See Issue-1, Issue-2, Issue-4 to Issue-8
Improvement Issues from line interviews		
NEW	10	Requirements reuse
NEW	11	Requirements traceability
NEW	12	Requirements version handling

Table 3. Improvement issues line interviews.

Issue-12: There is no version handling of individual requirements. The requirements document itself is however updated and released with different versions.

4.2.2 Line Study Documentation (D)

Several issues identified thus far (see section 4.1.1. and 4.2.1.) were not substantiated by the line documentation, i.e. Issue-2 and Issue-4 to Issue-8. Table 4 offers an overview of this.

D	Issue Id	note
☑	1	The template used for specifying requirements is fairly abstract, i.e. on a high level of abstraction. There are no detailed instructions and/or no detailed examples.
	2	
☑	3	No instructions/help/regulations/method description exists to aid in the prioritization of requirements.
	4	
	5	
	6	
	7	
	8	
☑	9	see Issue-1, Issue-3
Improvement Issues from line interviews		
☑	10	No policies for requirements reuse exist.
☑	11	No policies for requirements traceability exist.
☑	12	No policies for requirements version handling exist.

Table 4. Improvement issues line documentation.

4.3 Triangulation of Results

Table 5 offers an overview of all issues presented, and information about substantiation from the different data sources, i.e. A to D. Nine out of a total of twelve issues were substantiated by two or more data sources (see section 3.1.), and are considered to be triangulated.

Issues 2, 5 and 8 each have a total triangulation value of one, i.e. the proof for the issues could not be distinctively identified in the project documentation (B), were not mentioned during the line interviews (C), and could not be found in the line documentation (D). Thus, these are not viewed as being triangulated.

Issue Id	A	B	C	D	TOTAL
1	☑	☑	☑	☑	*4*
2	☑				*1*
3	☑	☑		☑	*3*
4	☑	☑	☑		*3*
5	☑				*1*
6	☑		☑		*2*
7	☑	☑	☑		*3*
8	☑				*1*
9	☑	☑	☑	☑	*4*
10	▨	▨	☑	☑	*2*
11	▨	▨	☑	☑	*2*
12	▨	▨	☑	☑	*2*

Table 5. Triangulation of improvement issues.

5 Relation to State-of-the-art

In this section each triangulated issue (see section 4.3.) is posted with a description of how some sources in state-of-the-art can contribute in resolving them.

Issue-1: *Abstraction level & Contents of requirements*. Requirements should be stated in a clear and unambiguous way. Although there are many textbooks describing how this is done many specifications are inadequate [1, 13, 5, 14]. Requirements are in addition totally dependent on the customer, i.e. the stakeholders making the demands. Here it is assumed that the elicitation process and stakeholder consulting [5, 11, 15] is not a factor, i.e. all information is available and understood by the party writing the specification. This is however seldom the case, i.e. lack of domain knowledge and miscommunication between developers and customers is often a factor influencing the quality of a requirement [16].

Natural language (NL) specifications are commonly used and there are several ways in which to approach this, spanning from the psychology behind getting the right requirements on paper [17], to how important linguistics [18] and ethnography [19] are in the context.

Formal specifications [20] are an alternative (or a complement) to NL specifications. Either way several other techniques can be used in combination with both, e.g. prototypes, scenarios, state diagrams, use cases, data-flow diagrams and so on, to validate and/or clarify a certain requirement [10, 13, 5, 14].

Issue-3: *Requirements prioritization*. To have a shortage of requirements is seldom a real issue, quite the opposite. This makes prioritization of the requirements a valuable tool in the initial stages of a project. There are several methods for prioritization [21, 22], one of the most well known is the Analytic Hierarchy Process or AHP for short [23]. The main issue with this method is scale-up issues, i.e. prioritizing of a lot of requirements is rather time consuming [21]. Getting the customers (or other departments) to give input, i.e. do some prioritization of their own, and show what is important to them, is an approach that could be beneficial, and combined with ones own prioritization efforts [24]. The exact method used depends on any number of factors, e.g. like amount of requirements, time devoted to prioritization and so on. The main issue here is not to recommend a prioritization method, but to recommend that a method be used.

Issue-4: *Requirements upkeep during & post project.* Requirements change. Making the requirements reflect these changes is crucial for several reasons. It is a prerequisite for being able to conduct tests based on the requirements (Issue-7). Furthermore in order to achieve any reuse (Issue-10) or re-prioritization of requirements this issue has to be resolved. Figure 5 illustrates an updated version (only triangulated issues) of dependencies between issues. Issue-4 is here dependent on Issue-6, i.e. that there is a clear and unambiguous definition of the roles and responsibilities pertaining to RE, this includes who should be responsible for making sure that the requirements are kept up to date.

Issue-6: *Roles and responsibilities RE process.* The importance of making roles and responsibilities clear is not an issue reserved for RE, but pertinent in any organization involved in software development [25]. One issue for RE is to identify the tasks to be conducted [5, 11], and then assign responsibility for them. Typically one would assume that there is one person responsible for the RE pertaining to a certain project, this does not however necessarily imply that this individual has to perform all tasks in question. The tasks could be delegated to any number of project team members. It is important that there be no question about who should perform what tasks. The delegation and division of tasks should be documented and rooted in the project organization in the same manner as the general project model.

Issue-7: *System tests performed against requirements.* Requirements are typically described as *what* is to be delivered to a customer [5]. By performing system tests on requirements it is possible to validate if the implemented features are really based on the requirements (functional requirements) [14], and if the system complies with other requirements (non-functional requirements) [14]. Making test scenarios/cases during initial requirements engineering can also be a good way of validating the requirements themselves at an early stage [5].

Issue-9: *RE process/methods*. All issues are included here. There are several RE processes suggested in literature [5, 11, 14], all more or less based on similar concepts and ideas. The main question for an organization to decide is what tasks are necessary to achieve their goals.

Issue-10: *Requirements reuse.* Reuse is a fairly common term in software engineering and often refers to the reuse of everything from code to architectures. The reuse of requirements however is not as commonly adopted [26]. This does not mean that the potential benefits of such a reuse are less, quite the opposite actually. By reusing requirements (traceability is a prerequisite, Issue-11, see Figure 5 [27]) everything from the requirement itself to analysis, design, implemented components, test cases, scenarios, and use cases etc. can be reused [5].

Issue-11: *Requirements traceability*. Several types of traceability could be identified in RE [11]. Only two are mentioned here, i.e. *Backward-from traceability* denoting a link from requirement to their source in other documents or people, and *Forward-from traceability* denoting a link from requirements to the design and indirectly to the implemented components [5, 11]. The first is important for verifying what sources there are for a certain requirement [28], and the second for making reuse of more that just the requirements themselves possible. There are however several views on requirements traceability [5, 11, 29] which should be studied before a distinction about what model for traceability should be implemented in an organization.

Issue-12: *Requirements version handling*. This issue could effectively be sorted under the umbrella of traceability [5, 11], but is separated due to the fact that this issue was identified separately from Issue-11 during the line interviews (see section 4.2.1.). Version handling of code in classes, components etc. is widely used in software engineering. The benefits are many, the most obvious one being the ability to go back and look at/use/compare to previous versions of an item. The same is true for a requirement, i.e. version handling enables an evolutionary view. In order to obtain this some kind of version handling system must be used, e.g. as a part of a requirements handling application.

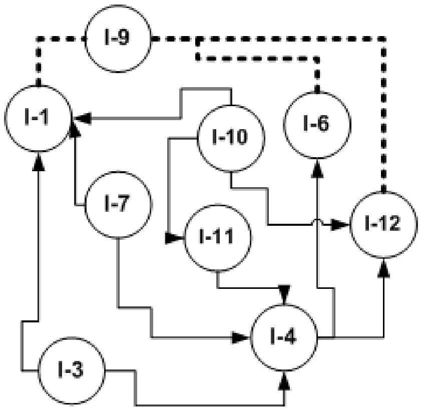


Figure 5. Dependency diagram (updated).

6 Conclusions

By conducting two separate studies, i.e. the project study and the line study, we were able to identify several improvement issues in DHR's RE process. Using the lightweight triangulation approach nine (out of a total of twelve) improvement issues were triangulated. The result was nine tangible issues that positively identified improvements to be made in the RE process at DHR.

A good enough RE process is crucial in order for development projects to be successful, it is however also crucial to have the ability to plan for the evolvement of the process. Process improvement is however often a costly enterprise tying up valuable resources during the improvement, and subsequent indoctrination of the improved process often means that substantial educational measures have to be taken. Ad hoc improvement measures, i.e. trying to improve the RE process as a whole, can mean that issues not crucial are improved/implemented, thus the process improvement costs are higher. Cost is a central concern for all organizations, and especially in the case of SME:s.

By identifying tangible and crucial improvement issues through eliciting information from personnel, management and documentation, and subsequently triangulating the issues identified, validate that the right problems are prioritized. A prioritization of issues gives the organization the raw material for the construction of a process improvement plan that addresses primarily crucial issues. Secondary issues, e.g. the ones identified during the studies but not triangulated, can be addressed at a later stage or the improvement approach could be used again.

It is also important to realize that the nature of the lightweight approach described in this paper bases its results on the knowledge and views of the people in the organization whose RE process is to be improved. This is beneficial in terms of rooting the coming changes in the organization at all levels.

Furthermore the usage of a lightweight triangulation approach to find improvement issues has a moderate price tag attached to it, as well as being an alternative to a more extensive evaluation process. This makes the use of a lightweight approach beneficial for SME:s. For a large enterprise the cost of using the lightweight approach would probably be higher. This is based on the assumption that the studies would be larger and the analysis more resource consuming, unless it is applied to a part of the enterprise. The investigation presented here took about 150 person-hours to complete.

The main risk in using the lightweight approach described in this paper is that issues may escape identification. This in turn can cause critical issues to remain unidentified and unattended. However, two things alleviate this risk. One, the identification of the dependencies between the issues was mapped, establishing an understanding of the interrelations of the issues in question. These dependencies help establish that there is not any crucial step missing as the issues are identified, as well as being a useful tool for the planning of the process improvement itself.

Second, following a process improvement plan, and the execution of such a plan, the studies and triangulation can be repeated. This would hopefully yield confirmation that the addressed issues were turned into non-issues, as well as identifying new improvement issues. The new issues could be issues not prioritized before, but issues missed the first time around could also be caught.

Future work consists of taking the next step, i.e. ascertaining what improvement issues are to be addressed first. This can be done using several different approaches. One is basically to base the priority of the issues on the relations in the dependency diagrams. Another is taking advantage of the expertise present in the organization that has been evaluated, i.e. making the decision up to the same people that were involved in the studies, and letting them prioritize the issues using some prioritization scheme, e.g. AHP.

Process improvement is an ongoing process with several steps, but the first one should be to identify what to improve.

7 Acknowledgements

The investigation described in this paper was conducted at DHR, in close cooperation with project and line personnel, as well as management. We would like to thank all involved for their help, patience, and enthusiasm for new ideas as well as seeing possibilities and not obstacles.

8 References

[1] I.K. Bray, *An Introduction to Requirements Engineering*, Addison-Wesley, Dorset, UK, 2002.

[2] R. Glass, *Software Runaways*, Prentice Hall, Harlow, UK, 1998.

[3] http://sme.cordis.lu/home/index.cfm, 2003-05-05.

[4] C. Robson, *Real World Research – Second edition*, Blackwell Publishers Ltd., Oxford, UK, 2002.

[5] I. Sommerville and P. Sawyer, *Requirements Engineering – A Good Practice Guide*, John Wiley & Sons, Chichester, UK, 2000.

[6] http://www.comp.lancs.ac.uk/computing/research/cseg/projects/reaims/index.html, 2003-05-05.

[7] The TickIT Guide – *Using ISO 9001:2000 for Software Quality Management System*, Construction, Certification and Continual Improvement, Issue 5.0, 2001.

[8] L. Bratthall and M. Jorgensen, "Can you Trust a Single Data Source Exploratory Software Engineering Case Study?", Kluwer, Netherlands, March 2002, pp. 9-26.

[9] B. Kitchenham and L. Pickard, "Case Studies for Method and Tool Evaluation", IEEE Software, IEE/IEEE, July 1995, pp. 52-62.

[10] I. Sommerville, *Software Engineering - 6th Edition*, Addison-Wesley, Essex, UK, 2001.

[11] T.B. Gorschek, K. Tejle, and M. Svahnberg, "A Study of the State of Requirements Engineering in Four Industry Cases", Proceedings of SERPS02, ISSN 1103-1581, Karlskrona, Sweden, 2002, pp. 111-119.

[12] C. Wohlin, P. Runeson, M. Höst, M.C. Ohlsson, B. Regnell and A. Wesslén, *Experimentation in Software Engineering – An Introduction*, Kluwer Academic Publishers, Boston, USA, 2000.

[13] G. Kotonya and I. Sommerville, *Requirements Engineering Processes and Techniques*, Wiley & Sons, Somerset, UK, 2001.

[14] S. Robertson and J. Robertson, *Mastering the Requirements Process*, Addison-Wesley, Harlow, UK, 1999.

[15] H. Sharp, A. Finkelstein, G. Galal, "Stakeholder Identification in the Requirements Engineering Process", Database and Expert Systems Applications, IEEE Computer Soc., 4/1999, pp. 387-391.

[16] K.S Hanks, J.C. Knight, and E.A. Strunk, "Erroneous Requirements: a Linguistic Basis for their Occurrence and an Approach to their Reduction", Proceedings. 26th Annual NASA Goddard Software Engineering Workshop 2001, IEEE Computer Soc., 2002, pp. 115-119.

[17] C. Rupp, "Requirements and Psychology", IEEE Software, IEE/IEEE, Issue 3, 2002, pp. 16-18.

[18] C. Potts, "Metaphors of Intent", Proceedings of Fifth IEEE International Symposium on Requirements Engineering, ISSN 1090705x, IEEE, 2001, pp. 31-38.

[19] I. Sommerville, T. Rodden, P. Sawyer, R. Bentley, and M. Twidale, "Integrating Ethnography Into the Requirements Engineering Process", Proceedings of IEEE International Symposium on Requirements Engineering 1993, IEEE Computer Soc., 1992, pp. 165-173.

[20] B. Potter, J. Sinclair, and D. Till, *An Introduction to Formal Specifications and Z – Second Edition*, Prentice Hall, Glasgow, UK, 1996.

[21] J. Karlsson, C. Wohlin, and B. Regnell, "An Evaluation of Methods for Prioritizing Software Requirements", Information and Software Technology, Issue 14-15, Elsevier, 1998, pp. 939-947.

[22] J. Karlsson, "Software Requirements Prioritizing", Proceedings of the Second International Conference on Requirements Engineering, ISBN 0818672528, IEEE Computer Soc. Press, 1996, pp. 110-116.

[23] T. Saaty and L. Vargas, *Models, Methods, Concepts & Applications of the Analytic Hierarchy Process*, Kluwer Academic Publishers, Boston, 2001.

[24] B. Regnell, M. Höst, J. Natt och Dag, P. Beremark, and T. Hjelm, " An Industrial Case Study on Distributed Prioritisation in Market-Driven Requirements Engineering for Packaged Software", Requirements Engineering, issue 1, Springer, 2001, pp. 51-62.

[25] K. Kivisto, "Roles of Developers as Part of a Software Process Model", Proceedings of the 32nd Annual Hawaii International Conference on Systems Sciences, IEEE Computer. Soc., 1999, pp. 1-19.

[26] W. Lam, "A Case-study of Requirements Reuse through Product Families", Annals of Software Engineering, issue 5, 1998, Kluwer, 1998, pp. 253-277.

[27] O. Roudies and M. Fredj, "A Reuse Based Approach for Requirements Engineering", ACS/IEEE International Conference on Computer Systems and Applications, IEEE Computer Soc., 2001, pp. 448-450.

[28] O. Gotel and A. Finkelstein, "Extended Requirements Traceability: Results of an Industrial Case Study", Proceedings of the Third IEEE International Symposium on Requirements Engineering, IEEE Computer Soc., 1997, pp. 169-178.

[29] B. Ramesh and M. Jarke, “Toward Reference Models for Requirements Traceability”, IEEE Transactions on Software Engineering, issue 1, IEE/IEEE, 2001, pp. 58-93.

[30] J. Calvo-Manzano Villalón, et al., “Experiences in the Application of Software Process Improvement in SMES”, Software Quality Journal, issue 3, Kluwer, 2002, pp. 261-273.

9 Author CVs

Tony Gorschek,

T. Gorschek is a PhD student in software engineering at the Department of Software Engineering and Computer Science at Blekinge Institute of Technology in Sweden. Prior to this, he has worked as CTO (chief technical officer) for a system development company, and later as a consultant in the area of systems engineering and adaptation. He has a M.Sc. in Computer Science from Blekinge Institute of Technology. His research interests mainly include software process improvement and requirements engineering.

Dr. Claes Wohlin

Dr. Wohlin is a professor of software engineering at the Department of Software Engineering and Computer Science at Blekinge Institute of Technology in Sweden. Prior to this, he has held professor chairs in software engineering at Lund University and Linköping University. He has a Ph.D. in Communication Systems from Lund University. His research interests include empirical methods in software engineering, software metrics, software quality and systematic improvement in software engineering. Claes Wohlin is the principal author of the book "Experimentation in Software Engineering – An Introduction" published by Kluwer Academic Publishers in 2000. He is co-editor-in-chief of the journal of Information and Software Technology published by Elsevier. Dr. Wohlin is on the editorial boards of Empirical Software Engineering: An International Journal and Software Quality Journal. He is a member of IEEE and ACM.

SPI Case Study in a Direct Response Marketing Company

Béatrix Barafort, Jean-Philippe Bodelet
Centre de Recherche Public Henri Tudor

Abstract

This paper relates the progress of a software process improvement project based on the ISO/IEC 15504 standard in a major international direct response marketing company. This project is a pilot project that concerns four processes of the organization: *documentation*, *supply*, *requirements elicitation* and *project management*. The entire project was mentored by the Centre de Recherche Public Henri Tudor that performed it in the context of a more global SPI research project.

This paper describes the three main steps of the SPICE approach that was developed such as process selection, process assessment and process improvement implementation. At the time of the writing of this paper, the third step was on the right track but not fully completed. The structure, the organization, the activities and the main deliverables of the project are described. The main difficulties that occurred in this particular project, and how they have been overcome, are explained throughout the lessons learned at each step. Finally the paper relates the outstanding events of the project, the key success factors of such a software process improvement project and re-usable knowledge for the IT Community.

Keywords

External project framing. Process assessment, Process improvement, Process project structure, SPICE, Software Process Improvement (SPI), SPI experience feedback.

1 Introduction

The Centre de Recherche Public Henri Tudor[1] located in the Grand Duchy of Luxembourg, is a Public institution where 220 engineers work on several main domains of expertise: Management & IT Technologies, Industrial Management & Technologies, and Environmental Technologies. The Centre Henri Tudor especially focuses on applied research and technology transfer, addressing most economical sectors and actors over the Grand Duchy of Luxembourg, through the setting and animation of specialized networks. It has been promoting information technologies as one of its key technologies. For this it relies on its CITI department (Centre for IT Innovation) which currently has seventy R&D engineers.

Two years ago, the CITI was contacted by a company named La Redoute, in order to assist it in an IT quality project. La Redoute is the leader of the direct response marketing on the French market and is present on several international ones (Belgium, Portugal, UK, USA, etc). La Redoute belongs to the Pinault Printemps Redoute (PPR) group with a staff of 6200 employees. Within the Information System Department (DSI) of La Redoute, there are 200 people distributed between two teams : Production and Development. The Production team ensures the daily work of customer services and its costs optimization. The Development one develops specific software applications on IBM Mainframe, and it installs and maintains software packages.

At the time of the Year 2000 and the Euro upgrade, DSI had decided to perform the overhaul of its system. Further to the definition of new objectives from the senior management due to this overhaul (customer satisfaction, cost control, pro-activity, customer relationship and bid management), the DSI had to adapt its strategy. A situation analysis had shown that [1]:

- Methods and practices were not used in a homogeneous way and controls were insufficient;
- Customer relationship management (requests, changes, incidents) was efficient at the front-office level but the follow-up and taking charge of the requests and incidents by the back-office level were performed in an unequal way and without commitment;
- The high level Mainframe environment knowledge does not benefit to the structuring in departments within La Redoute.

Further to this analysis, the DSI decided to start an IT quality project that concerns services definition and bid management, current practices improvement, processes definition, jobs and responsibilities clarification and activities control.

Development and Production were both concerned by the IT quality project. The Production team was the first to start it and they chose the ITIL[2] approach [2]. ITIL, produced by the CCTA (Central Computer and Telecommunications Agency in UK) in the late eighties, is probably the most comprehensive structured approach publicly available on providing IT services mainly for its service and process orientation, and also for its adaptability in use. Due to this choice, La Redoute had to find a compatible approach for the Development side of the project. Then La Redoute contacted the CRP Henri Tudor so that they could help them determining the best approach for software process improvement. SPICE[3] [3] was selected because it is both process and best practice oriented (as ITIL), and its use is not so constraining and can be gradual.

This paper describes the Development side of the IT quality project until nearly its ending. It begins by explaining how SPICE was selected, and then it presents the SPICE assessment of three DSI processes. Finally the progress of the IT quality project for the Development side is shown as well as some deliverables. At each step of the project, lessons learned are detailed. The paper ends with the outstanding events of the project and the CRP Henri Tudor's point of view in terms of SPI research results that can be transferred to the IT community.

[1] CRP Henri-Tudor – Public Research Centre Henri Tudor. See http://www.tudor.lu

[2] IT Infrastructure Library

[3] Software Process Improvement and Capability dEtermination officially called ISO/IEC 15504

2 Process selection and assessment

2.1 Process selection

2.1.1 Process selection stage progress

The CRP Henri Tudor was invited via the Spiral network[4] [4] to present to some DSI leaders the world of IT quality standards with a special focus on ISO 9001, CMM[5] [5] and SPICE.

ISO 9001 was directly put aside because the objective was not certification and its approach is not software dedicated. CMM was also put aside mainly because its scope is too wide, too global for La Redoute, which turned its attention to specific software development areas. CMM does not enable easily the assessment of a selection of processes. The intention was that this software process improvement project is a pilot project that involves only a selection of the organization's processes. After some weeks of research and thoughts, and a complementary presentation of the overall SPICE processes, La Redoute chose definitively SPICE to be their software development quality framework.

The three main stages in a SPICE approach for a process improvement project were followed: process selection, process assessment and process improvement implementation.

A specific committee was created to carry out the first stage. It took back the objectives defined by the senior management quoted above. It balanced them and defined key success factors. Some SPICE processes were put aside because they were already considered in the Production project (configuration management, operation) or because the context of La Redoute was not appropriate (organization process category).

A mark was given to each process according to its contribution to each strategic objective. The result was a prioritized list of processes and the first three ones were selected for improvement within the development side of the quality project. The three processes were:

- Supply
- Requirements elicitation
- Project Management

La Redoute chose on purpose a restricted number of processes to assess in order to perform an improvement project with clear and limited goals and to implement a smooth process improvement approach within the department.

The supply process was completed with the base practices of the customer acceptance process and one base practice of the acquisition preparation process in order to cover the definition and preparation of the mutual software acceptance.

Some base practices of the risk management process were added to the project management ones in order to increase staff's awareness of this process.

With the SPICE base practices definition as a basis, a reformulation of the practices with the company's vocabulary was also performed. A cartography of the three processes was established in order to give a whole view of them, to position the ones compared to the others and to underline interactions.

The choice of these three processes allowed La Redoute to have a significant pilot area to be assessed and improved.

[4] Network compound by IT professionals. See www.spiral.lu

[5] Capacity Maturity Model

2.1.2 Lessons learned

At the end of this stage, the first difficulty was the process approach. It was not easy to think and act in a process oriented way for unaccustomed people. The process approach is not so obvious to use when you want to pool daily activities around a same objective. The boundaries between the processes are not always easy to define.

The second difficulty was the adaptability of the SPICE processes definition to the context of La Redoute. A big work of vocabulary and sometimes of content adaptation had been made in order to tailor the SPICE processes to the context of the company. The supply process became an aggregation of several base practices from different SPICE processes.

2.2 Process assessment

2.2.1 Process assessment stage progress

In a Software Process Improvement (SPI) project, the Process assessment stage is highly recommended in order to determine the maturity level of the assessed processes, their strengths and weaknesses to objectively identify improvement opportunities.

Three persons of La Redoute were trained by the CRP Henri Tudor to become SPICE assessors [6]. So that when necessary, they were able to reformulate the questions with the company's vocabulary and adapt them to the company context. They were also able to ask for more accuracy when the answers seemed too superficial. The other advantage is that in the future, they can regularly self-assess their own practices and determinate the improving actions to implement.

To begin assessment, a preliminary report was drawn to precise objectives, scope and constraints of this step, size and description of the sample, plan and schedule for the interviews, quality criteria for the assessment process itself, etc.

Pairs of assessors (an internal and an external one) led nine interviews (three by process). To make interviews easier, work products were collected before the assessment and each interviewee handed them during the interview.

To write the final report, assessors proceeded by brainstorming on every process in addition to a systematic work of analysing results. It was a collaborative work. Further to the validation from La Redoute assessors, the final report was presented in a first time to the DSI management and in a second time to all DSI staff.

The main results of the assessment are shown on the following chart (Figure 1) with the assessment profile for the three processes [7]. La Redoute had determined a level 3 target and one attribute of level 4.

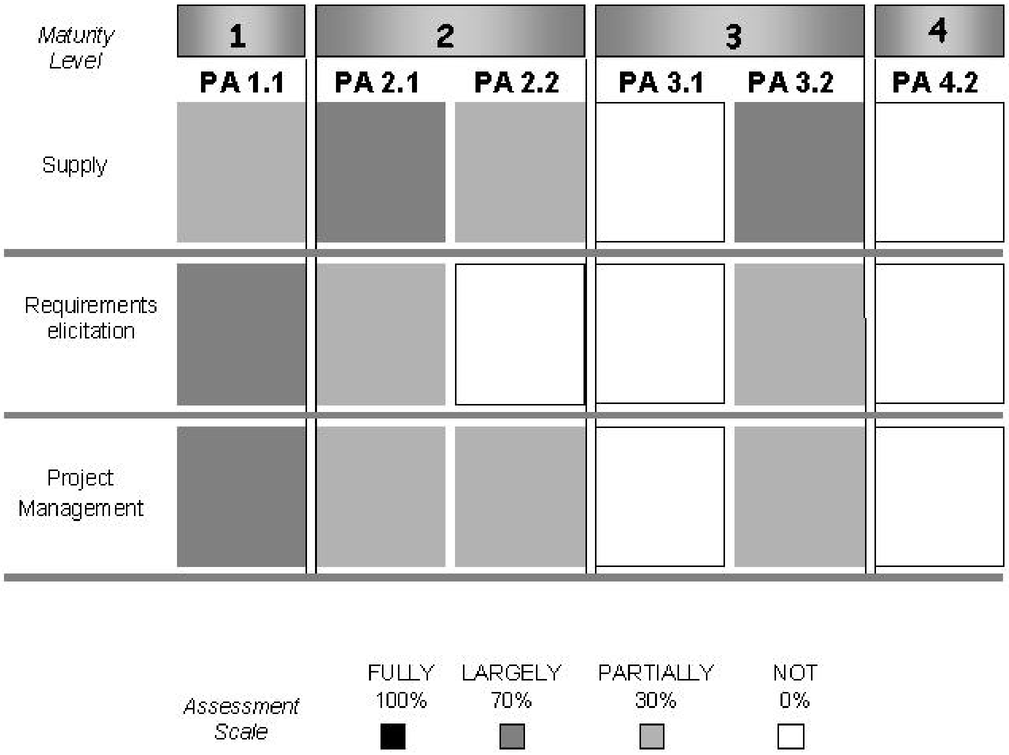


Figure 1 : Assessment Profile - La Redoute – DSI

Results were relatively weak but with a lot of good practices in each process, even if these ones were not used in a homogeneous way. Main gaps are in work product management and process definition. Except for the supply process, the processes reach level 1. It means that project management and requirements elicitation activities are generally performed. Supply process weaknesses are mainly because of a lack of formal contract within IT projects.

At level 2, planning and tracking management practices exist but work product management ones are deficient. For level 3, there is no process definition. Process resource (technology and human resource management) practices exist but are weak.

For each process, a SWOR[6] analysis has been performed with short-term recommendations (quick wins), medium-term recommendations and long-term recommendations (advanced concepts).

The main general quick wins were to establish a documentation process (including version management), to improve access rights management and to define a process definition template by process. The medium-term recommendations aimed at making project leaders aware of a process approach, establishing a common project storage space, to be shared by all IT project actors and setting up documentation and process quality control. Following long-term recommendations, La Redoute was encouraged to involve customers in the software process activities, capitalize experience, organize and perform quality activities.

This was an overview of the recommendations formalized in the assessment. The next chapter will describe the third stage of the SPICE approach: process improvement implementation.

To conclude this part, we can say that there are many good practices in La Redoute, but they are too individual. La Redoute should use these elements as basis and generalize them. According to priorities, the efforts of improvement will have to be more or less important.

In the future, the target for La Redoute is the SPICE level 3.

[6] **S**trengths **W**eaknesses **O**pportunities **R**isks

2.2.2 Lessons learned

The involvment of three La Redoute asssessors had facilitated this assessment step. The most difficult think to do when you prepare a assessment of a company is to adapt the questions to its own context and it's not easy when you come from outside this organization.

This assessor training is also a good point when a company or a department want to begin an IT quality project, those ressources are able to orient the project and to keep it in a specific framework. There are also able to performe middle assessment after some months of work without the help of a specialized external company.

3 Process improvement implementation

3.1 Improvement project definition and organization

After the presentation of the assessment results to the DSI management and staff, the definition of the improvement project began. The goal of the project was to achieve quick wins, medium-term recommendations and general recommendations.

Four sub-projects were set up, one by assessed process plus another one to include the general recommendations throughout the documentation process. A process owner by sub-project was nominated. One year ago, the DSI management authorised the Development side of the IT quality project to be kicked off.

Each sub-project was defined through a project plan based upon the global project plan. Its goals were to improve base practices, to define and formalize the process via a procedure, operating guides and templates, and to set up control management of the process. The generic sub-project life cycle is shown on figure 2.

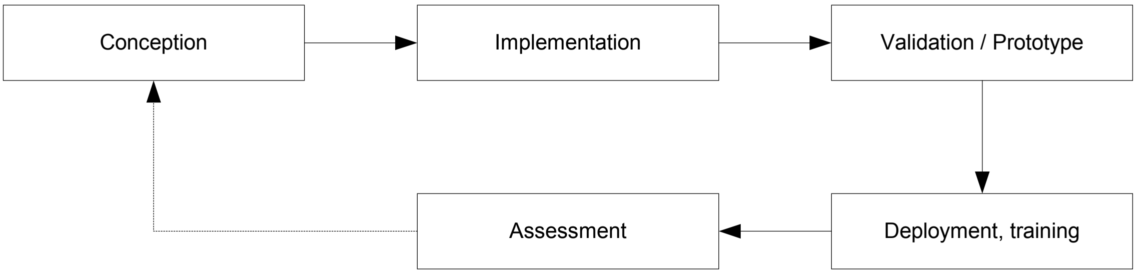


Figure 2 : Process improvement sub-project life cycle

The result of the *conception* is a formal definition (including the SPICE base practices) of the process and its boundary, documentation practices and metrics, which enables work product management, and all necessary tasks to fulfil objectives.

Implementation consists of defining how to perform daily process activities and giving the technical infrastructure to do them. It consists also of setting up a dashboard. The next step is implementation *validation*. Its goal is to make sure that *implementation* correctly enables process activities and to reach process objectives.

Next step covers the *deployment* of the process with training and monitoring of the process users. Three months after the *deployment*, the process should be re-assessed in order to verify its good implementation. Then the project director and the process owner perform the assessment. A global report on the project progress should be made at the same time.

The project director carries out the global project management and coordination tasks. A Global IT quality project Steering Committee deals with the strategic level, giving main orientations, following project evolutions and arbitrating conflicts. It meets bimonthly.

The project director, the process owners, the quality assistant, development department managers and correspondents of some other departments make up the Improvement Process Projects Steering Committee. It deals mainly with the validation of the project evolutions, risks and critical points identification. It communicates also persistent problems to the Global IT Quality project Steering Committee.

The organization of the project is shown on figure 3.

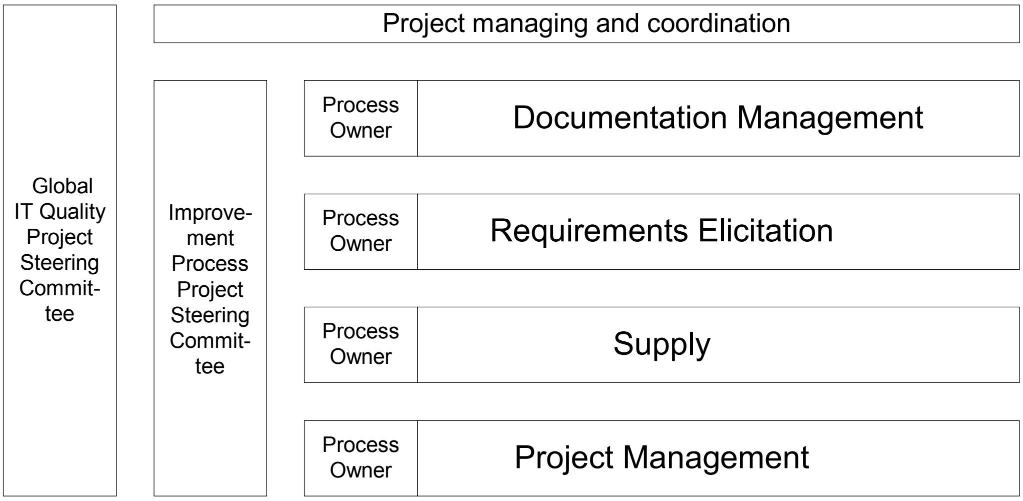

Global IT Quality Project Steering Commit-tee
Project managing and coordination
Improve-ment Process Project Steering Commit-tee
Process Owner
Documentation Management
Process Owner
Requirements Elicitation
Process Owner
Supply
Process Owner
Project Management

Figure 3 : Project Organization

Improvement Process Projects Steering Committee meeting takes place monthly. Every process owner must complete and present an updated dashboard, explain what has been performed (actions, results) during the last month and what is planned to be performed the next months. A CRP Henri Tudor engineer attends all meetings, gives advices and reviews some deliverables. He brings his SPICE expertise and also some documents [8] that help process owners to determine what results should be obtained.

3.2 Project progress

For each process improvement sub-project, a working group has been defined from its beginning. The main part of the improvement work was performed during these working group meetings. They gathered the main actors of the processes (essentially project managers and some customer's correspondents). These groups regularly met. The process owner led the meetings and formalized the results.

Now, for each process, here are the main SPI project results.

3.2.1 Documentation

Project progress

The documentation improvement project is transversal. It produces results that are used in the three other processes and then reduce their workload in avoiding some activities redundancy.

The main target of this sub-project was the definition and the formalization of a general documentation management policy. This one contents a documentation macro lifecycle, which presents all the project documents according to the project phases. In this policy is also defined the documentation best practices and the storage rules adapted to the DSI context. These rules were experimented by the other process improvement sub-projects before a global generalization to all DSI. A thought on some quality control techniques was also led in this project.

Lessons learned

The work done on the macro lifecycle enabled a thought on the utility of some existing documents. Their content was optimized, some old document templates were cancelled or distributed on a different way. Others were created or redefined.

The macro lifecycle was a key element in the definition of the process boundaries. This was very important in order to enhance the process approach and to monitor its deployment. During the project, the working groups often consulted it to know exactly what was their sphere of activity.

The working group for the documentation management process gathered all the process owners plus additional involved people. This made easy information exchanges between processes on the documentation theme.

Experimenting documentation rules by sub-project was an opportunity to show to all DSI what type of practice was developed in the IT quality project. For example, all the sub-project data storage hierarchy was visible by all the DSI. Sub-projects set an example and the staff followed them.

3.2.2 Requirements elicitation

Project progress

The most important recommendation for the requirement elicitation process was the partial and inadequate use of the requirement specification template. It was considered as inappropriate for the type of project that was performed by DSI.

Determinate the useless and missing elements of this template was the first job for the working group. Afterwards it completed and reorganized the document to take into account new elements [9] (NFR[7], responsibilities for DSI and customer,…). For example, some elements of the Volere template [9] were used to structure the requirement specification template (check of content completeness, requirements numbering, structured form by requirement,…).

A part of the project was devoted to the requirements changes tracking. A change notice template and rules of use were defined. The DSI customers who took part in the working group meetings gave their opinion on the building and the formalization of all those documents.

A definition of the entire requirement process structure was also formalized. This one explains all the activities that must be done to obtain a completed requirement specification document with a description by activities of the inputs, outputs and actors.

Finally, the working group has worked on the quality control and the metrics of this process. It was the first among the other process improvement projects to work on metrics and may serve as pilot for others.

Lessons learned

Customer participation was very positive and constructive. Its involvement enabled DSI to obtain better quality results and to improve the DSI-customers relationship.

Among the base practices, the work product management and the process definition attributes for the requirements elicitation process, a big effort has been made on documentation structuring. The next step will be to work on the method for elicitating requirements.

[7] Non-Functional Requirements

3.2.3 Supply

Project progress

Due to the supply process assessment results, the working group decided to focus on the bid management aspects (from the first contact with the customer to the contract signature).

Still more than for the requirements elicitation process, DSI customers were involved in the project. They particularly worked on two specific documents, a pre-study document and a bid template, plus several forms such as project launch, project stop, project amendment.

The pre-study document, created before the requirements specification, includes first costs and workload estimates. This document is the first official commitment between the two parties (customer / DSI). The Bid template defines formally the relationship between the two parties. It describes the answer of the DSI against the customer requests (formalized in the Specification document). This Bid document is still an “internal” contract between two La Redoute departments (DSI and one or several customer departments). This will prepare the future when DSI probably become an IT service provider for external organizations within the PPR group.

Lessons learned

Adding base practices to the supply process brought some problems during the project because DSI is considered as a supplier for several La Redoute internal users groups, and the additional practices concerned the customer (customer acceptance process and acquisition preparation process). Actors (in the working group) did not always know exactly how they had to position themselves on the activities of the process.

3.2.4 Project management

Project progress

Before this improvement project, La Redoute had defined a software development cycle but it was adapted for IT reengineering projects and not for other types of project. So project managers used some parts of this document according to their needs and to all types of projects.

Project management working group decided to update the software development cycle document to respond to other types of projects. But to reach this target, it began by defining a project typology [10] that classifies all La Redoute projects according to some characteristics. This typology is the base of a project evaluation table. This table gives the project profile with its appropriate software development cycle. In the future, the goal is collecting and capitalizing information by type of project, which could help in scheduling, risk management, workload estimate, … Six project types and dedicated software lifecycle were defined.

The working group also thought to a risks-checklist with the idea of making DSI employees aware on this theme. A more efficient tool might be developed later in the context of a Risk Management improvement project.

Lessons learned

The project evaluation table was a good idea to determinate the best project lifecycle but it is not easy to set up a tool that support it. To have a reliable result, characteristics must be exhaustive and easy to assess. An Excel table enabled a simple automation.

From now on, the project management process gives the opportunity to link all the deliverables of the IT quality project, to position them ones compared to the others via connected documents. It enables the structuring of all the work products in a kind of toolbox, it is the unifying thread of all the other improved processes.

At the beginning of the SPI projects, many people made mistakes by unconsciously incorporating activities to the project management process on a wrong way. The macro-project lifecycle helped to better understand the proper project management practices.

4 Outstanding events

In this section, an overall analyse of strengths and weaknesses is made for the Development side of the IT quality project by considering outstanding events of the project.

At the beginning of the project, one of the main goals was to implement quick wins (quick results: strong added value for limited effort), to keep up the project staff motivation and show the usefulness of the project. Among these quick results were the exact definition of the scope of main documents, and documents version management, etc.

In spite of these elements, some process projects had a period of light productivity fall essentially due to a lack of resource availability because more DSI projects had taken the priority on this one and had forced DSI to put it aside. But this problem did not affect the entire project; only a few groups took more time to produce their deliverables, but made up for it in the end.

The parallel use of ITIL and SPICE enabled to enforce one standard with the other. For example, on the one hand, some SPICE processes were put aside from the process selection because they were better covered by ITIL and the Production quality project. On the other hand, some SPICE elements were used to complete the process description document template performed by Production working groups, which was based on ITIL. Those two IT quality standards seemed to be perceived as complementary tools [11].

Adding base practices from some processes to complete another process is something to do carefully. You should pay attention to the nature of the additional practices, it must be the same as the nature of the original process. A risk of bad process boundaries can occur.

The main change brought to the DSI by this project was the process approach. At the beginning, it was very difficult for DSI members to think in a process oriented way. They rather thought in a project oriented way. The process description document, which was mentioned before, contributed in spreading the process approach to the DSI staff. But this difficulty generated an important organizational work (setting up a process owner, adapting the project organization to the processes,...).

The monthly follow-up of an external expert allowed improvement project members to set up regular checkpoints and verify the accomplished work. Also this meeting allowed them to precisely determine what was to be done during the next month. This external view on the project was a good way to make the scheduling of the concrete goals and the checking of their achievement easier.

In terms of communication, the SPI project managers had decided to publish a newsletter that explains monthly the project progress to all the DSI staff. These means enable firstly to communicate with the DSI staff and secondly to involve them in the process so that it reduces their resistance to change. This newsletter plus regular presentations of SPI project deliverables with new forms and documents (with directions for use) contributed to the change management of concerned staff.

4.1 CRP Henri Tudor’s point of view

The CRP Henri Tudor participated all along this SPI project in La Redoute. These mentoring services were performed in the context of a research project named SPINOV (Software Process Improvement and inNOVation). The national ministry of Culture, High Education and Research in Luxembourg funds the SPINOV project. Its main objectives are to develop competences within two fields :

- Requirements engineering;
- Software process assessment and improvement.

Both fields follow a process approach. The competence development is made throughout several activities such as thematic and normative watch, training, methodological guides formalization, process package documentation (for instance for processes such as requirements elicitation, documentation, configuration management, quality assurance), dissemination (SPI conference publications, thematic presentations for IT professionals, training courses…).

Within the SPINOV project, there are particular activities related to SPI that brought mutual benefits to La Redoute and CRP Henri Tudor. Firstly, participating to a normative expert group of AFNOR[8] in France enables to follow the ISO/IEC 15504 standard evolutions. Among these evolutions, the refinement of some processes of the ISO/IEC 15504 process assessment model fitted with the La Redoute tailoring of the Supply process (added base practices).

Within the SPINOV project, there were some works for defining guidelines for requirements management (non functional requirements checklist, use case formalization guide,…) that were useful for La Redoute (also the mapping of the Volere requirements template with the La Redoute Specification document redefinition).

Finally, in terms of dissemination, there were several events in Luxembourg such as an ISO/IEC 15504 and SPI concepts awareness presentation made in a professional exhibition (October 2002), a presentation of La Redoute improvement approach in the SPIRAL[9] conferences [1] (November 2002), and more recently a SPIRAL thematic meeting about requirements engineering, with a specific presentation of the improved requirements elicitation process of La Redoute. All these events are contributing to the SPI best practices spreading with lessons learned focus.

5 Conclusion

At the time this paper was written, the IT quality project of La Redoute was on the right track, the deliverables had been achieved and the work of the several groups had been serious and constant. This project covered only four processes of the organization and it can be considered as the first part of a more important process improvement project. Indeed, in 2004, La Redoute foresees to allocate new budget to include other processes in the improvement approach. It will be based on the actual SPI structure.

It is important to underline the fact that if this SPI experience was good, it was mainly due to the efficiency of the setting-up structure that organized and managed the project. It supported perfectly all activities and did not hesitate to communicate on the project. Also, it had the total support of the La Redoute management and this last point is the most important condition to make a success story out of this project.

The CRP Henri Tudor contribution has been considered positive and stimulating. It brought guarantees of the compliance with the SPICE standard. This collaboration enabled the SPI actors to be informed on the state of the art in the domain and so to do an adequate work. The very pragmatic SPI work performed by La Redoute addressed their needs with common sense and structuring. This success story and the research project run by CRP Henri Tudor reinforce the SPI knowledge for the IT community with re-usable results such as thematic presentations, SPI state-of-the-art synthesis, methodological guides, document templates with directions for use,…

[8] Association Française de NORmalisation

[9] IT professional network in the Grand-Duchy of Luxembourg, the Walloon region in Belgium and Lorraine in France

6 Literature

[1] Caplier C., Utilisation de référentiels qualité pour la mise en œuvre d'une démarche d'amélioration des pratiques informatiques, in : L'application de standards en service management, développement, gestion de projet, Conférences SPIRAL 2002, Luxembourg.

[2] IT Infrastructure Library – Service Support, Service Delivery, The Stationery Office Edition, 2000.

[3] ISO/IEC JTC 1/SC 7, ISO/IEC TR 15504, 1998.

[4] SPIRAL*NET ESSI ESBNET Project Number 27884 - Final Report, March 2001

[5] Mark C. Paulk, Bill Curtis, Mary Beth Chrissis and Charles Weber, Capability Maturity Model for Software, Version 1., SEI, CMU/SEI-93-TR-24, Pittsburgh, Pa, Feb. 1993.

[6] Compita Ltd, Process Professional Process Portfolio, Process Professional Library Services, 1996

[7] Barafort B., Di Renzo B., Merlan O., Rapport d'évaluation détaillé *La Redoute* – DSI, CRP Henri Tudor, 2002.

[8] Barafort B., SPIRAL Network development results, in: Proceedings of the International Conference EuroSPI'2001, Limerick, Ireland, 2001.

[9] Volere Requirements Specification Template, see http://www.volere.co.uk.

[10] Joly M., Muller J-L G., De la gestion de projet au management de projet, AFNOR, 1994.

[11] Baraford B., Di Renzo B., Merlan O., Benefits resulting from the combined use of ISO/IEC 15504 with the Information Thechnology Infrastructure Library (ITIL), in: Proceeedings of 4th International Conference on Product Focused Software Process Improvement, Rovaniemi, Finland, 2002.

7 Author CVs

Béatrix Barafort

Béatrix Barafort graduated in 1996 as a Software Engineer in the Conservatoire National des Arts et Métiers in Lyon (France). From 1988 to 1996, she has worked in a software house in Lyon as a programmer, software analyst, and then project leader for development projects in banks and insurance companies. Then she joined the Centre de Recherche Public Henri Tudor in Luxembourg for working in the CITI department within software process improvement projects. She coordinated several process assessment (SPICE) projects and improvement programs. She is a SPICE Qualified Assessor (certificate of achievement of "Process Professional Assessment"). She was the project leader of the SPIRAL*NET ESSI ESBNET Project (27884). This project was aiming at optimizing and generalizing software best practices in the Customer/Supplier Processes Quality Management. In this project, she led SPI actions such as micro-assessments, SPICE assessments, software engineering working group leading, definition and leading of a SPI training cycle, SPI monitoring actions for companies, publication and results dissemination throughout the SPIRAL network (IT professionals dedicated network covering the Grand-Duchy of Luxembourg, the Walloon area in Belgium and the Lorraine in France) and in SPI international conferences. She is now the head of a Research Unit related to Reference Systems for Certification and Modelling and she is leading a research project (national funding) aiming at developing requirements engineering activities and at enhancing software process assessment and improvement practices.

Jean-Philippe Bodelet

Jean-Philippe Bodelet has studied Computer Science at the Facultés Universitaires Notre Dame de la Paix in Namur (Belgium). He graduated in 2002 and joined the Centre de Recherche Public Henri Tudor in Luxembourg for working in the CITI department within two software process improvement projects. The first one called Qualinnove is aiming at elaborating and developing a project management guide for innovation projects. This guide contains a group of more reactive and proactive practices than classical project management practices. It gives an important role to risk management and change management activities. The second one called Spinov is aiming at developing requirements engineering activities and at enhancing software process assessment and improvement practices.

Making SPI Happen: Iterate Towards Implementation Success

Anna Börjesson, Ericsson AB and IT University of Gothenburg, Sweden

Abstract

Software Process Improvements (SPI) has been used to enhance the capability of software organizations for more than one decade, but software organizations still suffer from the high rate of SPI failures. Research has focused on this problematic situation and several new approaches to SPI have been introduced to improve the failure rate. SPI implementation has, however, not yet been sufficiently explored to address the difficulties that software organizations experience when introducing software processes into software practice. This paper explores 18 different SPI initiatives within Ericsson focusing on the relation between the number of iterations executed within each initiative and the SPI implementation success. This research explores patterns between the number and nature of iterations executed, the organizational responses to change, and SPI implementation success. The experiences from Ericsson indicate that the number of iterations executed has a significant impact on implementation success. The findings are contrasted and discussed in relation to the literature on SPI and change management.

Keywords
Software Process Improvement, SPI Implementation Success, Iterative Process Development, SPI Success Factors

1 Introduction

Since the late 80's when Watts Humphrey (1989) introduced Software Process Improvements (SPI) many software organizations have spent huge efforts to achieve SPI success. Success stories have been reported (Diaz and Sligo 1997; Hayley 1996; Humphrey et al. 1991), but critical evaluations have also been published (Bach, 1995; Bollinger and McGowan, 1991; Fayad and Laitinen, 1997; Humphrey and Curtis, 1991). Several new approaches to SPI have been introduced (McFeeley,1996; Abrahamsson, 2000; Aaen, 2002; Arent, 2000) during the last years to understand and decrease the SPI failure rate, but there are still areas that have not been fully explored for us to understand the barriers and enablers for successful SPI.

SPI implementation success is a first and necessary step towards SPI success (Börjesson and Mathiassen, 2003a; Börjesson and Mathiassen, 2003b). This paper explores and contrasts 18 SPI initiatives carried out at Ericsson AB in Gothenburg, Sweden, during the period 1998-2001 focusing on the relation between number of executed iterations and SPI implementation success. All SPI initiatives were similar in terms of SPI goal, management attention, engineering environment and SPI resources but they differed in number of executed iterations within the SPI initiative and also in SPI implementation success.

The experiences from Ericsson suggest, as we shall see, that it is important to execute several iterations within an SPI initiative to assure implementation success. SPI Implementation has many connections to the change management area and known models like the Satir Change Model (Weinberg, 1997) and the Development of Commitment to Change Model (Connor and Pattersson, 1982). These models are used to explain and discuss the experiences from Ericsson and in particular one of the SPI initiatives that executed several iterations. These theoretical interpretations help us understand what happened during the iterations. This analysis indicates that the number of iterations executed within the SPI initiative have a significant impact on the SPI implementation success. The theoretical context is described in chapter 2 and the research approach is presented in chapter 3. Chapter 4 describes the organizational context and the data from the Ericsson cases. Chapter 5 relates the experiences from Ericsson to SPI implementation success and discusses them in relation to the SPI and change management literature. The argument is concluded in chapter 6.

2 Theoretical Context

There are several approaches described in the literature to iterative SPI (McFeeley, 1996; Grady, 1997). Grady describes an iteration using four activities (Plan, Do, Act and Check) while McFeeley describes an iteration in four different, but similar activities (Diagnose, Establish, Act and Learn). McFeeley also has an initial activity, Initiate, that starts the iterations. Both these approaches include similar detailed activities, but they are framed somewhat differently. Both approaches discuss the importance of commitment to assure successful SPI, but they frame the activities for how to reach commitment differently. Both approaches also end up in activities that aim to learn from the executed work, but while McFeeley's approach ends up with proposed future actions that will be handled in the next iteration, Grady's approach includes revising the results from the first iteration. Both approaches have, however, the same intention to execute improvements in a stepwise fashion to learn from previous steps before taking the next.

Taking an iterative approach to successful SPI was initiated already in the late 80's by Watts Humphrey (1989). Humphrey does not use the word iteration, but he indirectly discusses the need of performing SPI work in steps and repeatable sequences. Humphrey points out "The Six Basic Principles" of software process change and one of them is "Change is continuous. Software process improvement is not a one-shot effort; it evolves continual learning and growth". Humphrey also states "Improvements should be made in small steps". He argues that people do not naturally perform well on highly rigorous and complex tasks. The software engineering process must therefore be tailored to their abilities in small steps. Humphrey (1989) is not explicit regarding iterative SPI work, but he initiated and discussed some of the basic ideas that iterative development is built on.

Grady (1997), McFeeley (1996) and Humphrey (1989) agree to stress commitment to SPI as a way to overcome resistance to change. Grady states the need of understanding the origins of resistance to change. McFeeley argues that change efforts of a certain magnitude cause substantial fear and uncertainty throughout the organization. Knowledge of change management is therefore needed by both SPI drivers and practitioners. Humphrey stresses that enthusiastic, technically and politically capable and dedicated resources with management's confidence are necessary to reach successful SPI implementation. Humphrey calls these dedicated resources change agents. "If SPI is not anybody's job, it is not surprising that it doesn't get done", argues Humphrey. Dedicated resources are needed to assure that practitioners overcome resistance, that practitioners are provided with adequate training, and that projects receive the necessary consultation.

Resistance to change is explored and described by Weinberg (1997). Weinberg's Satir Change Model (see Figure 1) describes four different phases that a change initiative passes through to reach higher performance. The first phase "Status Quo" is distinguished by no change in performance. The second phase "Chaos" is distinguished by performance going irregularly up and down. In the third phase, "Integration and Practice", there is a take-off to wards higher performance and the fourth and last phase is distinguished by higher performance, i.e. “New Status Quo”.

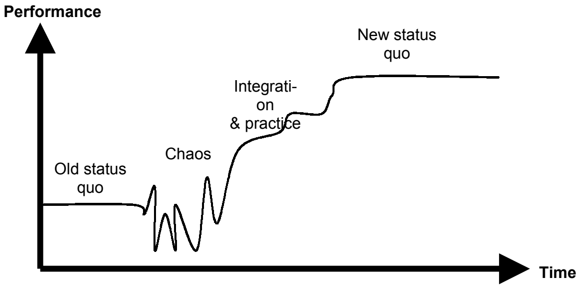


Figure 1. The Satir Change Model (Weinberg, 1997)

Weinberg also describes the Satir Change Model from an iterative perspective (see Figure 2). He shows how each activity to reach new status quo can fall back to old status quo. The first two activities can return directly back to old status quo, where the next three activities can lead back to chaos before eventually achieving a new status quo. This model describes a number of possible events and actions that can occur during a change initiative.

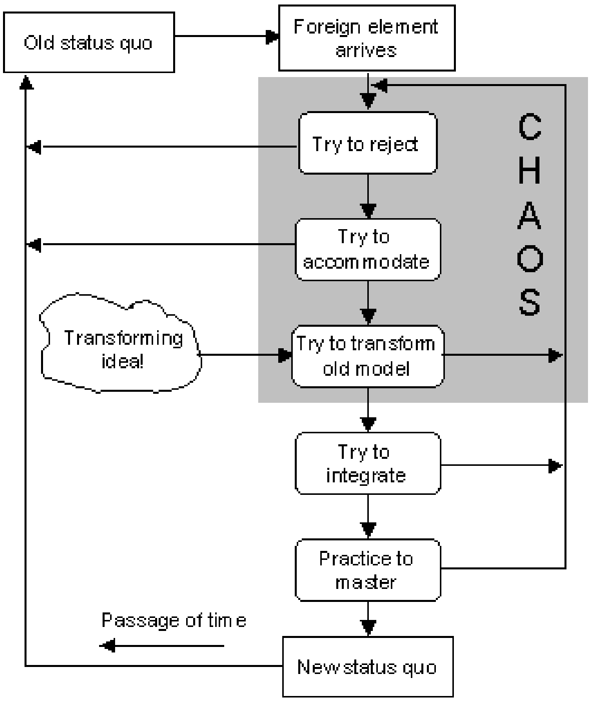


Figure 2. Weinberg’s Change Model

Another similar model (see Figure 3) that explains reactions to change is Development of Commitment to Change by Connor and Patterson (1982). This model describes how the degree of support changes over time during a change initiative. The degree of support to change goes from preparation and acceptance to commitment. The time scale is divided in phases from unawareness, confusion, negative perception, decision not to support, change aborted after initial utilization, and change aborted after extensive utilization. Each phase can also end up in returning to a low degree of support to the change.

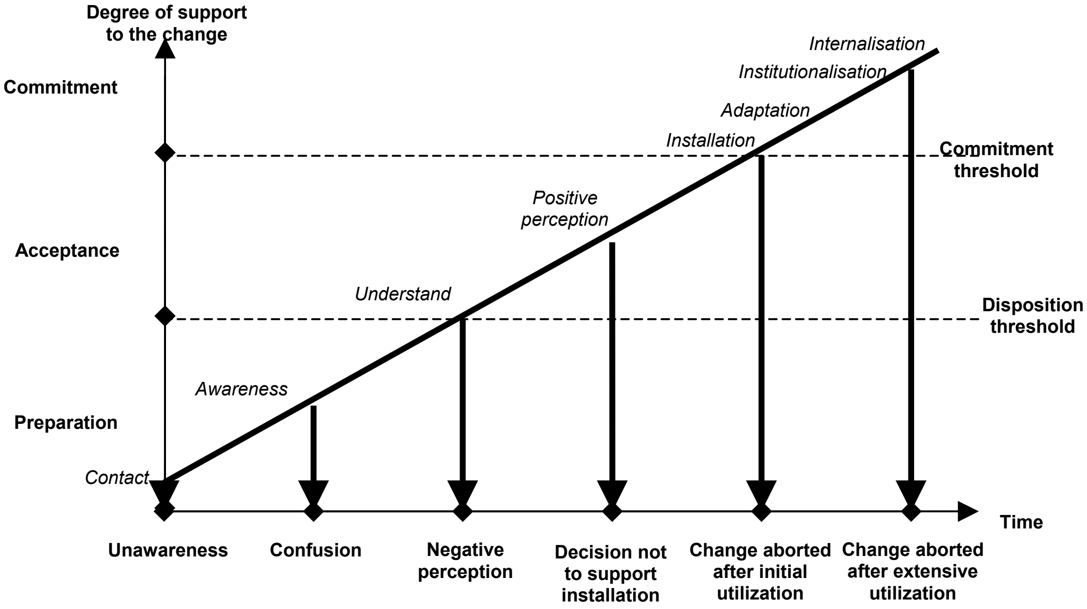


Figure 3. Development of Commitment to Change (Connor and Patterson, 1982)

Both Weinberg's (1997) and Connor and Patterson's (1982) models describe how a change initiative passes different phases to achieve higher performance. They also show a number of possible events that cause the change initiatives to fall back to the initial status in performance and commitment. The models are framed differently but the underlying understanding of implementation and change processes is similar. Weinberg's model describes the connection to higher performance over time, while Connor and Patterson's model describes how commitment to change evolves over time as a key driver towards successful SPI (Humphrey, 1989; Grady, 1997; McFeeley, 1996). Both models suggest that implementation is a challenging process with many obstacles on the way and a considerable likelihood for not successfully achieving change.

Commitment to SPI has also been explored and discussed by Abrahamsson (2001). Abrahamsson identifies a number of misconceptions in current thinking regarding the commitment concept. One of the misconceptions is the controllability of the development of commitment. Trying to get someone committed is not possible before that someone has experience within the area in question. The reactions to change as described in the Satir Change Model (Weinberg, 1997) and Development of Commitment to Change (Connor and Patterson, 1982) are not valid in all cases. The phases in the models will differ depending on experience, context, and engineering practice in question. Such models need therefore to be interpreted and adopted to specific situations before they can be used for guiding SPI management.

The commitment process is vital for successful SPI, but it needs constant attention and long-term thinking (Aaen et. al, 2000). Managers and practitioners may become so dedicated to solve a certain problem that the long-term perspective gets lost and the solution becomes sub-optimized. Aaen et. al. also discuss the possibility of getting too focused on formal goals of reaching a certain maturity level. It is important to have long-term management commitment to change initiatives and to constantly adjust goals and hence commitment as the process unfolds. It takes several years to bring an organization from CMM level 1 to CMM level 5. Long-term changes are delicate to manage even for the most sophisticated managers.

3 Research Approach

This research focuses on process implementation by contrasting successful and less successful SPI initiatives. The purpose is to learn more about key factors that enable or create barriers for successful process implementation. An interpretive, multi-case study (Galliers, 1992; Yin 1994) is well suited for that purpose. The research is part of a collaborative practice study (Mathiassen, 2002) carried out at one of Ericsson's system development centers with more than 20 years of experience developing packet data solutions for the international market. The organization has grown from 150 employees in 1995 to 900 in 2001 and SPI has become an increasingly important area in order to assure quality deliveries.

The author has been working in and been responsible for the initiatives. The potential bias and subjectivity is handled through discussion and interviews with engineers and researchers both within Ericsson and from the research community. The study is interpretive (Galliers, 1992) and action based (Yin 1994) with a focus on process implementation.

SPI outcomes are ideally measured in improvement success, i.e. differences in quality and productivity between old and new practices (Humphrey, 1989; Grady, 1997). Improvement success is, however, difficult to measure without comparable data from longitudinal studies. This research focuses on the extent to which initiatives lead to changes in software practices. This notion of implementation success is easier to measure and validate and it can be considered an important prerequisite and indicator for improvement success (Börjesson and Mathiassen, 2003a; Börjesson and Mathiassen, 2003b).

The SPI unit collected the basic data during the initiatives with help of time registration reports, SPI initiative final reports, and SPI project specifications. The evaluations are based on interviews and discussions with software practitioners, project managers and SPI initiative members and carried out in collaboration with those involved in the initiatives. The data are presented in Table 1 and the Appendix and they have been verified by the involved practitioners. The data from the detailed study have been collected through documentation from the SPI initiative but mainly from discussions and interviews with practitioners participating in the SPI initiative. It is important to stress that these data were not obvious during the time when the initiative was executed. Plans existed, but were not fully communicated at all times to everyone involved. The data from the detailed case are based on the author's extensive participation, plans and documents from the initiative, and discussions with practitioners participating in the SPI initiative.

4 Presenting Data from 18 SPI initiatives

An iteration is a repeatable sequence of activities. To compare and contrast iterations across different SPI initiatives, we need a baseline of what activities that are a part of an iteration. The baseline used in this paper is the IDEAL model (McFeeley, 1996). Figure 4 shows the phases and activities included in the IDEAL model. The baseline we will use when explaining an iteration goes from Initiating (only first iteration), Diagnosing, Establishing, Acting to Learning.

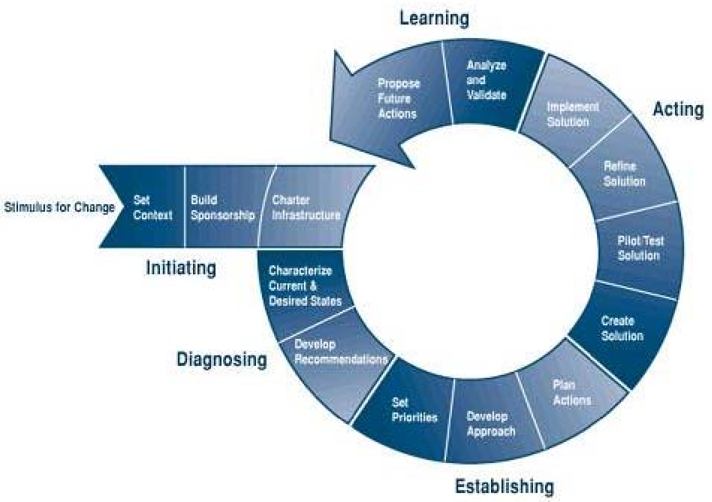


Figure 4. The IDEAL Model (McFeeley, 1996)

In Table 1, where a summary of the data is shown, the column '# of iterations' describes how many iterations that were executed within each SPI initiative. Each '# of iterations' is described by X.Y, where X is the number of full iterations executed and Y is the number of the activity where the initiative stopped in the last iteration (0 – no new iteration started, 1 – Diagnosing, 2 – Establishing, 3 – Acting, 4 – Learning). When "# of iterations" is for example 2.3, it indicates that two full iterations where completed and the SPI initiative stopped somewhere during the acting phase of the IDEAL model.

4.1 The Overview Study

All initiatives were planned and executed under similar conditions. They addressed the same engineering operation, they were part of Ericsson's intensive SPI program, they had the same strong level of management attention, they had the same level of resources, and many of the SPI practitioners were the same. Table 1 provides a summary of the data from the 18 cases. For each case we present the following data:

- #: Unique identification number.
- *Improvement area*: The process area addressed by the initiative.
- *Volume:* The # of weeks, # of man hours, and # of people participating in the initiative.
- *Structural Approach:* The number of (one or several) engineering projects that the initiative supported.
- *# of iterations:* The number of iterations executed within the initiative.
- *Result:* The result that was produced by the initiative (only in Appendix).
- *Implementation Success:* The degree to which the process was adopted as part of software engineering practices (further described in Appendix).

#	Improvement Area	Volume	Structural Approach	# of Iterations	Implementation Success
1	Configuration Management	10 weeks 300 man hours 4 participants	Generic	1.0	Low.
2	Design Information	21 weeks 400 man hours 6 participants	Generic	1.3	Medium/Low.

3	Estimation and Planning	14 weeks 600 man hours 11 participants	Generic	1.3	Low.
4	Historical Data	16 weeks 200 man hours 4 participants	Generic	1.0	Low.
5	Introductory Training	14 weeks 620 man hours 11 participants	Generic	1.0	Medium.
6	Module Test	12 weeks 400 man hours 10 participants	Generic	1.3	Medium/Low.
7	Project Tracking	9 weeks 300 man hours 7 participants	Generic	1.3	Medium/Low.
8	Resource Handling	4 weeks 250 man hours 8 participants	Generic	1.0	Medium.
9	Requirements Management	10 weeks 200 man hours 5 participants	Generic	1.0	Low.
10	Requirements Management Implementation	12 weeks 330 man hours 7 participants	Dedicated	1.3	Medium.
11	Subcontract Management	18 weeks 650 man hours 9 participants	Generic	1.0	Low.
12	Requirements Management	30 weeks 1200 man hours 3 participants	Dedicated	4.1	High.
13	Analysis & Design	30 weeks 1000 man hours 4 participants	Dedicated	5.3	Medium/High.
14	Implementation	30 weeks 2000 man hours 4 participants	Dedicated	4.0	High.
15	Test	30 weeks 1300 man hours 2 participants	Dedicated	4.0	High.
16	Configuration Management	30 weeks 1650 man hours 6 participants	Dedicated	4.1	High.
17	Project Management	10 weeks 150 man hours 2 participants	Dedicated	3.0	High.
18	Process Development Map	30 weeks 200 man hours 2 participants	Dedicated	4.0	High.

Table 1 – Extracted data from the 18 SPI initiatives

The data can be categorized in two groups, a) the ones that only concluded one full iteration (#1-11) and b) the ones that concluded several full iterations (#12-18). All SPI initiatives that reached level "High" in SPI implementation success executed at least three full iterations and all SPI initiative that executed at most one full iteration never achieved high SPI implementation success. The data also shows that there is a close correlation between structural approaches and number of executed iterations. Almost all SPI initiatives (#10 an exception) that were run according to a generic approach conducted less than two full iterations, while all the dedicated SPI initiatives conducted several full iterations.

4.2 The Detailed Study

To learn and understand what happens within each iteration, each SPI initiative was explored in detail. As an example, the requirements management initiative (# 12) executed 4.1 iterations. The first full iteration had the goal to define a Use Case template and define the strategy for the tool Requisite Pro. Two engineers (one process engineer and one consultant expert) were assigned to drive the work to define (a first version of) a new solution. The result was reviewed by a number of requirements engineers (the ones who were to use the new way of working). A few corrections were made and a first baseline version was ready. The baseline version was then presented to most of the requirements engineers and some more feedback (for instance, add a chapter to the Use Case template or define how to get passwords in the tool mentor) was given and a few more updates were made.

The second iteration had the goal to identify how to convert the current requirements artifacts into Use Cases and store the artifacts in the Requisite Pro tool. The drivers of the SPI work initiated a workshop to start merging the current Requirements picture into a Use Case overview model. The workshop ended up in total chaos. One requirements engineer left the room in fury. The result of the workshop ended up in a requirement to improve the new process*.* The process had according to the engineers substantial weaknesses when it came to merging existing requirements into new Use Cases.

The third iteration had the goal to correct the results of previous iteration as the practitioners had learned and understood more about the new way of working. The result was a new and improved version of the Use Case template and the Requisite Pro tool mentor. A new workshop was held where the practitioners started to convert the current requirements into a Use Case overview model. The result of the workshop was a rough overview model that needed further updates to include all necessary information.

The fourth iteration had the goal to update the way of working with the Use Case model to assure that all necessary information was included. A group of requirements engineers worked together with the process engineer and the consultant expert to address the lessons learned and problems from the last workshop. The result was a process that described the identified needs of the Use Case Model. The fifth iteration that halted during the diagnosing phase was stopped due to lack of resources. The goals for further improvement were not set and the initiative was ended.

The data from the detailed study of initiative #12 (requirements management) suggests that the engineers did not understand from the beginning how the initiative would impact their daily work. This understanding became clear during the second iteration when the result was exposed to them in a way where they should perform actual work to start converting the requirements picture. The data also shows that SPI implementation was a success only after the third iteration.

Data from all initiatives have been analyzed in a similar fashion in order to relate the iterations to the Satir Change Model (Weinberg, 1997). The result of this analysis is contained in Figure 5 showing in greater detail the correlation between iterations and phases in the different SPI initiatives (the circles denote the number and distribution of iterations over phases). These findings are further elaborated in chapter 5 – Analysis and Discussion.

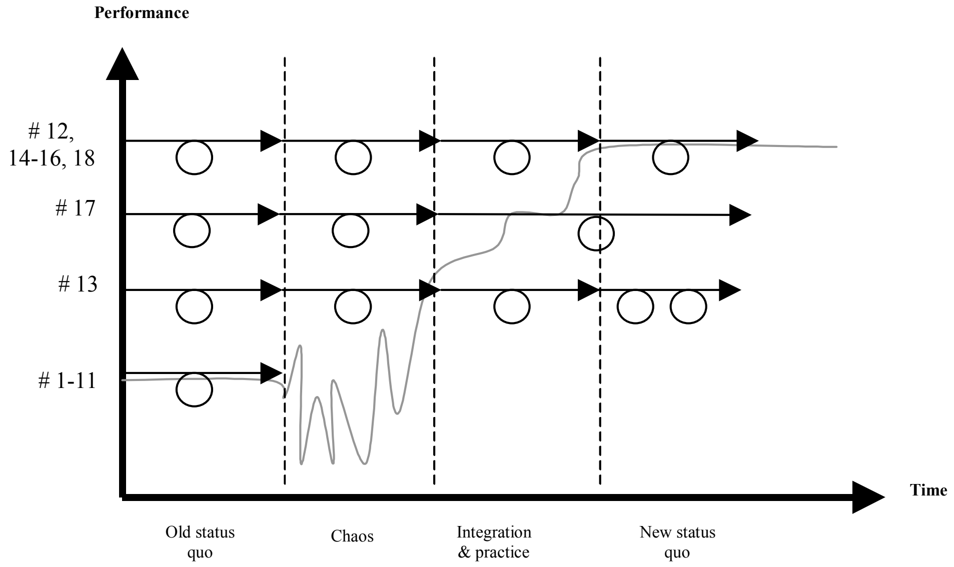


Figure 5. # of iterations executed within each phase of the Satir Change Model

5 Analysis and Discussion

This chapter discusses and analyzes the data from Ericsson and relates them to relevant literature about SPI and change management. Chapter 5.1 analyzes the data and identifies key findings. Chapter 5.2 discusses the implications for practice and research. Chapter 5.3 discusses possible limitations.

5.1 Interpreting the data

The relation between SPI implementation success and number of iterations is shown in Figure 6. The figure indicates that the number of iterations executed within the SPI initiative have a significant impact on SPI implementation success. The figure indicates a close correlation between low/medium SPI implementation success and less than two iterations executed within the SPI initiative. The figure also indicates a close correlation between high SPI implementation success and several iterations executed.

SPI initiatives #5 and 8 deviate somewhat from the “iteration curve”. One explanation is that others factors, apart from the number of executed iterations, affect SPI implementation success. Börjesson and Matthiassen (2003b) argue that process push (the competence, commitment and active participation of the SPI team to create and implement a process) and practice pull (the competence, commitment, and active participation of the software engineers to create and implement a process) affect the SPI implementation success. Other possible explanations can be the complexity of the improvement area and competence of participants within the improvement area.

Also, SPI initiatives #13 and #17 deviate somewhat from the “iteration curve”. SPI initiative #13, Analysis and Design, executed two iterations in the last phase, but still did not reach the score “high” in implementation success, while SPI initiative #17, Project Management, only executed three iterations in total and still achieved high implementation success (see Table 1 and Figure 5). The detailed study shows that initiative #13 was the far most complex and difficult initiative and was more time and expertise consuming than all the other initiatives. Initiative #17 had a well-known project manager who deploy the material. A possible explanation based on these data is that the complexity and scale of the initiative and the choice of champion in the SPI initiative also affect the outcome of the SPI implementation success.

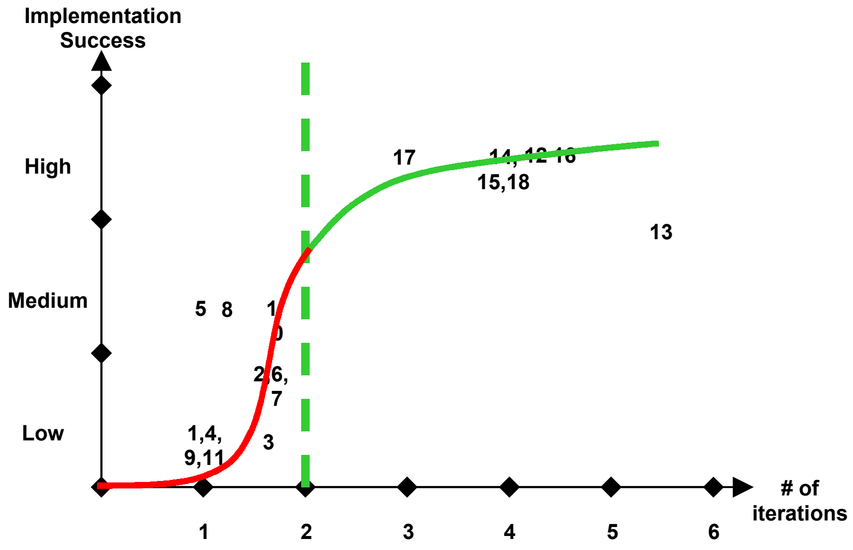


Figure 6. The correlation between SPI implementation success and number of iterations

SPI initiatives #1-9 and 11 all had a generic structural approach, i.e. the SPI initiative supported more than one software engineering project, while SPI initiative #10 and 12-18 all had a dedicated structural approach, i.e. the SPI initiative supported one software engineering project. Börjesson and Mathiassen (2003a) state that a dedicated structural approach facilitates SPI implementation success. The correlation between generic structural approaches, less than two iterations, and low implementation success can indicate that it is difficult to execute several iterations while supporting several software engineering projects. There is also a close correlation between dedicated structural approaches, several iterations, and high implementation success. That indicates that it is possible to run several iterations when supporting only one software engineering project. SPI initiative #10 had a dedicated structural approach, but still only executed one full iteration. This implies that iterations don't happen naturally just because the approach is dedicated.

This finding is supported by the widely accepted view that commitment to SPI is a necessity to assure SPI implementation success (Humphrey, 1989; McFeeley, 1996; Grady, 1997). To assure commitment we need involvement and change agents that stay current during an SPI initiative to assure that practitioners overcome resistance to change (Humphrey, 1989). When supporting many software engineering projects from one SPI initiative, it requires a lot of change agents to be able to stay current during several iterations in all the supported software engineering projects. There are most likely too few resources and too little time to manage it all. A possible explanation based on the data from this research is therefore that generic initiatives do not have the time and resources to run several iterations and therefore do not manage to implement the new or updated processes effectively enough.

From the detailed study we can see two main patterns for executing iterations within SPI initiatives. The first pattern shows initiatives only executing one full iteration and never entering the chaos phase. Most practitioners were never exposed to the change and did not feel threatened by the new or modified way of working. They were never exposed to the change in a way where they understood how the new process would affect their daily work. The second pattern shows initiatives executing several iterations. In the first iteration, most practitioners were not even aware of the ongoing initiative and as the new process was unknown there was no change. This phase is recognized as "Old Status Quo". During the second iteration the practitioners were exposed to the change and possibly hostile reactions to change arose (McFeeley, 1996). This phase is recognized as "Chaos". During the third iteration the practitioners started to work to achieve a useful solution. This phase is recognized as "Integration and Practice". The fourth and last full iteration made further improvements to the solution.

This phase is recognized as "New Status Quo".

Weinberg (1997) shows in the Satir Change Model that the way to reach higher performance is to pass through a number of phases (see Figure 7). According to data from the detailed study it took one full iteration to pass one phase in the Satir Change Model. A possible explanation to why the number of iterations executed within the SPI initiative has a significant impact on the SPI implementation success is therefore that an SPI initiative has to pass the chaos phase and enter the integration and practice phase before the new or modified processes will start to be implemented and used by the practitioners.

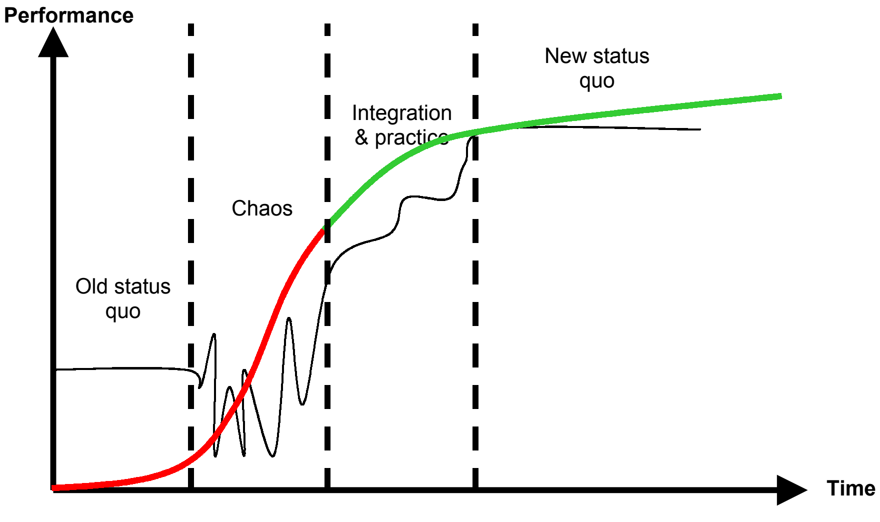


Figure 7. The Satir Change Model and "the Iteration Curve"

From Weinberg's Change Model (see Figure 2) we can learn that there are several risks to return to old status quo when performing an SPI initiative. This research supports this argument. In addition, the findings suggest that it can be a major risk that an SPI initiative never enters the phases where practitioners are exposed to the new or modified processes. In such cases there is a risk that the practitioners never will understand how the new or modified processes will affect their daily work.

The iteration curve in Figure 6 can be related to Connor and Patterson's Development of Commitment to Change model (see Figure 3). Over time, the commitment to the SPI initiative changes and these changes are facilitated by the number of iterations executed. The first phase in that model aim to create awareness, the second to achieve understanding, the third to get a positive perception, and the fourth to make it happen (divided into the phases installation, adaptation, institutionalization and internalization). If an SPI initiative only executes one iteration the initiative will never enter the phase where understanding for the new way of working is achieved. The third and the fourth phase where the positive perception among practitioners is reached and where the new way of working will happen are more or less out of reach.

Mathiassen et. al (2002) argue that participation makes improvement happen and there is a need to involve the people whose behavior is to change. This research supports this view and also states that it requires some time, i.e. several full iterations, before people start to change their behavior. An SPI initiative does not only have to plan for involving SPI practitioners. It also has to plan for how to make the practitioners stay current to overcome resistance to change. Aaen (2002) argues for the need of involving practitioners. Aaen's theory regarding End-user SPI, i.e. help the users to help themselves, requires that the users come to a point where they are willing and capable of giving input to the SPI initiative. In many cases they have to pass the hard chaos phase with most of their effort directed towards overcoming confusion and resistance to change. Only then will they reach a point where they feel integrated in and committed to the change initiative and they can start giving quality input and feedback to the design of the new process.

5.2 Implications for practice and research

As commitment to SPI is hard to establish and maintain, managers and SPI practitioners are advised to stay current during an SPI initiative to assure that SPI implementation will happen. To stay current with an SPI initiative means dedicating resources and time to assure SPI implementation and use. Moreover, quality feedback is necessary to arrive at a useful process and it often takes time (several iterations) before the engineers start to give relevant feedback.

SPI drivers need to be trained in the change management area to recognize chaos and resistance to change as an indication that the SPI initiative is actually going in the right direction towards higher performance. Far too often, the resistance to change by practitioners is misinterpreted to indicate that the new or updated processes are of poor quality. This might very well be the reason and the very idea with an iterative approach is to test and improve the solution. Nevertheless, resistance to change is natural because of the disturbance new processes cause and it must be expected to emerge independent of whether the solution is good enough or not.

This research does not reject generic structural approaches, but it indicates that we need to learn more about the relation between different structural approaches, the number and nature of iterations executed and practical ways to achieve successful SPI implementation. Weinberg’s (1997) and Connor and Pattersson’s (1982) models help us understand the impact change management has on a change initiative and how easy it is to return to the previous phase or never enter a new phase, but we still need to learn more about how iterative approaches relate to other barriers or enablers to successful SPI.

One particular factor that needs to be further explored and understood is the duration of an SPI initiative. The data in Table 1 show that the duration of the 18 SPI initiatives differ and one observation is that the initiatives with longer time periods seem to be more successful in implementing new practices. More research in this area is required to learn and understand how duration affects SPI implementation success.

5.3 Limitations

In this case study there are a number of limitations that affect the outcome of the research. At the outset it has to be stressed that SPI implementation success is not equal to SPI success. SPI implementation success is, however, a first and necessary step to achieve SPI success and is therefore considered an important factor. There is a possibility that SPI Implementation success could be a step towards SPI failure and the findings are therefore presented as indicators, not definite answers.

The finding regarding structural approach and SPI implementations success can also depend on the amount of resources and time spent in the SPI initiative. It is possible that the outcome would have differed if an SPI initiative with a generic approach had spent more time and resources. It is also possible that the outcome would have differed if the time spent within each iteration had been longer and more preparations to handle potential reactions to change had been made. It is also possible that factors like competence, culture, maturity in driving SPI initiatives, and previous knowledge in the area will affect the result in a way that this research does not consider. It is therefore important to further stress that the results from this research are indications rather than firm conclusions.

Abrahamsson (2000) argues that commitment to SPI is difficult to control. The reactions to change vary across initiatives, they are impossible to predict and difficult to manage. Arent (2000) also argues that there is a limit to what can be achieved by commitment to SPI. As the phases can be incontrollable, it is possible that other scenarios than the ones described in the Satir Change Model (Weinberg, 1997) and Development of Commitment to Change (Connor and Pattersson, 1982) can affect SPI initiatives, including the ones presented here. This paper builds on the belief that the change process is to a large extent manageable and the interpretations made are affected by this perspective.

6 Conclusions

This study indicates that the number of iterations executed within an SPI initiative have a significant impact on SPI implementation success. The data show a correlation between low/medium SPI implementation success and less than two iterations executed within the SPI initiative and between high SPI implementation success and several iterations executed. The research also supports the finding from Börjesson and Mathiassen (2003a), where a dedicated approach provides stronger support for SPI implementation success than a generic approach.

The findings are discussed in relation to the Satir Change Model (Weinberg, 1997) and the Development of Commitment to Change model (Connor and Pattersson, 1982). Two patterns arise from the data that show how iterations are executed per phase in the Satir Change Model. One pattern shows SPI initiatives only executing one iteration and never entering the phase where practitioners are exposed to new or modified processes in a way where they understand how the new process will affect their daily work. These SPI Initiatives seldom manage to implement in practice the new way of working. The other pattern shows SPI initiatives executing several iterations, where it takes one iteration to pass each phase. The number of iterations help the SPI initiatives through the different phases to reach higher performance and this is particularly important when passing the chaos phase, where practitioners are exposed to the change in a way that makes them understand how the new process will impact their daily work. These SPI initiatives typically manage to implement in practice the new way of working.

Appendix

#	Improvement Area	Volume	Structural Approach	# of Iterations	Result	Implementation Success
1	Configuration Management	10 weeks 300 man hours 4 participants	Generic	1.0 – the result was presented in an email where the updated processes could be found. Feedback was given that was considered, but no further improvements were started.	• CM policy for projects • CM role description • CM template for projects • Updated internal CM education	Low. The result was considered hard to use. Part of the result was used indirectly, when SPI participants applied the knowledge gained in other projects.
2	Design Information	21 weeks 400 man hours 6 participants	Generic	1.3 - the result was presented in an email where the updated processes could be found. One resource from the SPI initiative handled some of the feedback, but only a few updates were made and deployed.	• Template product information • Unified view on product structuring • A design model	Medium/Low. The intention was to provide a framework for design. The results were mainly implemented in one project that one process engineer worked in.
3	Estimation and Planning	14 weeks 600 man hours 11 participants	Generic	1.3 - the result was presented in an email where the updated processes could be found. One resource from the SPI initiative handled some of the feedback, but no major updates were made.	• List of tasks to be planned in a project • Guidelines for time estimation • Guidelines for re-planning	Low. The results were tried out in one project, but it ran into difficulties. Support was weak, and no one helped the project; resources from the SPI initiative were no longer available to make the necessary changes.
4	Historical Data	16 weeks 200 man hours 4 participants	Generic	1.0 – the result was presented to the steering group. Very little feedback was given and no new updates were made.	• Specification of historic project data • Diagrams for historic project data • Updated time reporting rules	Low. The purpose was to build a database of old data and take action from there, but no interesting data were found and thus no changes were made.

5	Introductory Training	14 weeks 620 man hours 11 participants	Generic	1.0 – the result was presented at a management meeting. Some feedback was given, but no resources was set free to make new updates.	• Activity flow for new employees • List of needed competences and courses • Manager's checklist for handling new employees • Checklist for tutoring • Template for planning introductory training • Guidelines for using checklists and templates	Medium. An estimated 50% of managers used the process. Some didn't know about it and weren't given the opportunity to learn about it. Managers were given supporting guidelines. More assistance was sometimes needed.
6	Module Test	12 weeks 400 man hours 10 participants	Generic	1.3 - the result was presented in an email where the updated processes could be found. One resource from the SPI initiative handled some of the feedback. A few updates were made and deployed.	• Policy for module testing • Generic test specification • Information about tools for module testing • Changed internal course	Medium/Low. The process was only used when process engineers were members of a project or when section managers strongly believed in systematic module tests.
7	Project Tracking	9 weeks 300 man hours 7 participants	Generic	1.3 - the ready result was presented at a project meeting. A few updates were made in the receiving project.	• Standardized project follow-up • Rules to ensure its use	Medium/Low. The process was only used in one project supported by the driver of the SPI initiative. That project was content with the outcome.
8	Resource Handling	4 weeks 250 man hours 8 participants	Generic	1.0 – the result was presented at a management meeting. There was an ambition to further improve the new solution, but no time and resources were prioritised.	• Process description • Process communicated to all line managers • The results were based on the use of a resource allocation tool	Medium. An estimated 75% of managers used the new process. Those not using the results either didn't get the help they needed or didn't believe in the approach.
9	Requirements Management	10 weeks 200 man hours 5 participants	Generic	1.0 – the result was presented in an email where the updated processes could be found. Feedback was given, but no further improvements were started.	• Policy for RM • Template for Requirements Specification • Change request procedure • Change request template	Low. The results were hard to use. The results were mainly used as a framework by the members of the SPI initiative.
10	Requirements Management Implementation	12 weeks 330 man hours 7 participants	Dedicated	1.3 – the result was presented at a project meeting. One resource stayed current to handle feedback. A few further improvements were made and deployed.	• Policy for connection between RM and project management • Checklist for milestones • Description of baselines • Updated change request template	Medium. The initiative was started because of the low impact of initiative #9. Here, the focus was on one project, and all results were designed to suit its needs.
11	Subcontract Management	18 weeks 650 man hours 9 participants	Generic	1.0 – the result was presented in an email where the updated processes could be found. A lot of feedback was given, but no resources were prioritised to stay current and work with further improvements.	• Updated purchase process • Checklist for agreement with subcontractor • Guidelines for tracking and communicating with subcontractors • Role descriptions	Low. The results needed further adaptation to be useful for different projects, but no effort was made to tailor them. Some of the results were indirectly used by process engineers in other projects.
12	Requirements Management	30 weeks 1200 man hours 3 participants	Dedicated	4.1 – the result was presented to an engineering project. SPI resources stayed current to handle feedback. New solutions were presented and new feedback was given in 4 iterations. The SPI work was ended when resources had to be prioritised to other work.	• RM workflow • Guidelines and templates • RM course • Tool course • Courses held • Mentoring performed	High. The process was adapted to a specific project, but needed further adaptation to actually be useful. Process engineers and practitioners solved these problems jointly and made the change happen.
13	Analysis & Design	30 weeks 1000 man hours 4 partici	Dedicated	5.3 – the result was presented to an engineering project. SPI resources stayed current to address	• A&D workflow • Guidelines and templates for A&D • A&D courses	Medium/High. This is a complex area, and required several iterations of experimenting with processes

		pants		feedback. New solutions were presented and new feedback was given in 5 iterations. The SPI work was ended during the deployment in the 5:th iteration.	• Tool course • Courses held • Mentoring performed	before the result was satisfactory.
14	Implementation	30 weeks 2000 man hours 4 participants	Dedicated	4.0 – the result was presented to an engineering project. SPI resources stayed current to address feedback. New solutions were presented and new feedback was given in 4 iterations before the steering group decided that we should not address more problems for now.	• Implementation workflow • Guidelines and templates for Implementation • Implementation course • Courses held • Mentoring performed	High. Two slightly different adaptations were made to fit the needs of different products developed on different sites. Collaboration between process engineers and software practitioners made the change happen.
15	Test	30 weeks 1300 man hours 2 participants	Dedicated	4.0 – the result was presented to an engineering project. SPI resources stayed current to address feedback. New solutions were presented and new feedback was given in 4 iterations before the work ended.	• Test workflow • Guidelines and templates for Test • Test course • Tool course • Courses held • Mentoring performed	High. The result was adapted to the specific needs of a specific project. Process engineers and software practitioners solved difficulties together.
16	Configuration Management	30 weeks 1650 man hours 6 participants	Dedicated	4.1 – the result was presented to an engineering project. SPI resources stayed current to address feedback. New solutions were presented and new feedback was given in 4 iterations. The work was ended during diagnosing in the 5:th iteration due to prioritisation difficulties.	• CM workflow • Guidelines, checklists and templates for CM • CM course • Tool course • Courses held • Mentoring performed	High. There are hundreds of possible solutions for each specific situation. It required extra attention to choose one and focus on making that happen. The dedication of the software practitioners played a key role.
17	Project Management	10 weeks 150 man hours 2 participants	Dedicated	3.0 – the result was presented to an engineering project. SPI resources stayed current to address feedback. New solutions were presented and new feedback was given in 3 iterations before the steering group decided that they were now satisfied with the solution	• PM workflow • Guidelines and templates for PM • PM course • Courses held • Mentoring performed	High. The results were actually used and the project managers' dedication to SPI work within project management continued.
18	Process Development Map	30 weeks 200 man hours 2 participants	Dedicated	4.0 – the result was presented to an engineering project. SPI resources stayed current to address feedback. New solutions were presented and new feedback was given in 4 iterations before a good enough result was achieved.	• Adaptation of Rational Unified Process • Rules for new updates to new releases • Guidelines for pictures, layout, weblinks and content • Deployment on the web	High. The use of the development process map is measured in both "hits" and subjective opinions of the need. All measurements are very positive.

Literature

Aaen I. (2002) Challenging Software Process Improvement by Design. Proceedings of ECIS 2002 The European Conference on Information Systems, Gdansk, Poland, June 6-8.

Aaen, I., Arent, J., Mathiassen, L. and Ngwenyama, O. (2001) A Conceptual MAP of Software process Improvement. Scandinavian Journal of Information Systems, Vol. 13, 123-146.

Abrahamsson, P. (2000) Is Management Commitment a Necessity After All in III: Software Process Improvement? Euromicro '00, Maastricht, The Netherlands, IEEE Computer Society, 246-253.

Abrahamsson, P. (2001) Rethinking the Concept of Commitment in Software Process Improvement. Scandinavian Journal of Information Systems 13:69-98.

Arent, J. (2000) Normative Software Process Improvement. Ph. D. thesis Aalborg Univerisity publication no. 11, ISSN nr:1399-8145

Bach, J. (1995) Enough About Process: What We Need are Heroes. IEEE Software, 12, 2, 96-98.

Bollinger, T.B and McGowan, C. (1991) A Critical Look at Software Capability Evaluations. IEEE Software, Vol. 8, No. 4, 25-41

Börjesson, A. and Mathiassen, L. (2003a) Making SPI Happen: The IDEAL Distribution of Effort. Hawaiian International Conference on System Science, 2003.

Börjesson, A. and Mathiassen, L. (2003b) Making SPI Happen: The Road to Process Implementation. Proceedings of the Twenty-sixth Information Systems Research Seminar in Scandinavia, 2003.

Connor and Pattersson (1982) Building Commitment to Organisational Change. Training and Development Journal (April): 18-30

Diaz, M. and Sligo, J. (1997) How Software Process Improvement Helped Motorola. IEEE Software, Vol. 14, No. 5, 75-81.

Fayad, M. E. and Laitinen, M. (1997) Process Assessment Considered Wasteful. Communications of the ACM, Vol. 40, No. 11, 125-128.

Galliers, R. D. (1991) Choosing Appropriate Information Systems Research Approaches: A Revised Taxonomy. In: Nissen, H.-E., H. K. Klein and R. A. Hirschheim (Eds.): Information Systems Research: Contemporary Approaches & Emergent Traditions, Elsevier.

Grady, R. B. (1997): Successful Software Process Improvement, Upper Saddle River, New Jersey: Prentice Hall.

Haley, T. J. (1996) Software Process Improvement at Raytheon. IEEE Software, Vol. 13, No. 6, 33-41.

Humphrey, W. S. (1989) Managing the Software Process. Reading, Massachusetts: Addison Wesley.

Humphrey, W. S., Snyder, T. R. and Willis, R. R. (1991) Software Process Improvement at Hughes Aircraft. IEEE Software, Vol. 8, No. 4, 11-23.

Humphrey, W. S. and Curtis, B. (1991) Comments on "A Critical Look at Software Capability Evaluations". IEEE Software, Vol. 8, No. 4, 42-46.

Mathiassen, L. (2002) Collaborative Practice Research. In: Baskerville, R., J. Stage and J. I. DeGross: Organizational and Social Perspectives on Information Technology, Kluwer Academic Publishers.

Mathiassen, L., Nielsen, P. A. and Pries-Heje, J. (2001) Learning SPI in Practice. In: Mathiassen et al. (Eds.) Improving Software organisations. From principles to Practice. Upper Saddle River, N. J.: Addison-Wesley.

McFeeley, B. (1996) IDEAL: A User's Guide for Software Process Improvement. Pittsburgh: SEI. Handbook, CMU/SEI-96-HB-001.

Weinberg, Gerald M. (1997) Quality Software Management volume IV – Anticipating change. Dorset House Publishing.

Yin, R. (1994) Case Study Research. Newburry Park, California: Ssage Publications.

7 Author CVs

Anna Börjesson
Gjutegården 60
436 45 Askim
Sweden

Mail: anna.borjesson@ericsson.com
Tel./Mob: +46 747 6148
Date of birth: 1970-05-07

Education

Industrial Doctoral Candidate IT University, Gothenburg	200201-
Master of Science in Informatics, 4 year (160 p), Gothenburg	199108-199506
Technical gymnasium 4 year, Gothenburg	198608-199006

Relevant working experience

Manger at Ericsson AB (EAB)	200201-
Manger at Ericsson Mobile Data Design AB (ERV)	199801-200112
Consultant at WM-data	199409-199801
Substitute teacher, for students 7-15 year, all areas	199008-199106

Domains

Mobile Packet Data	ERV/EAB
Intranet design	WM-data
Paper mill, administrative system	WM-data

Knowledge areas

Software Process Improvement Research	IT University
SW Quality Assurance	Ericsson
Configuration Management	Ericsson
Software Process Improvement work	Ericsson
RUP adaptation work and implementation	Ericsson
Process development and implementation (mostly SW)	Ericsson
Organisation and infrastructure for improvement and process work	Ericsson
Inspections, reviews, lesson learned, quality assurance, etc	Ericsson
Management/leadership	Ericsson
Project Management, teacher	WM-data
Intranet design	WM-data
HTML-programming	WM-data
System design and test, administrative system	WM-data
Administrative programming, MPL	WM-data

Tools

Windows Office packet
Frontpage
Image composer
Req Pro (from a process view)
Rose (from a process view)
Clear Case (from a process view)
Clear DDTS (from a process view)
Clear Quest (from a process view)

Training/courses

Leader ship reflections, 4 days	Jan-May-01
My leadership, 3 days	Oct-00
RUP internal courses, 4 days	Jan-Dec-00
Requirements management, 1day	June-00
Interview techniques, 1day	April-00
GPRS System overview, 2 days	Feb-00
Communication in the leadership, 3 days	Feb-00
RUP overview, 2 days	Oct-99
Business management step 2, 3 days	Oct-99
Business management step 1, 3 days	Sept-99
OOAD, UML & Rose, 4 days	Aug-99
GSM survey course, 4 days	Mars-99
Req Pro, overview, ½ day	Feb-99
Project management, 4 days	Oct-98
Ericsson management training, step 1-5, 15 days	–98/99

Conferences/Seminaries

IRIS 26, paper presentations, 3 days	Aug-03
ECIS Doctoral Consortium, 3 days	June-03
HICSS 36, paper presentation, 4 days	Jan-03
IRIS 25, paper presentation, 3 days	Aug-02
E/// Rational User conference - speaker, 1 day	Oct-01
Methods - do we have time for that, 1 day	Oct-00
E/// Rational User conference - speaker, 2 days	Sept-00
Change in Telecom, 2 days	March-00
Killer applications for the mobile society, 1 day	Oct-99

ECSE, system architectures, 3 days	June-99

Publications

2002-1) Making SPI Happen: The IDEAL Distribution of Effort. *Proceedings of the Twenty-fifth Information Systems Research Seminar in Scandinavia, 2002.*

2003-1) Making SPI Happen: The IDEAL Distribution of Effort. *Hawaiian International Conference on Systems Science, 2003. (Modified and revised version of 2002-1)*

2003-2) Proposal paper - Making SPI Happen. *ECIS Doctoral Consortium in Naples, June 2003*

2003-3) Making SPI Happen: The Roads to Process Implementation. *Accepted at Proceedings of the Twenty-sixth Information Systems Research Seminar in Scandinavia, 2003.*

2003-4) Making SPI Happen: The Impact of Iteration on Implementation Success. *Accepted at Proceedings of the Twenty-sixth Information Systems Research Seminar in Scandinavia, 2003.*

2003-5) Making SPI Happen: Iterate Towards Implementation Success Success. *Accepted at EuroSPI 2003. (Modified and revised version of 2003-4)*

Language

Swedish	Native
English	Good
German	Basi

Static Code Analyzers – Making a Wise Choice

M.R. Narasimhamurthy, T.S. Raghavan

Abstract

Static code analysis tools (Static Code Analyzers) help to cut down the large amounts of time required for manual code reviews and additionally support many aspects of software quality assurance for the project. Static Code Analyzers are capable of giving very useful code metrics from which an analysis database for the project can be created. In addition, visualization of code structure, and detailed reports on key aspects of the source code can be easily generated. This paper presents the experience of the team, SCCT (Static Code Checking Technology, a member of the Software Technology Appreciation Group in the organization, PSC (Philips Software Centre (P) Ltd., Bangalore), in making an in-depth study of the existing usage of the static code analysis tools in the organization, evaluating their functional scope with respect to the standard requirements of a static code analysis tool, suggesting thereby “appropriate” tools and provides tips regarding how to make a good choice of the tool.

Keywords

Static code analysis, Static code analysis tools, Standard requirements of a static code analyzer tool, Weighting factors

1 Introduction

Software systems, today, have become larger and more complex. In today's consumer electronic products, the high integrity software needs to be checked as thoroughly as possible for its correctness, before its fusion in the product. Static code analysis is a method of determining properties of programs by inspecting their code, without executing them. These properties can be used to optimize the programs [1]. Using the dynamic testing to verify the behavior of large programs such as embedded critical software does not provide enough information on the complexity and quality of the code. The failure rate of software can not be predicted, as software does not conform to any known statistical model. Static code analysis helps to reason about the structure of the code easily and thereby helps users to understand the code. In addition, visualization of code structure and detailed reports on key aspects of the source code can be easily generated. Of late, software development projects are defining static code analysis as an indis-pensable part of the set of procedures governing the software quality assurance for the project.

Static code analysis enables one to detect a significant number of defects in the code, at a much early stage in the software development cycle, even before the code is compiled. Performing code reviews manually to catch such static defects as coding standard deviations, language misuse, potential loopholes in the code etc. exacts vast amounts of time. Static code analysis tools (Static Code Analyzers) help to cut down the large amounts of time required for manual code reviews and additionally support many aspects of software quality assurance for the project. Static Code Analyzers are capable of giving very useful code metrics from which an analysis database for the project can be created.

This paper presents the experience of the team, SCCT (Static Code Checking Technology, a member of the Software Technology Appreciation Group in the organization, PSC (Philips Software Center (P) Ltd., Bangalore), in making an in-depth study of the existing usage of the static code analysis tools in the organization, evaluating their functional scope with respect to the standard requirements of a static code analysis tool, suggesting "most appropriate" tools and providing tips regarding how to make a good choice of the tool. This paper thereby intends to suggest how to make a good choice of the tool.

2 Significance

Software system often needs to be modified to meet new requirements or to add new features. Making changes to code which is spread across multiple text-files may induce inconsistencies to other modules of the code [2]. In such scenarios, code inspection will require vast amounts of time and becomes very difficult to perform since the code is not easily traceable. With a poor documentation of the source code it adds more woes to the code inspections since it becomes cumbersome to understand the code.

Collecting code metrics in such large software systems, when done manually, consumes lot of time. It is again time consuming to remeasure the quality of an entire system after one or more source files are modified [2].

Therefore, static code analysis tools are significant from the following two perspective:

- First, to reduce code review or inspection time and thereby speed up the review process
- Second, to gather useful code metrics to measure code quality and for supporting quality assurance actions

With the help of good static analysis tools, the user can gain a quick insight into the problematic areas of code, areas that need much focus for review and/or rewrite [3]. Static code analysis tools can check and identify syntax, unreachable code, infinite loops, initialization errors, code complexity, uncalled functions, possible array bound errors and much more. These tools can also operate quite fast, operating on about 100,000 lines of input in under one second. They also generate a variety of useful code metrics like for e.g., McCabe's cyclomatic complexity metric and simplify the code documentation by providing easy to read charts and reports in system/function levels.

3 Overview of our work

The Static Code Checking Technology (SCCT)-team started the work with the following main objectives:

- To evaluate the usage of the existing tools in the organization
- To evaluate the functional scope of these tools with respect to the requirements of a standard static code analyzer
- To recommend appropriate tools and guidelines for making a wise choice of a static code analysis tool
- To bring in the changes needed in the related processes and technologies in the organization to improve the effectiveness of static code checking

An extensive survey was conducted across the organization, to assess the usage trend of the static code analysis tools. Then, a detailed gap analysis of the functional scope of the existing tools with respect to the standard requirements of a static code analysis tool, was made. Based on the outcome, the “most appropriate” tools are suggested. Recommendations are made for a wise choice of the tools.

4 Towards making a wise choice

In this section, firstly is presented the approach followed by the SCCT-team to select the “most appropriate” tools (section 4.1) and secondly, factors and issues influencing a wise choice of the tool are addressed (section 4.2).

4.1 Our approach

As the first step, with the intent of gathering a clear outlook on the existing tool usage profile, an extensive survey was conducted throughout the organization. This survey was done based on a questionnaire that exacted critical details such as the name of the tool, purpose for which it is being used, language and platform on which the tool can work, ease of use of the tool and any problems being faced while tuning the tool to the needs of a project. Some additional general information such as the name of the project for which the tool is used, names of the user and department, which owns the project etc., were also gathered. It was found that broadly, three categories of projects existed in which static code analysis tools are being used. These categories are respectively, projects based on C -programming language, projects based on C++ -programming language and projects based on JAVA -programming language. The usage profile of the tools is presented in Appendix 1.

As the second step, standard requirements of a static code analyzer tool, in general, are formulated. These are presented in Appendix 2 and serve as ready reference.

As the third step, currently used tools: QA C (version 4.2.1), QA C++ (version 1.2), LCLint (version 2.5), CodeCheck (version 2.20), QStudio JAVA (version 1.7) and a proprietary tool (version 02 – an in-house built tool for the MIMIT project group of Philips Medical Systems product division) developed for the sole purpose of enforcing project specific JAVA coding standards, are evaluated with respect to these standard requirements. Statistics of this gap analysis report are presented in Appendix 3. Based on the gap analysis, “best” candidates for each of the three project category are selected and suggested.

As can be seen from the usage profile of the tools in Appendix 1, QA C [4] has the majority usage in the category of C-programming language based projects. LCLint [5] is specifically used for projects in one of the product divisions in the organization. QA C++ [6] and CodeCheck [7] share equal percentage of use in the category of C++-programming language based projects and QStudio JAVA [8] and the proprietary tool [9] share equal usage percentage in the category of JAVA-programming language based projects.

4.2 Weighting factors

Selection of an appropriate static code analyzer depends on several factors out of which some are governed by the software project constraints. The selection depends primarily on the coding language used in the project. Other factors like cost, ease-of-use etc. also play their role influencing the final selection. The following are the important factors that can be assigned a weight on an appropriate scale (say, a scale of 1-10) that can be fixed based on a consensus of the user community and the project management team, in the project:

4.2.1 Ease of use

Denotes the user-friendliness of the tool for the purpose of usage.

4.2.2 Ease of customization

Quantifies the effort required for configuring the tool for enforcement of project specific coding standards and desired code metrics. It should be simple and should take less effort to customize the tool to enforce a project specific coding standards/guidelines and to produce the required set of code metrics.

4.2.3 Ability to generate reports and documentation

Quantifies the ability to generate a wide variety of graphi-cal reports and diagrams on the structure of the analyzed code, easy-to-refer documentation on the analyzed source files, graphical reports on code metrics measured and to produce a database containing the analysis data for the whole software project.

4.2.4 Cost

The cost of the tool should be worked out so that it can be weighed against the total project cost and the available budget to see if feasible to be purchased. Any additional cost due to the tool support and maintenance should also be worked out for feasibility of purchase.

4.2.5 Tool support

Any maintenance support for the tool purchased is also important to be considered when offered by the tool vendor.

4.2.6 Tool functional scope

The functional scope of the tool can be evaluated against the standard requirements presented in Appendix 2. Then the criteria for selecting the “appropriate” tool can be determined based on the number of vital requirements met by the tool.

All of the above listed factors have to be considered and wisely weighed out before making a final choice.

5 Conclusions

As the importance of static code analysis tools are being realized and appreciated, gradually their use is being mandated for code quality assurance purposes in many software development projects, today. But choosing an "appropriate" candidate to fit well the scope of a software development project is not straight-forward. An in-depth insight is required before finalizing a selection.

A standard set of requirements for a static code analyzer tool has been formulated and provided in Table 1 in Appendix 2. This set of requirements were used to evaluate the functional scope of the tools QA C, QA C++, CodeCheck, LCLint, QStudio JAVA and the proprietary tool, used in various projects in the organization. This same set of requirements can be used to evaluate other static code analysis tools as well. As can be noticed from the gap analysis statistics with respect to the tool functional scope in Appendix 3, QA C, QA C++ and QStudio JAVA have emerged as the "most appropriate" candidates for the given projects in the organization.

6 Acknowledgements

We wish to sincerely thank and would like to express our appreciation for the following for all their support and assistance offered during the course of this work:

- Mr. K.V. Ramachandran, Head of the Technology Change Management Committee, Philips Software Centre (P) Limited, Bangalore
- Mrs. Veena Y., Technology Manager of the Technology Change Management Committee, Philips Software Centre (P) Limited, Bangalore
- All those project leaders, software developers who extended their co operation for the evaluation of the tools and answering the tool-survey questionnaire

We also would like to sincerely thank Ostfold Software (P) Limited, India, for granting us the evaluation/demo license for the QA C tool.

7 Literature

[1] Dr. Bruno Blanchet, "Static Analysis – Summary", Max-Planck-Institut für Informatik (January 2002), available at http://www.mpi-sb.mpg.de/units/nwg1/summary.html

[2] "Automating Source Code Static Analysis -- Techniques and Tools", International Software Automation, Inc., available at http://www.softwareautomation.com/analysis/www/problem.htm

[3] Jose Nazario, "Source Code Scanners for Better Code", Miscellaneous category, Web Articles Index, Linux Journal, Specialized Systems Consultants, Inc. (SSC) (January 26, 2002), available at http://www.linuxjournal.com/article.php?sid=5673

[4] "User's Guide, QA C Deep Flow Static Analyzer", Programming Research Ltd., Glenbrook House, 1-11 Molesey road, Hersham, Surrey, U.K., available at http://www.prqa.co.uk

[5] David Evans, "LCLint User's Guide", University of Virginia, Department of Computer Science, available at http://www.splint.org/guide/guide.pdf

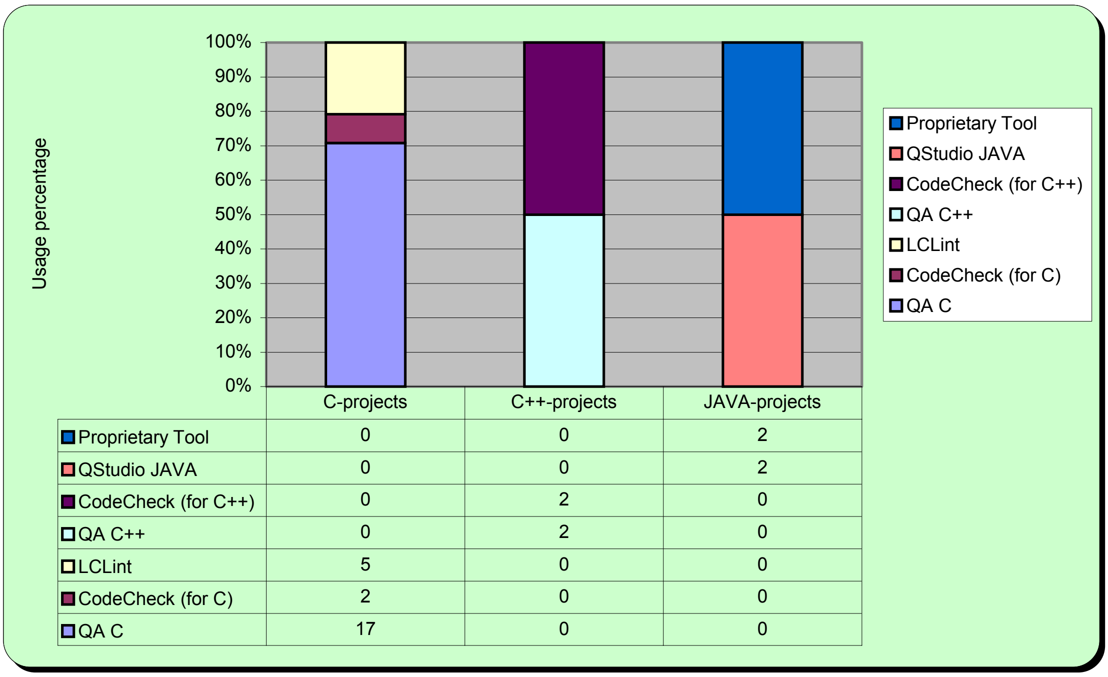
[6] "Introduction to QA C++ Generation 2000", Help-file: QACPP.HLP, Programming Research Ltd., Glenbrook House, 1-11 Molesey road, Hersham, Surrey, U.K., available at http://www.programmingresearch.com/

[7] "C and C++ Source Code Analysis using CodeCheck", Abraxas Software, Inc., Portland, USA, available at http://www.abxsoft.com/codchk_user.html

[8] Erik Reek, "QStudio JAVA Concepts", White Paper, QA Systems Inc., available at http://www.itpapers.com/cgi/PSummaryIT.pl?paperid=20903&scid=234

[9] "Joscar - An in-house built static code analysis tool for JAVA programming language", MIMIT-Group, Philips Medical Systems, Philips Software Center (P) Ltd., Murphy Road, Ulsoor, Bangalore-8, India

8 Appendix 1



	C-projects	C++-projects	JAVA-projects
Proprietary Tool	0	0	2
QStudio JAVA	0	0	2
CodeCheck (for C++)	0	2	0
QA C++	0	2	0
LCLint	5	0	0
CodeCheck (for C)	2	0	0
QA C	17	0	0

Figure 1: Usage profile of the evaluated static analysis tools in C, C++ and JAVA – projects

9 Appendix 2

Requirements
File Specific • Capable of checking for the presence of File-header and Function-header (comment blocks)[1] • Program/file/function compactness (active lines, comment lines, blank lines and the percentage, partial comment lines and the percentage, and total lines)[1] • Program/class/function complexity[1] • Complexities involved in Program logic, Control flows, Logic path, Class, Base class and Derived class, Data member and Method / Function member, Constructor and Destructor, Friend Function/Class, Class nesting, Template[1] • Identifying the classes affected when a class / function / data element is changed[1] • Template for control structures supported by the language (layout and code structure generated automatically when coding)[1]
Function Specific • Virtual functions[1]

[1] With a High priority preference

- Overloaded functions[1]
- Static functions[1]
- Unused header files, defines, externals and functions[1]
- Cross reference[1]
- Global/ static variable usage[1]
- Listing of all Functions/Data element that differ by one character (Close name analysis)[1]
- Report of Included files and preprocessor symbols[1]
- Global variable definition, usage and modifications[1]
- Static variable definition, usage and modifications[1]

Error checking/warning

- Syntax error[1]
- Semantic error[1]
- Confusing operator precedence in mathematical and logical expressions[1]
- Unused code (functions)[1]
- Unused global variables[1]
- Unused static variables[1]
- Unused labels[1]
- Uninitialized variables[1]
- ISO/ANSI standards violation[1]
- Portability Issues[1]
- Complexities[1]
- Use of unspecified and undefined language features[1]
- Maintainability issues (missing “break” in switch-case constructs)[1]

Graphic representation of the analysis results

- Capable to graphically display function call relationship, class inheritance, class-object coupling, program logic, program control flow, logic path, program tree, class nesting and quality measurement results[1]
- Command line Interface should be available[1]
- Highlightable (can highlight a logic path with the path execution condition extracted on a control flow diagram, highlight a function with the related functions calling it, called by it, or both on a call graph, highlight the functions related to a global variable or static variable on the generated function call graph, etc.)[1]
- Searchable with search engines automatically built-in with the reporter or graphical viewer[1]
- Traceable / linkable (linking the function call graph to the logic diagram or control flow diagram from each function box by a mouse click, linking a calling statement to the called function, a *goto* statement to the corresponding label, an object definition to the class definition / constructor, from a node of class /function from the program tree to the body of the class/function on a generated logic diagram of an entire program, etc.)[1]
- Printable in ASCII format or RTF format or postscript format[1]

[1] With a High priority preference

- Sorting/indexing all analysis results/data[1]

Others

- Client-server application support in a NFS-network[2]
- Multiusers support through network license sharing and control[2]
- Allow the user to tune error/warnings for the need of the project[2]
- Graphical (Motif/Openlook/Windows) user interface[1]
- Adding a small search engine to the reporter and Graphical viewer[2]
- Allowing user to build his/her *bookmarks* to mark chosen items on the generated reports or diagrams of an entire program[2]
- Attaching a click-to-jump table of calling statement locations with a function definition on the linked diagrams for easily checking the types and the numbers of parameters at each calling statement[2]
- Allowing user to change the colors of the diagrams / charts[2]
- Allowing user to view any sub-set of a program for all reports[2]
- Allow to determine the metrics/standard violations for legacy and newly added code[2]

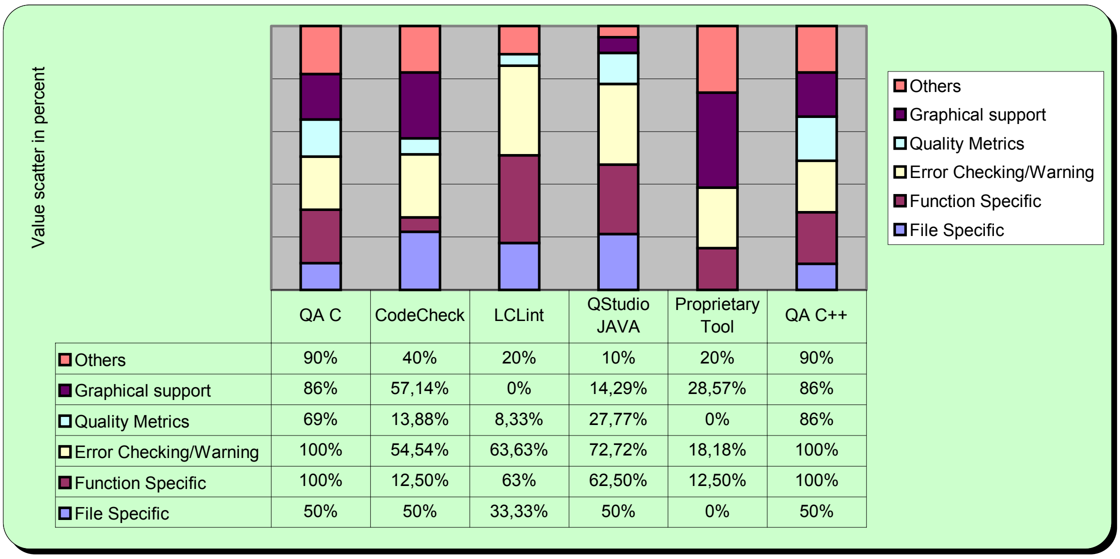
Quality measurement using Structure and Object-Oriented metrics

- Cyclomatic complexity[1]
- Akiyama's criterion[1]
- Estimated Static Program Path[1]
- Myer's interval[1]
- Deepest level of nesting[1]
- Number of dangling else if[2]
- Number of executable lines[1]
- Number of Goto's[2]
- Number of code lines[1]
- Unused Variables[2]
- Number of functional calls[2]
- Path density[1]
- Residual bugs[2]
- Code portability as percentage[1]
- Comment to Code Ratio[1]
- Number of External Variables declared[2]
- Number of functions in a file[1]
- Number of Static Variables declared[2]
- Total number of Variables[2]
- Total preprocessed code lines[2]
- Total unpreprocessed code lines[2]

[1] With a High priority preference
[2] With a Low priority preference

- Lines of Code / class[1]
- Number of Methods /class[1]
- Number of Method Users / class[2]
- Weighted Methods / Class in multiple complexity metrics[2]
- Depth of inheritance / class[1]
- Number of Children / class[1]
- Coupling between objects[1]
- Response for a class[1]
- Lines of Code Reused / class[1]
- Ratio of Code Reused / class[1]
- Metrics selection support[1]
- Standard value setting support[1]
- Quality data collection[2]
- Quality evaluation and graphic result display in 2D and 3D diagrams[1]
- Allowing user to select only one class or one function to view its quality measurement result[1]
- Allowing user to assign different weight to different metrics for evaluating the total quality of a program[1]

10 Appendix 3



	QA C	CodeCheck	LCLint	QStudio JAVA	Proprietary Tool	QA C++
■ Others	90%	40%	20%	10%	20%	90%
■ Graphical support	86%	57,14%	0%	14,29%	28,57%	86%
□ Quality Metrics	69%	13,88%	8,33%	27,77%	0%	86%
□ Error Checking/Warning	100%	54,54%	63,63%	72,72%	18,18%	100%
■ Function Specific	100%	12,50%	63%	62,50%	12,50%	100%
■ File Specific	50%	50%	33,33%	50%	0%	50%

Figure 2: Evaluation statistics for the tools: QA C, QA C++. LCLint, CodeCheck, QStudio JAVA and the Proprietary Tool

11 Author CVs

[1] With a High priority preference
[2] With a Low priority preference

M.R. Narasimhamurthy

M.R. Narasimhamurthy, presently, is working in the Philips Research division of Philips Software Center (P) Limited, Bangalore as Senior Research Specialist since January 01, 2002. Before, he has rendered services in the product divisions: Philips Semiconductors and Philips Consumer Electronics since he joined the organization in October 1996. He has contributed in various embedded software development projects such as Up-market TV, High-end VCR, TV-VCR combination, Hard Disk Recorder. He was also intensively involved in the organization wide software process improvement activities and has lead a team for formulating guidelines for static analysis enhancement. His interests include: Research in the software testing, software engineering and Embedded software design and development.

Contact information:
Address: Philips Research India – Bangalore, Philips Software Center (P) Limited, Philips Innovation Campus, #1, Murphy Road, Ulsoor, Bangalore – 560 008, Karnataka State, India
Telephone: +91-80-5579000
Fax: +91-080-556 0580
E-mail: murthy.m@philips.com

T.S. Raghavan

Raghavan.S.Tirumale, presently, is working in the Philips Consumer Electronics division of Philips Software Center (P) Limited, Bangalore as a Technical Leader since April 1999. He is Respected member in the competence center for Tools and Test Engineering Group, playing a part on deploying various Project life cycle related tools. Mainly concentrated on development of various tools related to Software Testing and Design in Philips consumer electronics worldwide, Philips Electronics.N.V. He was also intensively involved in the organization wide software process improvement activities His interests include: Software testing, software engineering and Embedded software design and development.

Contact information:
Address: Philips Consumer Electronics, Philips Software Center (P) Limited, Philips Innovation Campus, #1, Murphy Road, Ulsoor, Bangalore – 560 008, Karnataka State, India
Telephone: +91-80-5579000
Fax: +91-080-556 1241
E-mail: raghavan.tirumale@philips.com

A Case-based Assessment of Recommendations for Implementing Software Metrics

Helle Damborg Frederiksen

Abstract

Textbooks on software engineering agree that software metrics are useful in managing and improving software processes. Substantial amounts of recommendations have been given to practitioners on how to implement metrics programs. State-of-the-art recommendations were used in a Danish software organization implementing a metrics program for benchmarking and improvement purposes. After five years the organization's metrics program and practices were systematically analysed, and it turned out that the benefits from the metrics program were not as great as expected. Where did the recommendations – or the company – fail? The purpose of this paper is to assess the recommendations on implementing software metrics based on the case. The assessment is done in two steps: 1) An experience-based assessment, and 2) A theory-based assessment. The theory-based assessment draws on alternative perspectives on software metrics programs. From the assessment it is suggested how the recommendations for implementing software metrics could be improved.

Keywords

Software metrics, implementation, case study

1 Introduction

Management of software development processes is a complex task. Often projects are behind schedule and the resulting applications lack quality. To perform the task, managers depend on different types of evaluations of the software development process, e.g. written reports from project managers or software metrics like productivity, quality, adherence to schedule and budget. Several companies have implemented metrics programs to support managers in their decisions. However, the benefits from the implementation are not as great as expected. Nearly 80 % of software metrics programs fail within the first two years (Dekkers, 1999).

It is challenging and resource demanding to establish well functioning metrics programs. Various writers have expressed this, and some of them have captured key elements in successful metrics program implementation (Rifkin & Cox, 1991; Pfleeger, 1993; Dekkers, 1999; Iversen & Mathiassen, 2003). Hall & Fenton (1997) have summarized some of these insights in the following advice on metrics implementation: Incremental implementation, transparency, usefulness, developer participation, metrics integrity, feedback, automated data collection, practitioner training, gurus and champions, dedicated metrics team, and goal-oriented approach.

This kind of advice is helpful when implementing a metrics program in an organization. However, practitioners have to interpret and prioritize the advice in the setting of their own organization, e.g. how is the feedback going to be performed, and how should the metrics team be organized. In other words, the advice should be transformed into operational actions in the organization.

The Danish software company, SWH (the company is anonymous), has implemented an elaborate metrics program. This paper describes how the company used the recommendations to assist in their implementation of the metrics program. However, an analysis of the metrics practices revealed five years after the implementation that the benefits from the metrics program were not as great as expected. We ask: Where did the recommendations or the company fail?

We use the SWH-case to assess the recommendations. This is done in two steps as illustrated in Figure 1. The first step is an assessment based on the experience from SWH. Next we do a theory-based assessment in order to further understand the experiences from SWH and to get inspiration as to how to improve the recommendations. The improved recommendations will not necessarily guarantee success for practitioners that adopt them. However, the likelihood of success is expected to increase.

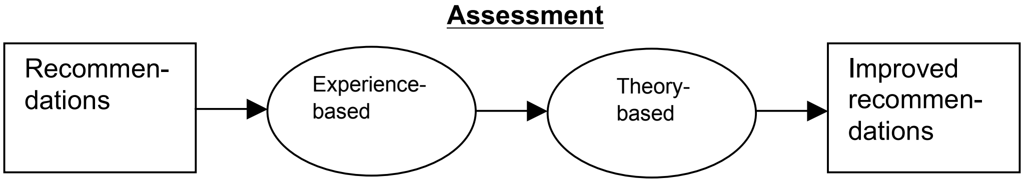


Figure 1: Structure of the assessment.

We conducted a case study (Yin, 1994) at SWH in order to do the experience-based assessment. To do the theory-based assessment we searched the relevant literature to find theories that could support and explain the findings from SWH. That led us to work by Niessink & Vliet (1999) and Kautz & Nielsen (2002). Niessink & Vliet introduce a model, which stresses that the implementation of a metrics program should always support the solving of a problem in the organizational context. Kautz & Nielsen emphasize issues related to organizational change and suggest a perspective that is suitable for implementing a metrics program.

Data for the experience-based assessment included the artifacts, the roles, the processes etc. of the metrics program at SWH. The collection and analysis of the data was one of the initial activities of a 3-year action research effort (Mathiassen, 2002) at SWH with the purpose of improving metrics practices. The author, who is an employee at SWH, performed the research and was originally a partici pant in the project to implement the metrics program. The investigation was performed in close collaboration with the practitioners that played an active role in the design and implementation of the metrics program at SWH. The primary source was the project manager of the implementation project, and the investigation included unlimited access to material from the implementation of the metrics program at SWH, e.g. numerous documents, minutes of meetings, e-mails, and a personal log kept by the author of this paper.

The structure of the paper is as follows. Section 2 provides an overview of the recommendations summarized by Hall & Fenton (1997). The company, SWH, and their metrics program are presented in section 3. The experience-based assessment is then presented in section 4, including SWH's interpretation of the recommendations and a summary of the conclusions of the analysis of the metrics practices at SWH. Section 5 provides a presentation of the theories used for and the result of the theory-based assessment. The resulting overall assessment of the recommendations given by Hall & Fenton is then discussed in section 6, and improvements to the recommendations are suggested. Lastly, section 7 concludes the paper and illustrates the implications for research and practice.

2 Recommendations on Implementing Metrics

Over the years a lot of advice on implementing metrics programs has been provided. Mostly, the recommendations in the literature are based on single program experiences. Hall & Fenton (1997) have studied available research, and identified a consensus on requirements for success of metric programs. Therefore it is of general interest to our field to assess and if possible improve the recommendations by Hall & Fenton. SWH used the recommendations to guide the implementation of their metrics program during 1996-97. We have recently conducted a critical analysis of the resulting metrics program (Frederiksen & Mathiassen, 2002) and this provides an interesting basis for assessment of the recommendations. Hall & Fenton summarize the advice as listed below:

- Incremental implementation: Implementing a metrics program over time holds significantly less risk than a "big bang" approach.
- Transparency: The metrics program must be obvious to practitioners. They must understand what data is being collected, why it is being collected, and how it is being used.
- Usefulness: The usefulness of metrics data should be obvious to all practitioners. Otherwise, they will collect data without enthusiasm and the data will probably lack validity.
- Developer participation: Developers should participate in designing the metrics program. This way, buy-in is more likely, as is the implementation of a more incisive metrics program.
- Metrics integrity: Practitioners should have confidence in the collected data. They should believe it is sensible to collect, accurately collected, and not being "fiddled".
- Feedback: When practitioners get feedback on the data they collect, it gives them a clear indication that the data is being used.
- Automated data collection: Data collection should be automated wherever possible. Minimizing extra work for developers also minimizes developer resistance to metrics. Furthermore, the data collected is more likely to be valid.
- Practitioner training: The metrics program should have a base of trained practitioners. Appropriate training must be targeted at all levels in a company and should range from awareness rising to training in statistical analysis techniques.
- Gurus and champions: Organizations can increase practitioners' initial enthusiasm by bringing in an external metrics guru. Organizations should also appoint internal metrics champions to help with the difficult and arduous task of sustaining a metrics program.
- Dedicated metrics team: Responsibility for the metrics program should be assigned to specific individuals.
- Goal-oriented approach: It is very important that companies collect data for a specific purpose. Usually the data will be for monitoring the attainment of an improvement goal.

The study of Hall & Fenton confirms that the success of a metrics program depends on a carefully planned implementation strategy. Furthermore, success seems linked to an organization's willingness to do background research on other metrics programs and use advice given in the published experiential reports. However, there is no advice on how the organization should use the recommendations.

3 The Metrics Program at SWH

The business of SWH is to develop, maintain and run IT-solutions for the local authorities in Denmark. The more than 2400 employees including almost 700 software developers make SWH one of the largest software organizations in Denmark, and it is operating at different sites located across Denmark. The software development is largely organized in projects and teams according to applications. A unit supporting the software organization with methods, tools and techniques is responsible for software process improvement initiatives and for the metrics program. SWH has a longstanding tradition of collecting simple measures, e.g. the number of faults in an application. In 1996 the CEO initiated a project to implement an elaborate metrics program with the intention of increasing productivity and quality in the software development process at SWH. In order to identify the weaknesses of the software development process, SWH should be benchmarked against other software development companies. Hence, an external contractor, Compass, supplied the metrics program called Development Compass.

The basic elements of the model in Development Compass are projects and applications. Furthermore, data on general overhead activities, personnel, development software, hardware, and systems software is collected. The model includes about 500 data points, a third of which are compulsory.

The supply of data (measures) to the metrics program is illustrated in Figure 2. The project and team managers supply measures on their own projects and applications to the metrics program every three months. The measures include time spent on various activities (study, analysis, project management etc.) and measures on errors reported on and corrected in applications in the measurement period. The measures are supplemented with characteristics of the project or application, e.g. business area, technology and size measured in function points. On demand from the project or application manager a function point analyst in the unit counts the number of function points.

Units external to the software unit supply measures on personnel, and expenses for consultants and software etc. A controller in the software unit supplies the data on time spent on activities not related to projects and applications, e.g. management and administration.

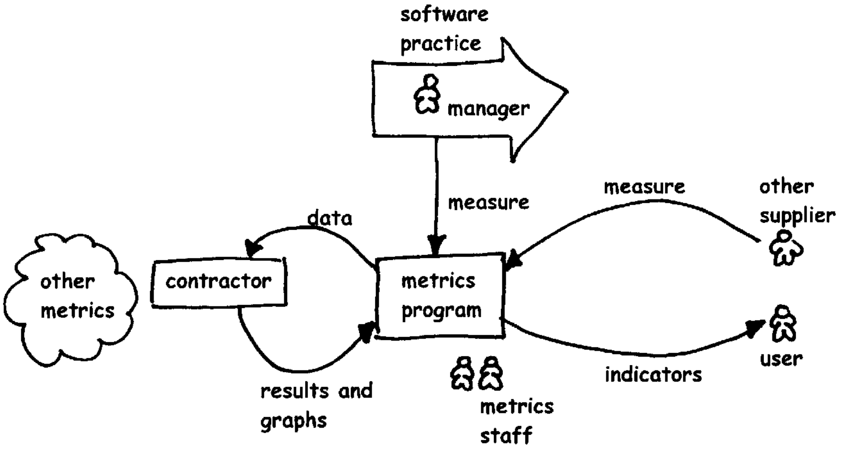


Figure 2: The metrics program at SWH.

The data is processed, validated and packaged at SWH before they are sent to the contractor, Compass. Having processed the data Compass sends a set of results and graphs to SWH. The results include indicators at a general level and at a project and application level, e.g. number of faults in ap plication per function point, number of function points delivered per day in projects etc. The indicators are used for benchmarking SWH against a reference group at Compass. Furthermore, the results are used for internal benchmarking among the software units. The purpose of the benchmarking is to identify possible areas for improvement. A selection of the results is fed back to the organization.

Once a year, Compass delivers a written report. The report includes an analysis of identified strengths and weaknesses at SWH, and a set of recommendations for further actions. Compass and the software process improvement organization present the report to top management. After top management has discussed the report the results are available to the project and team managers.

The metrics staff at SWH consists of a specialist and a group of controllers. The specialist is responsible for the metrics program. The responsibility includes specifying the strategy for the measurements and mapping the model from Compass to the software practice at SWH. The controllers are responsible for controlling the data collection in the unit, supporting the project and application managers in supplying data, and presenting results in the unit. The specialist and the controllers are organized in a group that meets on a regular basis. Generally, the members of the metrics staff spend 25 % of their time on the measurements.

4 Experience-based Assessment

The experience-based assessment has two inputs: 1) SWH's interpretation of the recommendations (presented in section 4.1 to 4.4), and 2) The conclusions from our recent analysis of the resulting metrics practices at SWH (presented in section 4.5).

In order to benefit from recommendations like those suggested by Hall & Fenton in section 2 you have to interpret them in the setting of your own organization. Interpretation in an organization consists of prioritizing the recommendations and transforming them into actual activities. The priorities of the recommendations specified by SWH are listed in Table 1. For each recommendation a priority of *high*, *medium* or *low* is indicated.

Recommendation	**Priority**
Incremental implementation	High
Transparency	Medium
Usefulness	High
Developer participation	Low
Metrics integrity	High
Feedback	Medium
Automated data collection	Low
Practitioner training	High
Gurus and champions	Low
Dedicated metrics team	High
Goal-oriented approach	Low

Table 1: SWH's priorities of recommendations.

In this paper we will look at four key recommendations. Similar arguments apply to the remaining recommendations. For a detailed description of SWH's interpretation of the total set of recommendations see (Bøttcher & Frederiksen, 2001).

4.1 Incremental Implementation

Hall & Fenton: *Implementing a metrics program over time holds significantly less risk than a "big bang" approach.*

An incremental implementation makes it possible to learn from experience before involving the entire organization. However, it is not trivial to decide on which parameters should be incremental. Should it be the number of measures, the number of projects or the number of sites?

In SWH Compass was implemented in three phases. Compass provided the definition of the metrics to be used. Before implementing the metrics program there was a need to translate and transform the metrics into the terminology and processes of SWH. To enable this transformation a rough translation of the Compass model was prepared.

When initializing the measurement, a pilot project was established to adapt the metrics program to SWH, to establish tools for data collection, and to get an idea of the types and usefulness of the results. In this pilot all the compulsory data points in the Compass metric were implemented. This left out a large part of the metrics program, but approximately 150 measures were still left for implementation. To limit the pilot further the first month of the pilot was limited to 3 projects and 2 applications. During the full length of the pilot the implementation was limited to a maximum of 8 projects (2 on each site) and 16 applications (4 on each site).

Piloting the metrics across all sites was considered important to allow for learning on the complexity in implementing the metric in different sites with potentially different processes, procedures and terminology. All of these differences had to be unified to the degree required by the metric.

A pilot based on approximately 150 data points led to some variation in the detail in which the data points were defined and tested. Another, less complex strategy would have been to limit the metric further by a sequential pilot implementation of the Compass main areas: projects, applications, and general IT-company metrics. However, the model requires that all of the compulsory data points are used.

Eighteen months after the pilot project started, an implementation project was established to roll out the metric across all projects and 80 % of the applications within a year and a half. In this phase, the limitation of only focusing on the compulsory measures in the Compass metric was maintained. A wider organizational implementation was chosen over a more complete use of the Compass metric. A wide implementation would provide an analysis that covered most of the company but in less detail.

4.2 Usefulness

Hall & Fenton: *The usefulness of metrics data should be obvious to all practitioners. Otherwise, they will collect data without enthusiasm and the data will probably lack validity.*

The collected data should be considered useful by the data suppliers in order to motivate them. Motivation is important to ensure quality in data. The complexity of adapting this recommendation depends on the metrics. If the metrics are useful in controlling the project, the project managers tend to be more motivated than if the metrics are used for long-term process improvement.

The implementation of the Compass metric was originally motivated by top management's wish to benchmark company performance and improve processes based on quantitative data. For instance the productivity of different tools should be compared. However, during and after the pilot and implementation projects we have aimed at making additional use of the data visible to all levels of the development organization. We have, for example, established an estimation model based on the metrics program. This model uses the historical information in the data to provide project estimates.

In the first two years of piloting and implementing Compass, it was hard to communicate the usefulness of the metrics data. The use of project data for estimating future projects was communicable to most parts of the organization. For project data this meant that data quality was high, whereas in the other parts of the metric, application areas and cost measures, data quality did not improve until actions were taken based on the results. These actions were process improvement initiatives, such as software configuration management, or setting goals for performance. This indicates that the use and usefulness of data were better understood at this point.

4.3 Dedicated Metrics Team

Hall & Fenton: *Responsibility for the metrics program should be assigned to specific individuals.*

A dedicated metrics team is needed to remain focused on the metrics program. More specifically, a project is needed to make the implementation happen. Otherwise, the implementation tends to be lengthy and usually unsuccessful. Adapting the recommendation is not very complex. However, retaining commitment and sponsorship is an ongoing problem.

To implement the metrics program a pilot and an implementation project was established. An important part of the implementation project was to design the organization to be responsible for the metrics program after concluding the project. The organization turned out to have close resemblance to the project. All the participants from the project became members of a metrics team called the Compass-group. The project manager became the chairman of the group and the involved tasks were similar. Hence, the transition from project to permanent organization went smoothly.

During the implementation the function point analysts were organized in a group. This group remained unchanged after concluding the project.

4.4 Goal-oriented Approach

Hall & Fenton: *It is very important that companies collect data for a specific purpose. Usually the data will be for monitoring the attainment of an improvement goal.*

According to Hall & Fenton a goal-oriented approach is not vital to the implementation of a metrics program. It is possible to have several goals, e.g. benchmarking and estimation. However, the usefulness of the metrics program will be clearer if there is a well-defined goal. Hall & Fenton mention that weak motivation is common, and that the relationship between weak motivation and failure of the metrics program is not clear. Adapting the recommendation on a goal-oriented approach is not simple. Defining the goal in detail is a complex task.

The goal of implementing the metrics program at SWH was to increase development productivity and application quality. The CEO formulated the goal at this very general level. The goal was not elaborated, and no criteria for fulfilling the goals were specified. To define the criteria a baseline of the productivity and the quality was needed. The intention was to set this baseline from the first reports from Compass. However, this never happened.

To achieve the goal of increasing productivity and quality the metrics program should be used for benchmarking SWH against other software companies. Benchmarking requires a common model for collecting data and for calculating indicators to be able to compare the results. The model and a set of data for benchmarking were supplied by Compass.

4.5 Analysis of Metrics Practices at SWH

Five years after the implementation of the metrics program at SWH we analyzed the resulting metrics program and practices. The analysis was conducted as a case study (Yin, 1994) viewing the metrics program as an information medium to support software management and improvement efforts. The analysis of the metrics practices of SWH is reported in (Frederiksen & Mathiassen, 2001). The conclusions from the analysis are in summary:

- The data suppliers do not understand the precise definitions of metrics and measures.
- The metrics program contains a lot of data that is potentially useful for several purposes, e.g. tracking the amount of resources used for correcting errors.
- The primary use of the metrics is by the software improvement agents for high-level software process improvement.
- Most software engineers and managers do not appreciate the actual use of the metrics program. Most managers believe they can run their business without the metrics program.
- A proper interpretation of data requires insight into the metrics program, the submitted measures, and the actual software practices. Few are capable of that, hence, results are often presented in quite general terms and the readers are left to interpret them on their own.
- There are few managerial responses based on information from the metrics program.

4.6 Summary of Experience-based Assessment

From section 4.1 to 4.4 it appears that it is a challenging task to interpret the recommendations in the context of an organization. For instance, the recommendation on incremental implementation could be operational in several ways depending on how you interpret the term incremental. Is it the number of measures, number of projects or sites that vary? In the recommendation on usefulness the aspect of time should be acknowledged. Measurement for software process improvement is useful in a long-term perspective. People need, however, measures to be useful in the short-term perspective as well. Furthermore, having a goal of software process improvement is too vague, it should be specified in more details.

The findings presented in section 4.5 strongly suggest that even though SHW used the recommendations, the metrics program is not an unconditional success. Major parts of the organization do not understand the rationale of the measurements and have difficulties collecting and interpreting data. Furthermore, the primary use at SWH is high-level software process improvement, i.e. the data is used for identifying potential areas of improvement. The general opinion at SWH is that there are no managerial responses based on the metrics program.

5 Theory-based Assessment

In order to further understand these findings from the experience-based assessment and to get inspiration on how to improve the recommendations we do a theory-based assessment. We broaden the perspective and draw on two complementary sources. The first, Niessink & Vliet (1999) focuses on how to improve the metrics by stressing the importance of the context of the metrics program. The second, Kautz & Nielsen (2002) contributes on how to facilitate the change that is required by an organization in order to implement a metrics program by pointing to a relevant perspective on organizational change. The two sources of the theory-based assessment are presented in section 5.1 and 5.2. We use these sources to explain the shortcomings of the recommendations or the way SWH used them.

5.1 Metrics in a Context

According to Niessink & Vliet (2001) there is a need to broaden the focus when implementing metrics programs, i.e. to look at the purpose and context for the measurements. In order to assess the suc cess of the metrics program the organizational context must be considered, i.e. the measurements should generate value, not just data. When introducing a metrics program, a set of success criteria that are explicit, attainable, and measurable should be specified.

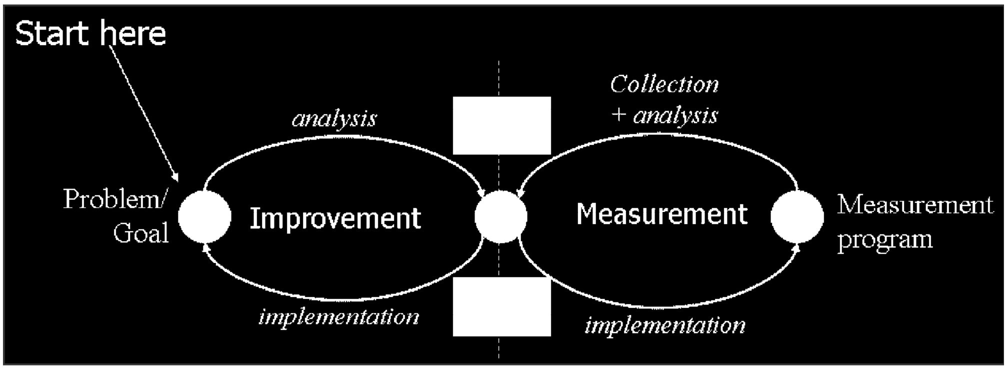


Figure 3: Generic process model.

To facilitate the implementation of a metrics program that supports solving a problem in the organizational context they introduce a generic process model with four steps, see Figure 3. The model describes an iterative, ongoing learning process that we here present one step at a time. The outset is an organizational problem, which is analyzed. This step is labeled improvement, analysis. Based on the analysis, the organization must decide on possible causes and possible solutions to the problem. To decide which of the possible causes is the real cause of the problem, or which of the possible solutions is the best solution the organization should collect information. The collection of information occasionally requires implementation of a metrics program. This step is labeled measurement, implementation. The data is gathered and analyzed in order to relate it to the proposed causes and solutions. This step is labeled measurement, analysis. Finally, the organization solves the problem by implementing the solutions found. This step is labeled improvement, implementation. Niessink & Vliet think of the steps labeled measurement as internal to the metrics program, and they argue that Hall & Fenton's recommendations are focused on this part. This is generally the case in this type of literature, which addresses practitioners (Frederiksen & Iversen, 2002).

To generate value rather than just data from implementing a metrics program attention should be paid to the proper mapping of organizational problems to the metrics program and from the results to identifiable actions. Niessink & Vliet point at additional factors 'external' to the metrics program:

- The assumptions underlying the metrics program should be made explicit and tested if necessary.
- The organization should consider all possible outcomes of the measurements and decide how to act on them.
- The organization should indeed act according to the outcomes.
- The organization should monitor the changes implemented based on the results from the measurement program to decide whether the changes really constitute improvements.

These additional factors are captured in the generic process model for measurement-based improvement suggested by Niessink & Vliet illustrated in Figure 3.

5.2 Facilitating Change

A successful implementation of a metrics program requires a considerable change to the organization:

"Implementing measurement often requires a significant cultural change within an organization. Just as with any other change, uncertainties associated with the introduction of measurement may cause anxiety… Measurement is most effective within a culture that encourages people to articulate problems and risks. This can be a major shift for many organizations" (McGarry, 2001).

Traditionally, the software industry has held a simplistic view of implementing change as diffusion of technology (Kautz & Nielsen, 2002). Kautz & Nielsen address this issue through three alternative perspectives on organizational innovation: the individualist perspective, the structuralist perspective, and the interactive process perspective. The individualist perspective focuses on characteristics of relevant individuals and core concepts are champions, leaders, entrepreneurs, and change agents. The structuralist perspective focuses on the organization’s structural element and core concepts are environment, size, complexity, normalization, centralization, and professionalism. A basic assumption of the individualist and the structuralist perspective is that change is a simple, linear process, and that focus is on the adoption stage. The interactive process perspective views innovation implementation as a complex interaction of individual and structuralist elements and core concepts are context, innovative capability, proliferation, and shocks. A basic assumption of this perspective is that change is a complex, social process.

Each of the recommendations from Hall & Fenton can be classified as having either the individualist or the structuralist perspective, e.g. *dedicated metrics team* has an individualist perspective, and *incremental implementation* has a structuralist perspective. However, Kautz & Nielsen argue that neither of these perspectives is sufficient in isolation. To fully understand the organizational change introduced by implementing a metrics program the complex interaction between individual action and structural influences must be considered. The interactive process perspective should hence be reflected in the supplementary recommendations on implementing metrics programs. In their paper Kautz & Nielsen illustrate how they gained a deeper understanding of actual change processes by using this framework.

5.3 Summary of Theory-based Assessment

From the experience-based assessment it appears that the recommendations suggested by Hall & Fenton can be improved to be of more value to a practitioner. These experiences are complemented well by the theoretical perspectives.

The experience-based assessment reveals that the recommendations fail to guide the practitioner in the implementation of the metrics program on the additional factors suggested by Niessink & Vliet. For instance it is not stressed that the organization should prepare for all possible outcomes of the measurements and decide on how to act on them. SWH hadn’t considered all possible outcomes, and the managerial responses were sparse. The few actions taken were very long-term and never monitored using data from the metrics program. There is a need for contemporary managerial responses. High-level software process improvement is a long-term goal and it isn’t a sufficiently specific goal to motivate the metrics program in the generic process model suggested by Niessink & Vliet.

In their presentation of the recommendations Hall & Fenton are mainly focused on implementing the metrics program as a technological innovation. The recommendations do not fully appreciate the complexity of the change process. Some of the recommendations could be interpreted in a manner that would take the interactive process perspective, e.g. the recommendation on developer participation. However, taking a simple view on the change process SWH interpreted this differently and chose to give the recommendation low priority. In general, planning the implementation of the metrics program SWH held the traditional view of implementing change as rather straightforward diffusion of technology.

In summary, the theory-based assessment supports the findings from the experience-based assessment. Furthermore, it gives inspiration on how to improve the recommendations.

6 Improved Recommendations

Building your implementation of a metrics program on a set of recommendations does not guarantee success. However, our assessment illustrates that the recommendations from Hall & Fenton can be improved to increase the likelihood of success. Some of the recommendations should be modified and additional recommendations should be added.

First, it is important to broaden the focus when implementing metrics programs, i.e. to look at the purpose and context of the measurements. The role of the measurements in the broader context of improvement should be clarified in the organization. This is stressed by Niessink & Vliet (2001), and several of the problems SWH experiences with the metrics program are covered by this recommendation. For instance SWH experiences that the primary users are software improvement agents, and that other people do not appreciate the use of the metrics. This call for a modification of the recommendation on Goal-oriented approach: *It is very important that companies collect data for a specific purpose, which is closely linked to the context of the metrics program. Make sure that the purpose is specifically related to short-term needs and not merely to long-term process improvement.* Having a more specific purpose it would have been easier for the management of SWH to act upon the information from the metrics program. They didn't know what they were expected to do. Furthermore, they would have benefited from a recommendation that you should prepare for all types of outputs: *All possible outcomes of the measurements should be considered and the management should decide how to act on each of them.* In this way the organization will prepare for and commit to responses to measurements which points to areas for improvement. Otherwise, you could easily spend time on questioning the measurements and deciding on how to respond.

Second, the metrics program should generate value. Specify a set of success criteria for the metrics program that are explicit, attainable, and measurable. As pointed out by Niessink & Vliet (2001) the metrics program will not be considered successful unless it leads to successful changes. It is important to demonstrate that actual decisions and changes can be traced to the information delivered by the metrics program. At SWH there are few managerial responses based on information from the metrics program. Furthermore, the actions are not perceived as results of measuring the software process. The recommendation would be: *Specify a set of success criteria for the metrics program, which are explicit, attainable, and measurable. The criteria should make sense in the context of the measurement program.* Having this kind of success criteria the expected benefits of the measurements at SWH would have been more realistic.

Third, it is important to be aware of the complex aspects of organizational change. Introducing a metrics program constitutes a considerable change for an organization. The software industry has traditionally seen change as simplistic diffusion of technology. Kautz & Nielsen (2002) argue that the interactive process perspective is vital in order to fully understand the change processes. This perspective can explain events that cannot be foreseen or explained by the individualist or structural perspectives. More insight into the change process will help you plan the change and help you navigate through the complex situations involved in a change process. The recommendation would be: *Handle the implementation as a complex and challenging change to the organization. Avoid the pitfall of viewing the change as simple diffusion of technology.* SWH would have gained from recommendations that would broaden the perspective on the change process.

Finally, interpreting the recommendations is an important task that needs to be explicitly addressed. To guide the practitioner in the interpretation details should be added to some of the recommendations. Giving examples of the interpretation of the recommendation on incremental implementation would inspire the practitioner to see the complexity of the recommendation. Another phrasing of the Incremental recommendation would be: *Implementing a metrics program over time holds significantly less risk than a "big bang" approach. Be aware that there is numerous ways to plan the incremental implementation.* Likewise, the recommendation on Usefulness needs modification in order to point out that there is a short-term and a long-term view on usefulness. The recommendation would then be: *The usefulness of metrics data should be obvious to all practitioners. Otherwise, they will collect data without enthusiasm and the data will probably lack validity. Consider usefulness from a short-term as well as a long-term perspective.*

To interpret the recommendations a practitioner should consider the priority of the recommendations and the complexity of adapting the recommendation to the organization. This will provide valuable input to create and explicate a specific strategy for implementing a metrics program. We hence propose to group the recommendations into four categories to help the practitioner address this issue and make priorities:

- Creating approach: Recommendations on how to interpret and prioritize the remaining recommendations
- Designing measures: Recommendations on how to design the specific measures

- Implementing program: Recommendations on how to design and implement the processes
- Managing change: Recommendations on how to manage the change

As a practitioner you should consider all the recommendations. However, working your way through your own adapted strategy your focus should shift from designing measures to implementing program and finally managing change. This grouping is used for summarizing the improved recommendations in Table 2 in the Conclusion.

7 Conclusion

We have presented the recommendations given by Hall & Fenton (1997) on metrics implementation. When a practitioner has to implement a metrics program, the recommendations based on prior experience and research are helpful. However, recommendations always require interpretations. As a practitioner you have to prioritize the recommendations in your arriving at your own strategy and translate the recommendations into concrete activities in your own organization. Consequently, different organizations will come up with different interpretations of the recommendations and the formulation of the recommendations should facilitate this process.

SWH used the recommendations in the implementation of their metrics program. However, five years later an analysis of the metrics practices revealed that the benefits from the metrics program were not as great as expected. That is, the recommendations do not guarantee a successful implementation. We used the case at SWH to assess the recommendations. This was done in a two-step assessment – an experience-based and a theory-based assessment. The theory-based assessment gave theoretical support for the findings from the experience-based assessment and it also gave further inspiration on how to improve the recommendations for practitioners on implementing metrics programs.

The improved recommendations are summarized in Table 2. New recommendations and modifications of existing recommendations are emphasized for easy comparison with Hall & Fenton’s original recommendations.

Group	Recommen-dation	Description
Creating approach	*Specific strategy*	*A specific strategy should be build from the recommendations below. Adapt and prioritise the recommendations in a manner, which fits the organization. Focus should shift from designing measures to implementing program and finally to managing change.*
Designing measures	Goal-oriented approach	*It is very important that companies collect data for a specific purpose, which is closely linked to the context of the metrics program. Make sure that the purpose is specifically related to short-term needs and not merely to long-term process improvement.*
	Usefulness	*The usefulness of metrics data should be obvious to all practitioners. Otherwise, they will collect data without enthusiasm and the data will probably lack validity. Consider usefulness from a short-term as well as a long-term perspective.*
	Transparency	The metrics program must be obvious to practitioners. They must understand what data is being collected, why it is being collected, and how it is being used.
	Metrics integrity	Practitioners should have confidence in the collected data. They should believe it is sensible to collect, accurately collected, and not being “fiddled”.

Group	**Recommen-da-tion**	**Description**
Implementing program	Feedback	When practitioners get feedback on the data they collect, it gives them a clear indication that the data is being used.
	Automated data collection	Data collection should be automated wherever possible. Minimizing extra work for developers also minimizes developer resistance to metrics. Furthermore, the data collected is more likely to be valid.
	Practitioner training	The metrics program should have a base of trained practitioners. Appropriate training must be targeted at all levels in a company and should range from awareness rising to training in statistical analysis techniques.
	Specific success criteria	*Specify a set of success criteria for the metrics program, which are explicit, attainable, and measurable. The criteria should make sense in the context of the measurement program.*
Managing change	Incremental implementation	*.Implementing a metrics program over time holds significantly less risk than a "big bang" approach. Be aware that there is numerous ways to plan the incremental implementation.*
	Developer participation	Developers should participate in designing the metrics program. This way, buy-in is more likely, as is the implementation of a more incisive metrics program.
	Gurus and champions	Organizations can increase practitioners' initial enthusiasm by bringing in an external metrics guru. Organizations should also appoint internal metrics champions to help with the difficult and arduous task of sustaining a metrics program.
	Dedicated metrics team	Responsibility for the metrics program should be assigned to specific individuals.
	Change process	*Handle the implementation as a complex and challenging change to the organization. Avoid the pitfall of viewing the change as simple diffusion of technology.*
	Possible outcomes	*All possible outcomes of the measurements should be considered and the management should decide how to act on each of them.*

Table 2: Improved recommendations.

These improved recommendations will not guarantee success. However, the likelihood of success is expected to increase. SWH could have interpreted the original recommendations along the line expressed by the insight captured in the improved recommendations in Table 2. However, the original recommendations did not challenge SWH in their understanding of the change process, and they focused on the 'internals' of the metrics program. The set of improved recommendations would have helped SWH in their implementation of the metrics program in the sense that the priorities of the recommendations would have been clearer and there would have been more recommendations focusing their attention towards the 'externals' of the metrics program.

This research has implications for practitioners and researchers. Practitioners should be careful in choosing recommendations from the literature. The recommendations are build on assumptions, and therefore they will be biased. This was also the case for the recommendations given by Hall & Fenton, which were biased towards the 'internal' of the metrics program. Researchers should elaborate on the improved set of recommendations suggested in this paper. More rigorous and empirical research into implementation of metrics programs should also be conducted. There is a need for more theory in studying metrics programs in order to broaden the understanding of the metrics programs and their successful implementation into software practices.

8 Literature

Bøttcher, P. and H.D. Frederiksen (2001). Implementing Software Metrics for Benchmarking. In Proceedings of 11ICSQ, Pittsburgh, USA.

Dekkers, C.A. (1999). The Secrets of Highly Successful Measurement Programs. Cutter IT Journal, 12(4), 29-35.

Frederiksen, H.D. and J. Iversen (2002). Implementing Software Metrics Programs: A Survey of Lessons and Approaches, IRMA2003, Pennsylvania, USA.

Frederiksen, H.D and L. Mathiassen (2002). Diagnosing Metrics Practices in a Software Organization. In New Perspectives on Information Systems Development: Theory, Methods and Practice. Kluwer Academic, New York, USA.

Hall, T. and N. Fenton (1997). Implementing Effective Software Metrics Programs. IEEE Software, Vol. March/April 1997, 55-64.

Iversen, J. and L. Mathiassen (2003). Cultivation and Engineering of a Software Metrics Program. Information Systems Journal, Vol. 13, No. 1 (3-21).

Kautz, K. and P.A. Nielsen (2002). Knowing and Implementing SPI. In Improving Software Organizations: From Principles to Practice. Addison-Wesley.

McGarry, J. et al (2001). Practical Software Measurement. Objective Information for Decision Makers. Addison-Wesley.

Mathiassen, L. (2002). Collaborative Practice Research. Information, Technology & People, Vol. 15, No. 4. Also published in: Scandinavian Journal of Information Systems, Vol.14.

Niessink, F and H. van Vliet (1999). Measurements Should Generate Value, Rather Than Data. In Proceedings of the Sixth International Software Metrics Symposium, Boca Raton, Florida, USA, November 4-6, 1999, pp. 31-38.

Pfleeger S. L. (1993). Lessons Learned in Building a Corporate Metrics Program. IEEE Software, 10(3).

Rifkin, S. and C. Cox (1991). Measurement in Practice. Pittsburgh: Software Engineering Institute.

Yin, R. K (1994). Case Study Research Design and Methods, Second Edition, Applied Social Research Methods Series, Volume 5, SAGE Publications.

9 Author CVs

Helle Damborg Frederiksen

Graduated with a master's degree in computer science from the Department of Computer Science at Aalborg University, Denmark in 1992. Teaching undergraduates in several aspects of computer science in the period 1992 to 1995.

Employed at the method's department at SWH since 1995. Involved in implementing the metrics program since 1997, and now responsible for a team of people working with the collection and use of metrics. Since fall 2000 doing a PhD-study funded by SWH and the Department of Computer Science at Aalborg University. The study is an action research effort aimed at both the scientific and practicing communities, giving two purposes. (1) Further the scientific understanding of how data on software development are used and can be used in software management and improvement. (2) Provide useful guidelines on how to design metrics programs that become useful for managing software organizations.

A software metric research to evaluate the dimension of standard systems' customization

Margherita Martellucci, Anna Cavallo, Francesco Maria Stilo, Domenico Natale, Nicoletta Lucchetti

Abstract

The issue concerning the evaluation of the customizing dimension was only recently born with the progressive increase of the ERP systems' customizing costs. After having chosen the ERP system, many enterprises found it rather difficult to conform to the standard procedures during the implementing phase; as a consequence, there was the request of new and ad-hoc procedures to be added to the standard ones supplied by the ERP system. So, the customizing effort increased exponentially.

The Function Point Analysis is a standard dimensional software metric having an international spread and it is applied by a large number of enterprises in the world.

Its use is well known for "ad hoc" software projects but it is a novelty when applied in the acquiring and maintenance phase of "Commercial-Off-The-Shelf" (COTS) software (whose category includes the ERP systems).

Our research describes the main cost items of ERP systems, the FPA actual use applied to ERP environment and a simple equation model for the effort estimate: this model is just a first step towards the control of the ERP system customizing effort, that will be even more accurate when a greater quantity of empirical data are collected.

Keywords

Software standard, customizing, metrics.

1. INTRODUCTION

Conceived in 1972, ERP systems offered the possibility of implementing new enterprise software systems by means of customizing software standard: the coding and testing phase, previously needed to develop "in house" software, seemed to be drastically reduced. The ERP systems could be implemented by assembling independent, yet integrated, reusable and yet operating standard modules [1].

The novelty concerns the functionalities introduction technique: the ERP functionalities are introduced by “customizing” the ERP standard modules. The "customizer" work regards the assigning of the enterprise chosen or preferred parameters to existing ERP standard functionalities. The coding phase could occur when a particular functionality, not supplied by the standard modules, was needed by the Enterprise [2].

So, the ERP system advent prospected an easy adaptability of the enterprise business process to the standard business processes of the ERP system, since they represented the "best practice" of the major enterprises operating in the same area of the customer enterprise. A marginal role was wrongly assigned to the development of "ad hoc" functionalities inside the software standard.

In fact, after the choice of the ERP system, many enterprises found it difficult to acquire the standard procedures instead of the past enterprise process. The enterprise processes, improved during the whole enterprise life, were well known by the enterprise human resources that, on the other hand, were not sufficiently involved in the changing process. So, the effort of the customizing process rapidly increased since the enterprise asked for an increasing number of "ad hoc" functionalities, that were not supplied by the standard procedures. In some cases, the exponential growth of the customizing costs led to the failure of entire projects.

Therefore, the need of a coherent estimate regarding the customizing effort was recently born.

Nowadays, the Function Point Analysis (FPA) is a standard dimensional software metric having an international spread and it is applied by a large number of enterprises in the world. This kind of metric is commonly used in estimating the dimension of "ad hoc" software projects but it is a novelty when applied in the acquiring and maintenance phase of "Commercial-Off-The-Shelf” (COTS) software (whose category includes the ERP systems).

The use of the FPA technique in measuring ERP systems has shown some practical difficulties and high costs as a consequence.

Nevertheless, the need of a coherent measure able to quantify the customizing cost (concerning the parameterization activity of standard modules and the development of non-standard functionality inside the standard software) has rapidly increased.

Using the FPA to measure the dimensional value of an ERP system within the software producer enterprise doesn’t require a particular interpretation of measurement rules because what is being implemented basically concerns the same functionalities listed in the counting rules manual. Therefore, from the producer point of view there isn't any difference in the counting practices in comparison with those rules that should be applied in other more traditional development software environments.

The final dimension of the ERP system counted from the software producer point of view will determine the software license cost [3]. So, the problem occurs when measuring the software development effort that will be added to make the standard software effective and efficient in the specific enterprise context. The main difficulty derives from the lack of a coherent estimate model about cost, time, and effort that will be added by the customizing activity.

On the basis of the research methodology mentioned in session 2, in session 3 this research carries out an analysis of the main cost items that are linked to the ERP system choice. In session 4, it is showed how the FPA can be applied to evaluate the dimension of the customizing activity for ERP systems. Finally, in session 5, a simple model is discussed; it allows the calculation of the ERP system acquiring effort. In session 6, some conclusions are drawn.

2. THE PROJECT AND THE RESEARCH METHODOLOGY

After a phase of study and analysis concerning the ERP systems, as well as its project and implementing problems, the research has focused on the problem of dimension estimation regarding the ERP projects.

The “technology independence” principle, carried out by the FPA, has determined the choice to analyse the characteristic of this dimensional software metric under the guide of FPA experts.

Further, there were several meetings with ERP project experts: these meetings showed the opportunity to validate the potentiality of the FPA through its application to real cases and the comparison of the resulting estimation values with the actual data. Finally, an equation model has been implemented, in order to derive an effort value from the dimension in Function Point of the ERP system.

3. A COST ANALYSIS OF THE ERP SOFTWARE

On deciding to choose an ERP System, an enterprise does not only buy a support system for its processes but it also adds new processes in its own organizational structure as well as modifying many of the already existing processes[1].

The new context proposed by ERP is expected to be the most suitable one as it represents “the standard” reached by the implementation of ERP systems applied to the major companies operating in the same business as the client enterprise [2]. As a matter of fact, though, enterprises are often unlikely to give up already consolidated and optimised processes. Such attitude could determine an extra effort in order to get highly demanding ERP customization. Thus, the bigger is the difference between the proposed model and the enterprise existing model, the higher will be the customization cost that the Enterprise should take into account while planning its budget; indeed, if the difference is bigger than expected, there might chance a failure of the entire project.

Therefore, a consistent budget of customization cost is needed.

With regard to the cost concerning ERP software only, two main entries are given [3]:

- *The license cost*, i.e. the cost of the standard modules supplied by the software house to the client company according to previous agreements as well as the customization tools.
- *The customizing cost*, i.e. the cost of human resources that will be employed for ad hoc modifications of standard modules, in order to meet the specific company needs.

In detail, the customization process is formed by two kinds of activities: parameterization and enhancement.

Parameterization means assigning control parameters to already existing and operating functionalities; that is, among several functional modalities, as well as data typologies and quantities provided by ERP system for standard functionalities, the customizer chooses the ones that are acknowledged by the company itself and required for its specific needs.

Enhancement means implementing a non-standard functionality when the pure customizing procedure is not able to answer the enterprise needs. Such non-standard functionality will be perfectly integrated into the ERP system, being implemented through the owner code of the standard system itself. The enhancement could concern:

- The development of a new functionality that will be added to the standard ones;
- The development of a “customer exit” that adds a further treatment or processing within a standard functionality.

The enhancement size cannot be reckoned beforehand as it depends on how much the client Enterprise is willing to adapt its own processes to the ERP standard ones.

Therefore, on buying an ERP system, an enterprise is fully aware of the licence cost, whereas it might have a wrong perception of customization cost, whose relevance becomes effective after the standard software installation.

It is then necessary to work out a coherent preventive assessment of the customization dimension and effort.

4. A COUNTING MODEL FOR ERP SYSTEMS BASED ON THE FUNCTION POINT ANALYSIS

Focusing our attention on the implementing process that will make the ERP system effective for the enterprise work system, two activities are involved: the parameterization and the enhancement activities, as previously mentioned.

On the parameterization cost estimate there are two different approaches (Figure 1): on one side new measurement methods apart from FPA are called for, on the other there are experts that wish FPA could be appointed to measure any kind of software, from “ad-hoc” to “Commercial-Off-The-Shelf” (COTS), whose category includes the ERP systems. This last approach would also enforce the “technology independence” principle carried out by the FPA in compliance with the ISO 14143-1 regulation.

The goal of this research is answering to the following question: how can the FPA be applied to the ERP project from the customer point of view?

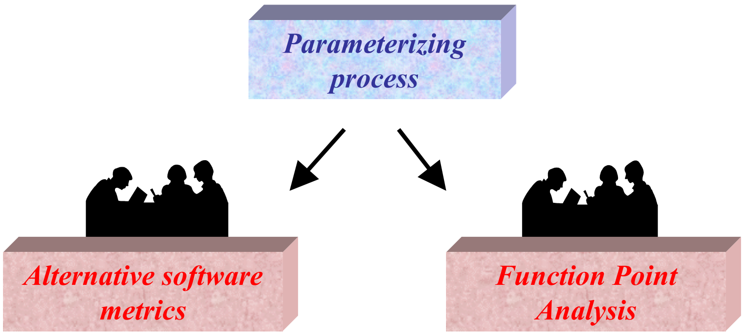


Figure 1. The two different approaches

The FPA is based on five standard elements that are:

External Input (EI), External Output (EO), External Enquiry (EQ), Internal Logical File (ILF), External Interface File (EIF)

The FPA rules that classify the typical aspects of parameterization are shown as follows: as already mentioned, the first ERP installation provides the enterprise with a standard system that needs to be customized to the specific organisation. In detail, once the company has chosen the relevant modules, for each of them the software customizer selects the proper standard functionalities.

The required standard functionalities are then installed and implemented, so that the following parameterization operates within a prototype system, in order to make each standard functionality to act according to specific needs: for example, the display of information which are not provided by the specific enterprise Data Base (DB) could be omitted.

Consequently, the parameterization has the primary intent “to alter the system behaviour”, making the system responds in a certain way rather than in another one.

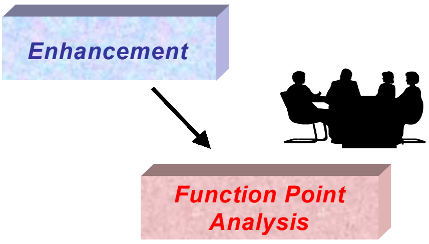


Figure 2: The Enhancement count

In the IFPUG manual an EI is defined as follows: “An EI is an elementary process that processes data or control information coming from outside the application. The primary intent of an EI is to maintain one or more ILF and/or to alter the system behaviour” [5].

For the parameterization, the customizer selects some control parameters allowed by ERP system from a fixed range and excludes some others.

The primary intent is to alter the system behaviour by making the selected functionality respond as required; as a consequence this parameterization process can be compared to the EI as an “elementary process that processes control information coming from outside the application and whose primary intent is to alter the system behaviour”, as mentioned in the IFPUG manual. To count the complexity of this EI, that represents the parameterization process, the FPA rules consider two elements: Data Elementary Type (DET) and File Type Referenced (FTR). DET represents the number of parameters altered by the parameterization process and the FTR represents the number of ILF and/or EIF referenced by the same parameterization.

Now, there is another important question: ”Is the EI of the Parameterization (EIP) an "Elementary Process" (EP), in accordance with the IFPUG definition?

According to the FPA rules, an "EP is the smallest activity unit meaningful for the final user". It also "must be independent and leave the application measured in a coherent functional status"[5]. Usually, the EI is to be found within one single application, whereas the EIP crosses and involves more than one application modules before achieving a meaningful sense for the final customer and therefore before leaving the application in a coherent functional status. In fact, the ERP is an integrated system and, when a functionality must be parameterized, more modules are involved within the EIP.

In conclusion, it has been found among the FPA rules, the way to measure the parameterization process. In particular, it has been found a correspondence between the EI (illustrated, in the manual, as a type of EP) and the EIP.

On the other hand, it has been found a difference between the EP's boundary and the usually wider boundary of EIP. An EIP that crosses more than one application is not an obstacle as to the counting rules, it is rather an obstacle from the point of view of the practical application of such rules. In fact, it is rather difficult to find out all modules involved within the EIP boundary.

Moreover, an EIP involves a higher number of DET, i.e. parameters, compared with the number of DET counted by a traditional EI meant for ad-hoc applicatons, as it is shown in Albrecht’s complexity charts.

Furthermore an EIP interests a number of DET considerably elevated compared to the number of DET interested from a traditional EI to apply an “ad-hoc”, so as expected in the chart of complexity of Albrecht.

In conclusion, therefore a EIP can be considered as an EI in all the effects, it is necessary to lay out a suitable specific chart of complexity for the EIP. In the formula of the FPA we need to introduce a new: the External Input of Parameterization, as a sixth functional standard added compared to those individuate from the FPA traditional.

In definitive in the formula of the FPA we need to introduce a new: the External Input of Parameterization, as a sixth functional standard added compared to those individuate from the traditional FPA.

In this way the FPA will be based on six standard elements which are:

External Input (EI), External Input of Parameterization (EIP), External Output (EO), External Enquiry (EQ), Internal Logical File (ILF), External Interface File (EIF).

Metric experts recognise the FPA rules correct and coherent to be applied to all those activities known as “enhancement” (Figure 2). Actually, the enhancement is a maintenance activity that adds new functionalities to the commercial software.

Such functions, in the area of this study, have been classified in:

- Non-standard functionalities required
- Reports
- Interfaces
- Conversions

The functions of no-standard requested correspond to those in which the manual IFPUG are defined as transaction function, and can be either type EO, or an EI or an EQ.

The function of the report type are EO, since they are functions of transaction of questioning that are not limited to a simple require of data.

The function of the interface type corresponds to the EIF or better to the functions of types created from the external from the application object of calculation and from them reference. Furthermore such interfacce can be transitory or definitive.

The function of conversion type is an EI and it is a transaction function which has only one purpose: the migration of the data from the precedent files to the new ones.

The empirical data which has been collected showed that all the functions described have an average complexity, to the exclusion of the interfaces which have a low complexity.

For the enhancement, therefore, we apply the traditional FPA rules based on the five elements.

5. ESTIMATION USING EQUATION

The International Software Benchmarking Standards Group (ISBSG) is an organisation that helps improve the management of Information Technology resources, by both business and government, through improved project estimation, productivity, risk analysis and benchmarking. ISBSG maintains and exploits two repositories of software project metrics:

- Software Development and Enhancement
- Software Maintenance and Support

These Repository data can be used to estimate software project metrics using three macro-estimation techniques, as suggested by ISBSG: estimation using equation, estimation using comparison, estimation using analogy.

The FPA represents the fundamental ingredient of the ISBSG estimation using equation. This technique involves the use of regression equations and is commonly used to produce indicative project estimates early in the life of a project. This technique is not sufficiently accurate to produce an estimate that could be relied on for quoting or business case requirements. ISBSG suggests a "ballpark" estimate that can be used for an early indication of whether a project idea is feasible, or when you are short of time and detailed information.

To produce a project metric, it is required to insert the functional size of the project into the appropriate equation whose choice depends on the technological environment. These equations are periodically updated conforming with the new development software environments and with the new technologies that support them.

The ISBSG has not still elaborated a regression equation to calculate the customizing effort starting from the FP count for ERP systems, since it has not arranged sufficient historical dates. This has occurred from the fact of the attempt of measurements of the effort of personalising commercial systems (COTS), in particular ERP, is up to date.

One of the goals of this research is to write a suitable medium equation, supported by empirical data, able to estimate the effort (person/days) related to the customizing phase of the ERP systems.

It has been collected actual data through meetings with ERP project experts.

As shown before, the customization process mainly includes two different kinds of activities: parameterization and enhancement.

To this ones they must be added further ones having a complementary role (e.g. testing and training phase). To correctly estimate a project metric of an ERP system from the user point of view, they must be determined, among the different cost voices, those ones that could be configured as the independent variables of the medium equation. The other cost voices could be represented as the variable depending from the independent variables. For example, the cost of the test phase depend on the customizing and on the enhancement activities.

The analysis of the empirical data regarding some ERP projects has quantified an estimation of the parameters of the following medium equation:

$$Y= \alpha P+\beta E$$

Y is the effort value (person/days) of customizing. The depending variables are:

· P: is the parameterization size (function points of EIP type)

· E: is the enhancement size in terms of number of functionality added to the new system (function point of EI, EO, EQ, ILF, EIF type)

The coefficient α and β are respectively the average effort, in person/days, for each Function Point of parameterization and for each Function Point of enhancement. Is has been noted a value of β higher than α:

$$\beta \cong 1{,}5\ \alpha$$

Therefore the effort needed to produce a function point of enhancement is higher than the effort needed to produce a function point of parameterization. The logical reason must be found in the different nature of the two activity i.e. enhancement and parameterisation as previously described.

This result is just a first step towards the control of the ERP system customizing effort, that will be even more accurate when a greater quantity of empirical data will be collected.

6. CONCLUSION

On deciding to choose an ERP System, the new context proposed is expected to be the most suitable one. As a matter of fact, though, enterprises are often unlikely to give up already consolidated and optimised processes. Thus, if the difference between the proposed model and the enterprise existing model is bigger than expected, there might chance a failure of the entire project. Thus it is necessary to work out a coherent preventive assessment of the customization dimension and effort.

The main cost voices concerning ERP software are the license cost and the customizing cost. This last voice involves two activity: parameterization and enhancement. On buying an ERP system, an enterprise is fully aware of the licence cost, whereas it might have a wrong perception about customizing cost whose dimension is willing to increase if the enterprise isn't able to adapt itself to the standard ERP procedures.

In this research it has been shown a way to estimate the customizing dimension through the FPA: actually the parameterization has the primary intent "to alter the system behaviour", so it can be compared to the EI mentioned in the IFPUG manual. But a boundary of an EI of Parameterization (EIP) is usually wider than a boundary of a traditional EI.

Moreover, an EIP involves a higher number of DET and FTR in comparison with the number of DET counted by a traditional EI. This is not forbidden by the IFPUG rules but it is necessary to lay out a suitable specific chart of complexity for the EIP.

The enhancement is a maintenance activity that adds new functionalities to the standard software. The enhancement involves four main cost voices: non-standard functionalities required (they can be EO, EI or EQ according with their nature), reports (EO), interfaces (EIF) and conversions (EI).

The empirical data which has been collected showed that all the functions described have an average complexity, to the exclusion of the interfaces which have a low complexity.

The International Software Benchmarking Standards Group (ISBSG) maintains and exploits two repositories of software project metrics to estimate software project metrics using three macro-estimation techniques: one of these is estimation using equation.

To produce a project metric, it is required to insert the functional size of the project into the appropriate equation whose choice depends from the technological environment. The ISBSG has not still elaborated an equation to calculate the effort starting from the FP for the ERP systems.

This study exploits an equation to estimate the customizing effort for ERP systems; according with this result, the effort needed to produce a function point of enhancement is higher than the effort needed to produce a function point of parameterization. The logical reason must be found in the different nature of the two activity i.e. enhancement and parameterisation as previously described.

This is just a first step towards the control of the ERP system customizing effort, that will be even more accurate when a greater quantity of empirical data are collected.

The equation is very easy to use: it could be applied to any kind of customizable standard software i. e. Workflow Management Systems (WMS), or Data Warehouse (DW) since their new entry like "customizable standard software" is growing over the world.

7. LITERATURE

[1] Wilhelm Scheer, A, Habermann, F. (2000). Making ERP a success. Communications of the ACM.

[2] Blain, J. (1999). ASAP World Consultancy: Il manuale di SAP R/3. Jackson Libri.

[3] Amiconi, F. Beretta, S. (1998). Information Technology e creazione di valore – Analisi del fenomeno SAP. E.G.E.A. S.p.A. Milano.

[4] Soh, C. Kien S. S. et al. (2000). Cultural Fits and Misfits: is Erp a Universal Solution? Communications of the ACM.

[5] Garmus, D. H. (2000). Function Point Analysis - Measurement Practices for Successful Software Projects. Addison-Wesley Information Technology Series.

[6] Buglione, L. (2003). Misurare il software. Franco Angeli.

[7] Natale, D. (1996). Qualità e quantità nei sistemi software. Franco Angeli/Informatica.

[8] Cavallo, A. (1998). Studio di tecniche e metodologie per la valutazione di prodotti commerciali di Workflow Management. Edizioni Kappa

[9] Capers, J. (1996). Applied Software Metrics. McGraw Hill.

[10] Capers, J. (1998). Estimating Software Costs. McGraw Hill.

8. AUTHORS CVS

Margherita Martellucci

She has taken the Economy degree at the University of Rome "La Sapienza" after discussing a thesis whose title was: "Reengineering Information System: characters and opportunity of ERP solutions". After the degree she begun a study and research activity financed by the Pisa Research Consortium on the correlation between the qualitative and quantitative aspects of the software measurement. She is Certified Function Point Specialist (CFPS).

Currently she collaborates with the Department of Mathematics for the Economic decisions, Insurance and Financial Institutions and is attending the doctorate research cycle in "Organisation of the Enterprise Information System".

Anna Cavallo

She is assistant professor at the Dipartimento di Matematica per le decisioni economiche finanziarie ed assicurative, faculty of Economics of the University of Rome “La Sapienza”. Her main research interests include: software metrics, software economics, e-learning. She is the author of several papers, publications and book on these research topics. She is teaching since 1996 on Business Information System, Information System and Informatics, Informatics Tools at the faculty of Economics of the University “La Sapienza”. She is teaching the Informatics Management of Business Data at the University of Rome LUMSA (Libera Università Maria Santissima Assunta). She is a member both of GUFPI (Italian Function Point Users Group) and an ordinary member of AICA (Associazione Italiana di Calcolo Automatico). She is Certified Function Point Specialist (CFPS).She holds the “Certificazione di Verifica Ispettiva Interna AICQ”. She received the Laurea in Mathematics from University of Rome “La Sapienza”.

Francesco Maria Stilo

He is full professor of both Information Computer System and Information System Organisation at the Dipartimento di Matematica per le decisioni economiche finanziarie ed assicurative, faculty of Economics of the University of Rome “La Sapienza”. His main research interests include: software economics, e-government, e-learning. He is the author of several papers, publications and book on these research topics. He is a member of GUFPI (Italian Function Point Users Group), an ordinary member of AICA (Associazione Italiana di Calcolo Automatico), a member of ACM (Association of Computing Machinery). He is one of the main experts of the Italian Public Administration concerning the Information Systems. He received the Laurea in Statistics from University of Rome “La Sapienza”

Domenico Natale

He has been responsible for WorkFlow & Document in Sogei, a large IT company in Italy dedicated to the Ministry of Finance, since 1979. He has been a member of IFPUG since 1994 and in particular served for three years on the Board and is currently on the Academic Affairs Committee. He is Certified Function Point Specialist (CFPS).

He is involved in ISO (International Organization for Standardization), GUFPI (Italian Function Point Users Group) and ISBSG (International Software Benchmarking Standard Group). He has given presentations on software metrics at several national and international conferences.

Natale holds a degree in Statistics and a Master’s in Computer Science from the University of Rome. He is the author of several papers, publications and a book on quality and quantity in Information Systems. He is also personally on the web, participating in the “new culture” of internet (www.mondomatica.it).

Nicoletta Lucchetti

She has worked in Sogei, a large IT company in Italy dedicated to the Ministry of Finance, since 1985, where she coordinates a “virtual team” of 30 Certification Function Point Specialists. She also works on the Metrics Project regarding Code and Applications quality. She has been a member of IFPUG Certification Committee since 2000. She is Certified Function Point Specialist (CFPS).

She is involved in ISO (International Organization for Standardization) and GUFPI (Italian Function Point Users Group). Nicoletta Lucchetti holds a degree in Economics from the University of Rome. She is the author of several manuals on the application of Function Point in her company.

eXPERT Approach Implementation in Software SME

Sylvia Ilieva, Penko Ivanov, Ilian Mihaylov, Eliza Stefanova, Avram Eskenazi
Sylvia@acad.bg, Penko.Ivanov@rila.com, Ilian.Mihaylov@rila.com, eliza@fmi.uni-sofia.bg,
eskenazi@math.bas.bg

Abstract

This article presents software development approach developed within the EC IST 34488 *e*XPERT project, which is systematic combination of Extreme Programming (XP) and Personal Software Process (PSP). In particular, the article describes an experiment under a project in Rila Solutions (a Bulgarian software company). The experiment consisted in performing a detailed gap analysis between the existing certified development process and the eXPERT approach, ranking the differences and showing the ways to overcome them. The article further outlines the advantages of applying the eXPERT approach, as well as the lessons learnt.

Keywords

software development process, eXtreme Programming (XP), Personal Software Process (PSP), eXPERT approach.

1 Introduction

The latest studies on the well-known software development methodologies and their applicability to e-projects reveal that these methodologies do not fit very well the priorities, the abilities and the corporate culture of SMEs developing e-business or e-commerce applications. Traditional methodologies (incl. those more recent of an evolutionary type) assume that software engineers could more or less anticipate the complete set of requirements early enough and reduce cost by eliminating change. This result can be reasonable in stable environments, but it is not the case at all in e-projects. E-project as defined in [4] is a project which: must be delivered rapidly; is both research-like and mission critical; have to be managed in a turbulent business and technology environment. In the fast changing environment of e-projects change cannot be eliminated. Rather than eliminate rework, the new strategy is to try to reduce its cost, while retaining quality.

The main objective of EC IST 34488 project eXPERT was to define and provide SMEs with a powerful and lightweight approach for software development of e-business applications. This approach is applicable to small teams developing projects characterised with often changing requirements, tight schedules, and high quality demands. Its aim is to facilitate all related activities with crucial importance for a project success: pure development, management of time, changes, quality, customer relationships, and professional growth of the employees. The approach is called *e*XPERT and combines principles of eXtreme Programming (XP) [1,7] and the Personal Software Process (PSP) [5,3].

The eXPERT project consortium established a network of Centers of Expertise (CE) in Spain, Germany and Bulgaria, and user companies associated to each CE. Seven pilot projects are run at the user companies who experimenting the application of the defined approach to e-project development. The case study this paper describes is one of those pilot projects – the Bulgarian company Rila Solutions.

The article is organised as follows: Section 2 presents eXPERT approach, which is based on XP and PSP. Section 3 describes the Rila Soutions' development environment with major outcomes of the gap analysis performed in Rila Solutions and experimental results. Section 4 discusses lessons learnt. Section 5 concludes the paper.

2 eXPERT Approach Basics

2.1 eXtreme Programming (XP)

Extreme Programming is a lightweight, efficient, low-risk, flexible, predictable and scientific way to develop software [1]. XP is a discipline for software development with values of communication, simplicity, feedback and courage.

The recommended practices in applying Extreme Programming are [7,9]:

- On-site customer
- Small releases
- Planning game
- Simple design
- Pair programming
- Refactoring
- Test-first

Collective ownership

Continuous integration

Metaphor

Coding standards

40-hour week

Stand up meetings

Iteration retrospective

XP emphasizes on the team style of work, understanding the team as consisting of customers, man agers and developers. Involving the customer in all the processes of software development is one of the reasons for having a successful product delivery in time and with a happy customer.

Of all the lightweight methodologies, XP tends to be best accepted by the e-project developers due to the simplicity of its rules and practices, its flexibility to changes in a project run, refactoring orientation and collaboration ideology.

2.2 Personal Software Process (PSP)

PSP is a structured framework of forms, guidelines and procedures for developing software [3,5]. The PSP is a self-improvement process designed to help software developers to control, manage and improve their way of work. It educates them to estimate and plan their work and to do this before committing to start doing the job. The first step in the PSP process is planning. In order to make better plans software engineers have to gather data on:

- the time they spend in each task - using time recording log,
- the sizes of the products they produce - the principal measure is lines of code (LOC) but any size measure can be used if it provides a reasonable correlation between development time and product size
- the quality of these products – using defect recording log where there is data about injected defect, when they are injected, when they are found and fixed, how long it took to fix them.

The PSP planning process itself includes: requirements, conceptual design and estimation of product size and resources. PSP, similarly to XP, focuses on building quality products and tracking quality from the initial development phase, instead of checking it right before delivery. PSP quality management is based on two principles – early defect removal and defect prevention. Last but not least, PSP like Capability Maturity Model (CMM) [8] is organized in levels:

- PSP0 – the baseline process
- PSP1 – the personal planning process,
- PSP2 – personal quality management process
- PSP3 – a cyclic personal process

The PSP applies not only to critical software development phases as design, code and test but also to other phases like requirements engineering, product maintenance, documentation development and others.

2.3 The eXPERT Approach

Although the XP practices seem to be very simple, they require strong individual and team discipline in order to achieve good results. There are a number of reports about XP experiments showing that they have failed due to developers' reluctance or incapability to apply the practices in a disciplined and professional manner. The incapability aspects are mainly related to making wrong estimations of individual work and failing to create a correct plan of the tasks that have to be performed. It seems that complementing XP with the PSP is a good way for resolving the problem. Even more, the PSP could also contribute to coping with the problem of reluctance. That is why the eXPERT approach is built on two well-known software development approaches XP and PSP.

The eXPERT approach [2] is described by means of five processes: Customer Requirements Management, Project management, Design, Code and Test. The processes are described in terms of *Activities*, *Tasks* that have to be done to complete an activity, and responsible *Role* for performing a task. The activities described into the *e*XPERT approach [6] are based on the enumerated XP practices. Certain modifications are introduced mainly related to measuring the effectiveness of these activities and the defect rates. This measuring is needed in order to identify problem causes and to eliminate them in the future. The logs for collecting project data follow the templates and the principles of PSP, but are modified as to fit the XP method, in particular to reflect its specifics that developers work in pairs and that design, testing and coding processes are strongly interrelated and executed in parallel. *e*XPERT approach defines the following roles: Customer, Project Leader and Developer. They are very close to the roles as defined in XP (Programmer, Customer, Coach, Tracker, Tester, Consultant, Big Boss), with some additional responsibilities coming from the application of the PSP practices. More than one role can be assigned to a single team member into a project. In such a case it is important that this team member has the necessary knowledge and skills as well as the time for playing all the roles assigned to him.

The eXPERT approach, if compared to XP, seems to be more structured, easier to be understood and applied by SME's because of the process oriented architecture. At the same time it is not as light as XP (especially for developers) because of the need to measure activities and tasks to be done - each one of the processes contains one special activity, which goal is to measure effort and time spent on tasks into the process. The idea for the measurements came from PSP, but in all other directions the eXPERT approach and PSP seem to be incomparable. All measures in the eXPERT approach are performed only when they are necessary, and their main goal is to optimize the time and cost planning. These measurements itself should be not so heavy extension for the SME's applying the eXPERT approach, if specialized tools for gathering and analyzing data from the measurements are available. That is why eXPERT could be considered as a light but well balanced approach.

3 Rila Solutions Context of Applying Expert Approach

3.1 Reasons and Prerequisites

At Rila Solutions, as in most software developing SMEs, the necessity for providing high quality software solutions has increased enormously. Clients need both to have a feasible system as soon as possible and to be able to constantly amend the functional scope. The need for flexibility of the software development process apparently has become one of the major issues in the new development practices. Delivering software solutions is hard, but delivering quality software on time and budget is even harder. The new trends in software development are more than clear: to deliver high quality faster with smaller budget.

Rila Solutions' Development Process (developed and applied by the company) is flexible enough to allow the adoption of different Project Management (PM) methodologies depending of the matter of the project. Some of Rila Solutions projects (a small number at present) are characterized with well written specifications clear understanding of customer needs and the means to achieve this. These projects have well defined business models, technologies and goals. The technology used is well described and no variations upon the solution development are possible. In the rest of projects (possibly around 70%), changes are most likely to occur because the system definition is not stable enough. Rila is working to improve its PM process and to adopt world best practices, which should allow delivering solutions on time, within the budget and with the desired functionality. The eXPERT methodology was applied experimentally to such kind of projects.

To make the current situation clearer, we should add that Rila Solutions uses a Quality Management System, which is ISO 9001:2000 certified. It also has an impact on the application of new methodologies, and consequently on the gap analysis, which is the first step towards discovering the discrepancies between the existing Rila Solutions Software Development Process and the eXPERT approach. Description documents, standards, as well as dedicated questionnaires were used as sources to this analysis.

3.2 Gap between Rila Process and eXPERT Approach

Prior to applying the eXPERT approach in the company, Rila Solutions performed gap analysis for the differences between the current process and the approach that should be applied. The subject matter of the present analysis is the implementation of the new process of development in a professional service company.

The process of gap analysis was a typical business reengineering process. Business objectives were set for the following types of e-projects:

- Projects that do not have properly specified requirements at the beginning of the project
- Projects for clients that do not have a detailed vision of the software they need from the beginning
- Projects, detailed requirements of which cannot be entirely gathered at the analysis phase.

The gap analysis discovered various types of differences between the current project lifecycle and the project lifecycle that will become as a result of applying the eXPERT in the company. All the changes in the process that should take place were carefully analyzed and described in a “GAP Analysis” document that was presented to the top management and the team members. A short course on the principles and the practices of the eXPERT and its implementation in the company was carried out in order to prepare the team for the new framework and discipline required by the approach. The major outcomes of the process of Gap Analysis are:

- The project goals are not changed. Process phases remain the same. The changes take place in the sequence of the tasks within the phases, roles and intermediate outcomes.
- Control and status reporting – the granularity of data and the number of tracked data are significantly different according to the PSP requirements. New tracking and controlling tools are involved in the process.
- The total number of project documents was reduced by 7 tracked documents.
- The most effort consuming documents: “Basic Solution Specification” and “Detailed Solution Specification” were lightened or modified and become much easier for preparation. This led to less time for documents writing and lower total project cost and shorter time prior starting the development of the e-project having the customer available.

3.3 Experiment Results

As a result of the gap analysis, Rila Solutions has developed a tailoring guide on how the new approach should be implemented and used within the company. This tailoring guide was presented to all team members that were involved in the pilot project using the eXPERT methodology. A short course on the principles and the practices of the eXPERT and its implementation in the company was carried out in order to prepare the team for the new framework and discipline required by the approach.

The selected project was conducted strictly within the framework of the new tailoring guide and the eXPERT, hereinafter referred to as “the pilot project”. For the purpose of ensuring comparability of results, Rila Solutions used the time and effort tracking system, already implemented by the company, which keeps information on all activities performed by the project members on each project as well as variations on the time schedule and effort.

One of the most critical factors applying new methodologies is the adequate baseline that will be used for comparing the newly achieved results and measuring whether and how the company had achieved the desired benefits and what is the actual profit. Prior applying the eXPERT for the first time we were aware that correct result comparison may be achieved if we apply it on one and the same project with same team members that do not use the know-how collected during the first implementation. As the matter of fact this situation cannot be created and the following steps were taken:

A project with identical technology and similar project achievements was selected to serve as a baseline project. For the correct selection of the baseline project, several other metrics were used for comparison like Lines of Code (LOC), number of database tables, number of user forms and controls. The purpose was to select the best match for a baseline project with similar achievements and activities to the ones that will be performed in the pilot project. The data for the baseline project was taken out of the time tracking system and some of the results are presented in the figures below.

Figure 1 presents the effort that was invested in the development of the pilot project using the eXPERT approach versus the effort used in the development of the quite similar baseline project. The result is decreasing the overall effort on all iterations and as a whole – for the entire project.

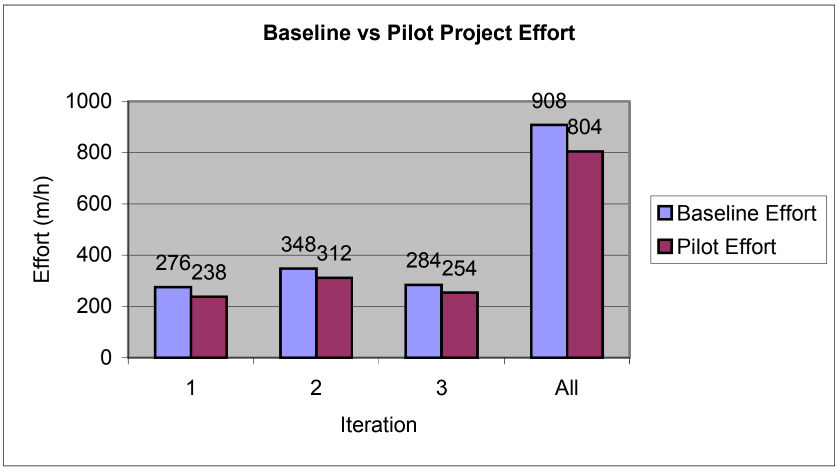


Figure 1: Decreasing the development effort by iterations with the eXPERT approach

Figure 2 presents the overall decreasing of the development effort for the pilot project according to the baseline project in percentage. It is created on the basis of the data represented in Figure 1. The negative effort deviation is a result of the less effort that was used in the implementation of the pilot project than the baseline with similar achievements.

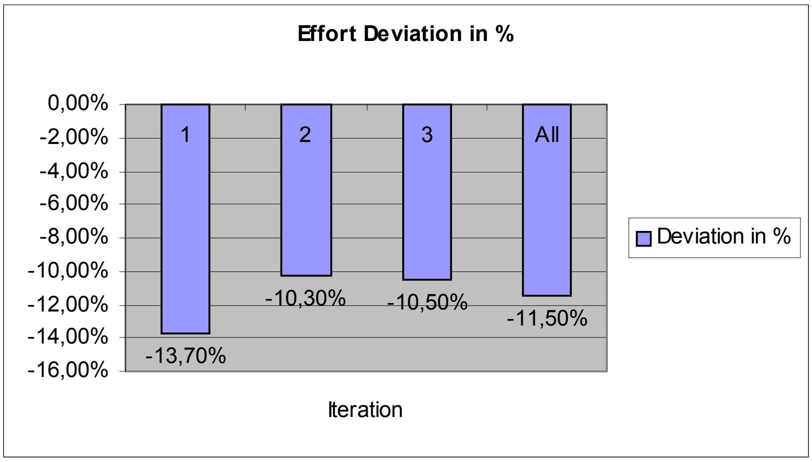


Figure 2: Effort deviation in percentage by iterations using the eXPERT approach

Apart of these measured results gathered during the pilot project implementation the whole lifecycle of the project was influenced by the eXPERT practices. Having the customer available “on-site” helped us to minimize the task “Gathering User Requirements” duration and move the start of the development phase earlier in time. The reduced document flow and testing practices from the XP also helped us to save some effort and bring the project to earlier and successful project ending.

4 Lessons Learnt

SMEs planning to apply the eXPERT approach should keep in mind some of the steps in Rila experiment as an example. They could repeat at least two of them: to identify whether they really need the eXPERT approach (Rila Solutions’ approach was to prepare a questionnaire for all employees to analyze current process weaknesses) and to train the staff on how to apply eXPERT approach.

It is essential for the project success that the appropriate selection of the staff in general is carefully performed. When the company tries to implement a new methodology the management should choose an appropriate pilot project. Young and enthusiastic people should carry out the pilot project. The training and the adapting to the new approach are easier. If part of the team is already trained then the training of the new developer(s) involved in the team could be done on the way.

4.1 Benefits

Using eXPERT approach in Rila Solutions' real project has shown that its benefits are the shorter time for customers' feedback, the reduced number of documents and the simplification of some documents. The project documentation is kept to a reasonable minimum with the application of the eXPERT approach. At the same time, the total time needed for shipping the product to the customer is reduced.

The improvement of the discipline is considered to be the most important benefit from the eXPERT implementation. All measurements and records stimulate better discipline of developers. The eXPERT approach positively influences developers and, as a final result, there is an organizational and business impact.

Applying the eXPERT approach decreases the time and effort spent on design mainly because there is no detailed design documentation. From cost perspective, the cost reduction comes from the fact that there is no need to have high-qualified specialists for analysis of requirements and the design of the system developed and to spend long time for interviewing the customer on their side.

4.2 Specifics

Applying the eXPERT approach in Rila Solutions led to some specific peculiarities and changes to the practices described in the pure eXPERT definition. These changes resulted from the developers' and project management view on how to apply the approach to real projects.

Pair programming is used also as a way to improve the skills for one of the developers. But in cases when team members work on other company projects at the same time, applying pair programming all the time requires additional efforts to ensure the co-ordination of pair members.

Unit tests (automated tests of class or cluster of classes and their functionality), which are important part of Test-first practice, are performed easily and mainly for business logic and they are not applied or difficult to apply for the user interface part.

The defect logs has shown that the application of Test-first practice as part of the eXPERT approach leads to reduction of the errors' appearance in later project phases, reduction of not fixed errors and finally to reduction of the total time for the development and increased quality of the product.

The Customer on-site practice could not be applied as described in XP and eXPERT. In Rila Solutions experiment, it is unrealistic to have the customer present at the team office during the development – the communication with him is achieved through meetings on the customer site, phone calls, e-mails, as well as providing him with web-access to the application. The developers' team is certainly the initiating site in that communication.

It is difficult to plan in an e-project from the beginning how many releases will be performed and how much iteration will be needed in each release. When the customer performs a constant monitoring on the project and its results, the new ideas on the functionalities come on the way.

Using strict Coding standards as recommended by XP makes the code readable and understandable for all project members. Several code reviews should be performed for in order to fully adopt this practice and reduce the time for further refactoring of the code smells.

In general, the recommended practices are applicable to particular projects but, depending on the project specifics, they have to be adjusted for others: most of the eXPERT practices can be introduced separately, but their full effect is reached only when using all of them. In our view, the team should start with coding standards (in most software SME-s they already exist) and could continue with unit testing, refactoring, pair programming and so on – finally finishing with customer on-site (the most difficult and unrealistic practice from XP). According to the pilot experiment practices coming from PSP are mandatory and are applied as they are.

4.3 Difficulties

Nevertheless some of the practices in the approach were found to be unrealistic to some extent for use in the Rila Solutions pilot project (apart from customer on-site) and they are reported as not used eXPERT practices. These are as follows:

- Design checks - could be substituted by very informal peer reviews
- Pair programming is difficult to be applied 100% - Developers find it very hard to work 40 hours a week at such heavy and exhausting paces, especially if they use pair programming all the time and have never used XP practices before. Nevertheless pair programming is vital for the team discipline and should be applied as much as possible.
- Test-first is difficult to be applied 100% (there are cases where tests are more time consuming than useful)

The general conclusion derived from the detailed and ranked results shows that the gap between the Rila Solutions' processes and the eXPERT approach is essential. The main reasons seem to be the very important difference in the granularity of the processes, as well as the typical top-down approach of Rila Solutions. However we can say that the gap existing with the eXPERT approach has successfully been overcome due to the flexibility of the Rila Solutions approach.

We also have to note that factors not directly connected to the application of the eXPERT approach influence its success. The most important among them are the willingness and readiness of the staff to try to apply the eXPERT approach in appropriate projects, the explicit interest of a number of staff members in eXPERT and the high qualifications of the persons directly involved of the experiment.

5 Conclusions

The eXPERT approach addresses various aspects of e-project development and according to the experience we acquired - helps to deal with the respective arising problems like imprecise and slipping project scope, effective communication, integration of the customer into the project development process, project estimation and planning, following sound personal and team discipline principles. These features make it particularly appropriate for a wide range of e-projects. The adoption of this approach creates new professional and personal values that are necessary for SME to enter the digital economy - flexibility, quality, efficiency, communication skills, as well as working with uncertainty, intangible assets, rapidly changing requirements, and the crippling cost of failure.

In SME's with processes comparable with Rila process, the expected benefits from the eXPERT approach are the reduced number of documents produced during the project implementation and the shorter time for feedback. As a result, the time for the whole project development is reduced. From cost prospective, the expectations are to reduce the cost at least because of the elimination of the need to have high-qualified specialists for analysis of requirements and the design of the system developed.

The biggest difficulty in the eXPERT approach implementation appeared in the necessity for introducing some changes in the people's culture and mindset. It seems that a lot of efforts will be necessary to acquire the new knowledge needed, to formulate the new rules, to start thinking differently and to stick to a new discipline. This has been the case in Rila where established processes were quite different and new specific changes were introduces during the adoption of the eXPERT approach and the requirements of the PSP.

Acknowledgments

This work was the result of the IST-2001-34488 project, called “eXPERT“, that was funded by EC 5FP.

6 Literature

1. Beck K., “Extreme Programming Explained: Embrace Change”, Addison-Wesley, 1999
2. Bozheva T., Deliverable D1 – eXPERT approach, July 2003
3. Hayes O., “The Personal Software Process: An Empirical Study of the Impact of PSP on Individual Engineers”, Carnegie Mellon, 1997
4. Highsmith J., "e-Project Management: Harnessing Innovation and Speed", e-Business Application Delivery Journal, Vol. 1, No. 1, Cutter Consortium 2000
5. Humphrey W., “A Discipline for Software Engineering", Addison-Wesley, 1995
6. Ilieva S., E. Stefanova, eXPERT approach for e-business software development, International conference “Basic Technologies for E-business’2002”,16-18 September, Albena, 2002,
7. Jeffries R., A. Anderson, C. Hendrickson, „Extreme Programming Installed”, Addison-Wesley, 2000
8. Paulk M., B. Curtis, M. Chrissis, C. Weber, “The Capability Maturity Model for Software”, SEI, 1999
9. Westphal F., J. Eckstein, Extreme Programming course, 2002

7 Author CVs

Sylvia Ilieva

M.Sc. in Electronics, Technical University - Sofia, Bulgaria, Ph.D. in Computer Science, Technical University - Sofia, Bulgaria. Currently she is an Associate Professor at Sofia University. She has a good record of successful participation in a number of European RTD Projects, such as: ACTS project W4ATT; ACTS project NICE; INCO project MONHX; INCO project IDEALIST-EAST; and 5FP projects IDEALIST-5FP, ECUA+, eXPERT, DIOGENE. She delivers lectures in: Mobile computing; Software technologies; Object oriented programming in Internet; Tools for integrating server software applications, UML.

Penko Ivanov

M.Sc. in Computer Science, Sofia University, Bulgaria. Started as a software developer in 1998 in a German Company in Bulgaria. In 1999, joined RILA Solutions as a Java developer. Currently working as a technical leader of software projects. Worked as software architect for some of the company major projects oriented in the J2EE development. Involved in the application of the eXPERT approach in the company development process

Ilian Mihaylov

M.Sc. in Biotechnlogy, Sofia University, Bulgaria. Started in 1986 as developer at SPS Sofia. In 1999, joined Rila Solutions as C++ developper, since 1999-2002 he has been Group Director of e-business department. At Rila Solutions, he participates in more than 20 projects at different senior’s roles (PM, SA, BA, TL) besides his administrative functions.

Eliza Stefanova

M.Sc. in Computer Science, Sofia University, Bulgaria. Started in 1989 as a programmer at theSofia University Computer Center. During the 1990-1999 she was a head of divisions of Network & Communication and Advance Telematics Technologies. In 1999, she joined the Department of Information Technologies at Sofia University as an Assistant Professor. She also performed research activities in the field of computer networks and applications. She participated in a number of research projects and 5FP projects. Currently she is a Senior Assistant Professor.

Avram Eskenazi

M.Sc. in Mathematics and Computer Science Sofia University, Bulgaria, PhD in Mathematics and Computer Science, Sofia University. Currently - Head of the Software Engineering Dept. at the Institute of Mathematics and Informatics, Bulgarian Academy of Science, as well as Professor of Informatics in the University of Economics, Varna, Bulgaria. Main fields of scientific interest - software quality, information technologies in education

Improving Software Configuration Management for Extreme Programming: A Controlled Case Study

Juha Koskela, Jukka Kääriäinen and Juha Takalo
Technical Research Centre of Finland, VTT Electronics
P.O.Box 1100, FIN-90571 Oulu, Finland
{Juha.Koskela, Jukka.Kaariainen, Juha.Takalo}@vtt.fi

Abstract

Extreme programming (XP) is currently the most popular agile software development method. It is as its best for small teams developing software subject to rapidly changing requirements. Software configuration management (SCM) is a method of bringing control to the software development process. SCM is known as an indispensable activity that must take place whenever developing software. It is inseparable part of quality-oriented product development regardless of development method. Existing studies show that SCM is partially addressed via XP's collective ownership, small releases, and continuous integration practices. However, currently there exist very few empirical data on SCM exploitation in XP. This paper reports results from a controlled extreme programming case study supported by well-defined SCM activities and tools. Results show that SCM activities and tools, when properly used, provide essential support for XP development process and its practices.

Keywords

Agile methods, extreme programming, software configuration management

1 Introduction

Extreme programming (XP) developed by Kent Beck is currently the most popular agile method [e.g. 1,2]. XP like other agile methods focus on generating early releases of working products. They aim to deliver business value immediately from the beginning of the project. Software configuration management (SCM) is a method of bringing control to the software development process and has been proved to be an invaluable part of developing high quality software [3]. It is also known as an indispensable activity that must take place whenever developing software [4] and, therefore, it can be seen important also in XP and other agile methods. Paulk [5] presents that SCM is partially addressed in XP via collective ownership, small releases, and continuous integration. However, traditional definition divides SCM into configuration identification, configuration control, configuration status accounting and configuration audits [6, 7]. Thus, XP's approach to SCM can be seen implicit and incomplete.

Currently there exist very few empirical data of SCM exploitation in XP. This paper reports results from a controlled extreme programming case study supported by SCM. A team of four developers was acquired to implement a system for managing the research data obtained over years at a finish research institute. As the project team had no earlier experience on XP or SCM, they were given two days practical training before the start of the project. The focus in this paper is on reporting the SCM experiences and observations found during the project. In this case software configuration management implementation was taken into account right from the beginning of the project. Results show that SCM activities and tools, when properly used, provide essential support for XP development process and its practices.

The paper is organized as follows. The following section introduces the XP method, software configuration management and related research. This is followed by a description of the research methods, research settings, the results and the discussion. Lastly, section six concludes the paper.

2 Related Research

This section briefly introduces Extreme Programming method and software configuration management. In addition also the current knowledge of SCM in XP is presented.

2.1 Extreme Programming

Extreme programming (XP) developed by Kent Beck is an agile method for teams developing software subject to rapidly changing requirements. It is focused on delivering immediate business value to the customers. The XP process can be characterized by short development cycles, incremental planning, continuous feedback, reliance to communication and evolutionary design [8]. According to Beck [9] rather than planning, analyzing, and designing for the far to the future, XP suggests to do all of these activities at a little at a time, throughout software development.

Primarily, XP is aimed at object-oriented projects using at most dozen programmers in a one location [10]. According to surveys, XP is currently the most popular and well-known method in the agile family of methodologies [e.g. 1, 2]. It is made up of a simple set of common-sense practices. In fact, most of XP ’s practices have been in use for a long time and therefore they are not unique or original. Many of them have been replaced with more complicated practices in the process of time, but in XP they are collected together. Practices are planning game, small releases, metaphor, simple design, testing, refactoring, pair programming, collective ownership, continuous integration, 40-hour week, on-site customer, coding standards, open workspace and just rules. From the viewpoint of our study, the most interesting practices are collective ownership and continuous integration. Collective ownership means that anyone can change anything at any time. Respectively, continuous integration recommends to integrate changes often with existing code. For an overview of other agile methods readers are re ferred to [e.g. 11].

2.2 Software Configuration Management

ISO standards [6, 12] introduce configuration management (CM) as a support process for a product development. It is a process of controlling the evolution of complex systems. CM process includes elements containing the basic CM activities and their CM planning [12, 13, 14, 15]. Traditionally CM activities have been divided into configuration identification, configuration control, configuration status accounting and configuration audit [6, 7].

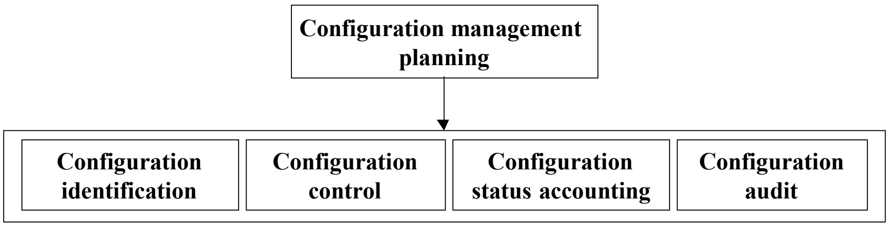

Configuration management planning
Configuration identification
Configuration control
Configuration status accounting
Configuration audit

Figure 1. Basic CM elements.

CM planning is used for planning and documenting certain configuration management solution for a project. Buckley [13] views CM plan as one of the major ways to communicate comprehensive understanding on what should be done to maintain the integrity of the products. CM planning provides means to define CM practices for a project: who is going to do what, when, where and how [13]. Standards such as IEEE Std-828 [16] and ISO 10007 [6] provide recommendations for CM plan contents. These recommendations can be utilized when creating CM plans for a company.

The role of CM tools is to support and automate CM functions and provide help for developers. However, Leon [17] states that CM tools do not solve configuration management problems, but they can be one step towards more effective CM. Even though tools can automate some functions, it is important that project team knows CM procedures defined for a project to ensure understanding why certain CM functions are needed as well as when and by whom they should be performed.

2.3 SCM in XP

Currently, there exist very few studies of software configuration management in XP. Paulk [5] has reviewed XP from the perspective of the Capability Maturity Model (CMM) and presents that SCM is partially addressed via collective ownership, small releases, and continuous integration. Christensen [3], who has researched change tracking in rapid and extreme development, presents that SCM has something to offer fast-paced development processes like XP. Our point of view is similar to Christensen as we think that SCM is needed in XP and it can support the XP development process.

XP literature emphasizes the importance of SCM automation to support XP practices [e.g. 10, 18 and 19]. Jeffries et al. [10] state that, in general, SCM tool should be easy to use. Further they emphasize that there should be as few restrictions as possible in SCM tool. For example, no passwords, no group restrictions, as little ownership hassle as possible. They mention SCM tools such as ENVY, CVS and Visual SourceSafe that can be used in XP projects. Succi and Marchesi [19] present that XP practices, such as collective ownership and continuous change integration, are not particularly well supported by traditional version and configuration management systems. Therefore, they have developed a new paradigm for supporting team software development, called *team streams*, that provides dynamic and easy to use team support. Succi and Marchesi [19] enumerate the team streams' characteristics that make them well suited for XP, such as easy to learn and to use, continuous integration, collective code ownership, fully optimistic concurrency, conflict detection and merging and tightly integrated team support.

Bendix and Hedin [18] report how CVS is used by students for simple configuration management on XP projects. According to their results, the students found CVS to be indispensable for the success of the group's effort. However, empirical data of comprehensive SCM exploitation in XP is rare and it indicates that more studies are required.

3 Research Approach

The purpose of this section is to clarify research methods used, and to introduce the basis for the research.

3.1 Research Methods

We used literature study as basis for our research and adopted principles from case study. Literature study was carried out to reveal the current state of software configuration management in extreme programming and to assist us in defining the configuration management principles and technical support for them. Research itself was carried out as a case study.

Järvinen [20] and Yin [21] presents the characteristics of a case study. Case study, as well as controlled experiment, uses research questions like how and why and it focuses on contemporary events. The difference between controlled experiment and case study is that experiment requires the control over behavioral events but the case study does not require such a control [21]. According to Järvinen [20], one specific aspect for controlled experiment is that researcher should be "a neutral observer" when the experiment is carried out in a laboratory environment. In our case, a researcher was in a role of customer who is an active participant in XP based product development and has a control to experiment through the required product features. Therefore, our research can be seen as a case study instead of controlled experiment. The framework, i.e. research manuscript, was created to guide us throughout the experiment. To decrease the number of data needed to collect during the case study the research focus was defined beforehand as the principals of case study emphasizes. The other factors we defined for the experiment were product features, product development environment and procedures, and templates for data gathering and SCM purposes.

3.2 Research Settings

A team of four developers was acquired to implement a system for managing the research data obtained over years at a finish research institute. The developers were 4-6th year students with 1-5 years of industrial experience in software development. Because team members had no earlier experience of XP nor software configuration management, they were given one day training on XP and other on SCM. Before actual training team members studied two books on XP to get the basics for the training. Both training days included theoretical and practical parts. Theoretical part of SCM training dealt with main SCM activities and their organization during the project. Practical part was focused on SCM tool usage in the XP environment. The project was conducted in 1-2 weeks iterations total of two months work effort. At the end of every iteration project team had produced full working software release, which was given to 17 allocated testers for the purpose of system testing. Table 1 shows the technical environment used in the development of system.

Table 1. Technical implementation environment.

Item	Description
Language	Java (JRE 1.4.1), JSP (2.0)
Database	MySQL (Core 4.0.9 NT, Java connector 2.0.14)
Development Environment	Eclipse (2.1)
SCM	CVS (1.11.2); integrated to Eclipse
Unit testing	JUnit (3.8.1); integrated to Eclipse
Documents	MS Office XP
Web Server	Apache Tomcat (4.1)

Planning of SCM implementation was conducted in the beginning of the first iteration using the generic SCM plan, which already included some general information and served as a template for project specific SCM plan. The generic SCM plan template was done according to IEEE standard 828-1998 [16]. Tailoring of generic plan template included adding of roles and responsibilities, schedules and project specific SCM practices.

The role of SCM tool in XP was found important in the literature study and, therefore, it was considered very carefully. The chosen tool was CVS mostly because the development environment (Eclipse) was shipped with a built in client for the CVS. In addition, CVS also supports XP's collective ownership and continuous integration -practices.

Traditional change management approaches can be seen too rigid for XP. Therefore, the purpose was to create simple change management approaches for the needs of this project including both release change management and customer change management processes. Basis for the release change management process was that team should be empowered to make changes through comprehensive unit testing according to collective ownership and continuous integration -practices. Table 2 shows the six stepped development process that was supported by CVS. The purpose of these steps was to ensure that developers would integrate only working versions of software to the repository.

Table 2. Release change management process.

1. Implement a task using the test first methodology
2. Compile and run all unit tests
3. Repeat the first two steps until the task is finished and all unit tests run at 100 %
4. Synchronize with repository
5. If conflicts or incoming files merge and go back to step 2
6. If no conflicts or incoming files commit changes to the repository

The basis for the customer change management process was that in XP the customer decides what to change and s/he can change the requirements at any time [8]. Thus, customer maintained a simple change request form (Excel form), in which change requests were listed. Figure 2 shows that the customer served as a filter between testers and project team. Testers send feedback to customer and he removed, for example, duplicates and impractical proposals. Every change request was equipped with additional information like classification, description and priority of change. New change request were approved or disapproved in the next iteration's planning game. As mentioned customer decides what to change, but project team helped by bringing their technical knowledge to decision-making. This simple customer change management process was mobilized after the first product baseline had been approved and the system had been sent to testers.

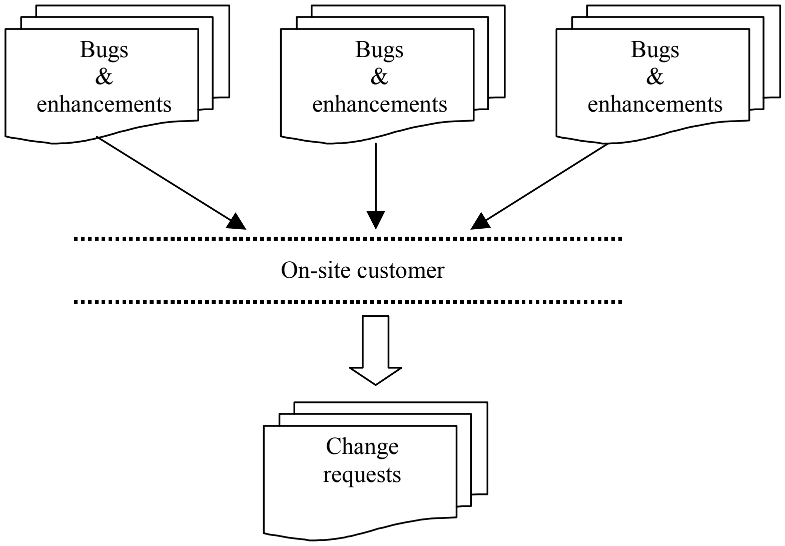


Figure 2. Customer change management process.

Three types of audits were performed during the project. Functional (FCA) and physical configuration audits (PCA) were conducted using a simple checklist method containing a number of requirements of both FCA and PCA. This checklist was examined at the end of every development cycle. FCA's requirement was, for example, that built software system corresponds with the user stories. The third type of audits, in-process audits, were conducted to ensure that SCM practices were followed as planned and everything was working correctly.

4 Results

The focus of this section is on reporting the SCM experiences and observations found during the project. This includes both quantitative and qualitative data of the project. First three development cycles were two-week releases and the following two were one-week releases. The release number six was two days long and included final bug fixes and enhancements of the system. Table 3 shows the data obtained from the project's releases. Data was drawn from the CVS tool.

The essential role of SCM tool emphasized during the project. Development artifacts were continuously safe in CVS server and developers integrated their changes through well-defined process. Team integrated code changes an average of two times per hour. Development team was interviewed after the project had been finished and asked that how often the code should be integrated. According to their answers, team tried to integrate at least once during the task, but even more often if possible.

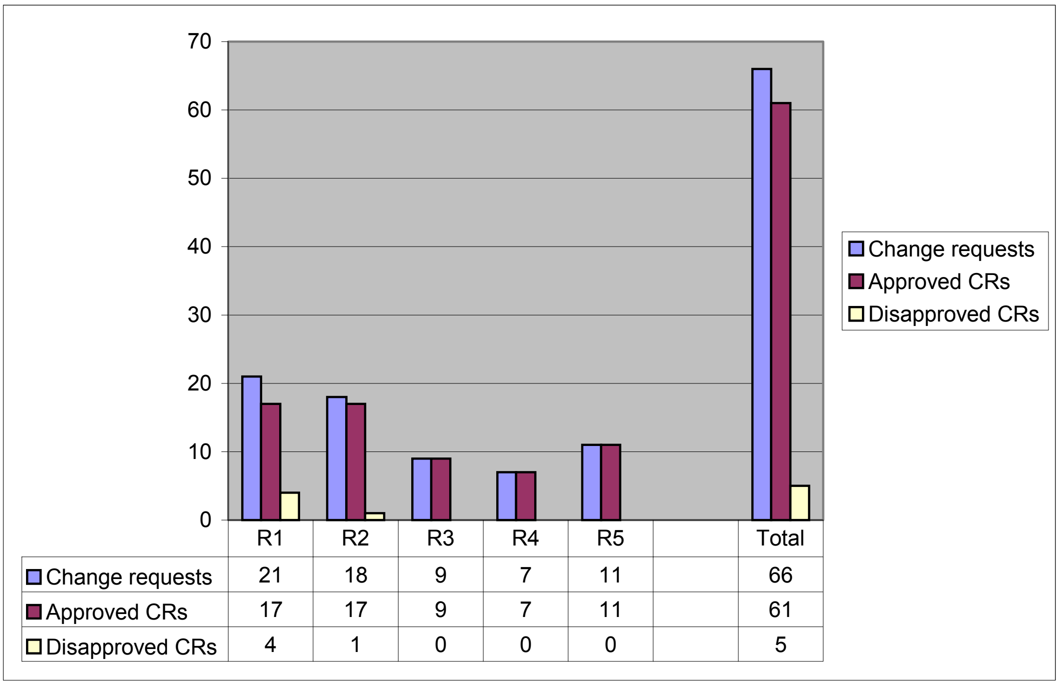
There was couple of times during the project, when intervals between integrations stretched too long and lots of changes directed to the same files. One reason for this was that developers wanted not to integrate unfinished items to the repository. Regardless of the reason, not following of continuous integration -practice caused complex merge-operations. Usually changes of one code integration cycle focused to a one file. However, the average number of files per code integration was 2.6. Due to CVS, team was able resume any earlier version of development artifacts they needed. There were situations when problems with item under development could not be solved without comparing the current item version with the previous ones. However, generally only the last versions of items in the repository were relevant.

Table 3. Concrete data from releases.

Collected data	Rel. 1	Rel. 2	Rel. 3	Rel. 4	Rel. 5	Rel. 6	Total
# Code integrations	65	81	71	42	41	17	317
Code integrations/workingday	8.1	10.1	7.9	10.5	8.2	8.5	8.9

Avg. time between two code integrations (min)	26	21	40	31	27	30	29
Avg. number of files per code integration	1.7	2.4	3.1	2.6	3.0	3.0	2.6
# New files (code)	22	15	16	1	9	0	63
# New files (other)	3	3	18	20	2	0	48
# New versions (code)	111	198	219	110	123	49	810
# New versions (other)	7	9	30	58	14	1	119

Both the release change management and customer change management processes proved to practical for this project. However, this was not so straightforward as both processes required commitment from the persons performing. The six stepped release change management process (see Table 2) ensured that everything in the CVS repository was always 100 % working. The basis was that release change management process steps were carefully defined and the SCM tool supported the process. In addition developers had to strictly follow the process steps and make unit tests comprehensive enough. Respectively the customer change management process required commitment from the on-site customer. Figure 3 shows data of releases' change requests. The on-site customer filtered testers' feedback to a list of change requests. As we can see from Figure 3, some of the change requests in first two releases had to disapprove, because they would have been too complicated to implement within the project's time schedules. A one example of disapproved change requests was version control feature. Approved change request were written down to tasks during the planning game and scheduled as a part of next release's content.



	R1	R2	R3	R4	R5		Total
Change requests	21	18	9	7	11		66
Approved CRs	17	17	9	7	11		61
Disapproved CRs	4	1	0	0	0		5

Figure 3. Data of releases' change requests.

Functional and physical configuration audits were conducted at the end of development cycles total of 5 times. If the results reviewed satisfied the audit requirements, system was ready to be released. Not once audit results were rejected. However, 2/5 times of FCAs and PCAs the results were acceptable with changes. These were the times when coding standards was followed insufficiently. SCM system audits were conducted at the beginning of every development cycle (except the first one). Therefore, the total count for in-process audits was four. In practice only two first of in-process audits produced changes to SCM practices documented in SCM plan. These changes concerned, for example, roles and responsibilities and change management practices.

In the final interview project team were asked that how important they see SCM in this kind of projects. Answers were very similar and emhasized the importance of SCM. SCM tool support was seen indis pensable. The most experienced team member answered as follows:

> *"It (SCM tool) is an essential part in the support of teamwork and should be axiomatic in every project."*

5 Discussion

The results presented in the previous section pointed some important requirements for SCM implementation in XP. Bendix and Hedin [18] have reported that students considered the merge support extremely helpful in XP projects. During this study there were only four persons (two pair programming pairs) developing the system. Results show that development team integrated their code changes on an average of 9 times per working day. XP literature [e.g. 8, 10] do not give exact integration intervals, but suggest to integrate often. Despite the fact that team integrated their changes often, merge operations occurred regularly, almost daily. Therefore, from SCM tool perspective it is important that there is straightforward and easy to use merge support, because concurrent changes are likely to happen. Bendix and Hedin [18] also report that very few of students in their XP projects had ever retrieved an older version of file. The results of this study support their findings, because generally only the last versions of files in the repository were relevant. However, there were some exceptions when previous versions were needed and, therefore, it is important that previous items and releases are identified and traceable. The great number of files and their versions speak for importance of version management. Overall SCM tool should be easy to use and not disruptive to encourage developers in tool exploitation. Leon [17] has argued that the role played by SCM tools is becoming more and more important in today's complex software development environments. In this study students found SCM tool very important and also the results show that SCM tool had an important role in this project. CVS proved to be good choice for project's SCM tool. Development artifacts were continuously in safe and developers had no fear to implement and integrate their changes.

SCM audits revealed, for example, that coding standards were not always completely followed. Therefore, audits had a positive influence to the internal quality of software releases. Results show that in practice only two first of in-process audits produced changes to SCM practices documented in SCM plan. This indicates that in-process audits were essential so that SCM practices matured to the level, where they were viable for this project.

Jeffries et al. [10] suggest to write problem reports to cards and schedule them in a current iteration or future iterations. Basically our customer change management process was based on their approach. However, our solution contained also change filtering, documentation and evaluation phases. Then, according to Jeffries at al.'s [10] suggestion, approved change requests were written down to task cards by development team and scheduled to the release. The results of this study show that this definite customer change management process works if the on-site customer have time to commit.

6 Conclusions

Agile software development methods have attracted great attention in the last few years. XP, currently the most well known agile method, is focused on delivering immediate business value to the customers. SCM is a method of bringing control to the software development process and is known as an inseparable part of quality-oriented product development regardless of development method. Thus, the value of SCM should not be underestimated in the case XP and other agile methods. Current studies show that SCM is partially addressed in XP via collective ownership, small releases, and continuous integration. However, traditional definition divides SCM into configuration identification, configuration control, configuration status accounting and configuration audits. Currently there exist very few empirical data of SCM exploitation in XP. This paper reports results from a controlled extreme programming case study supported by well-defined SCM activities and tools. Project team was trained to SCM before the start of the project. The SCM implementation was taken into account right from the beginning of the project, which means that project team tailored generic SCM plan template for the purpose of this project. During the project SCM implementation was audited regularly to ensure that SCM practices were followed as planned and everything was working correctly.

Results show that especially SCM tool has a remarkable role in XP project. Easy to use SCM tool with well-defined release change management process enables that project team can develop the system according to XP's collective ownership and continuous integration -practices. In other words the SCM tool can support the use of these practices. The results also show that SCM tool should have straightforward merge support and previous versions of files need to be identified and traceable. However, SCM tool was only a one part of project's SCM implementation. A simple customer change management process proved to be practical, but it was find out that on-site customer's commitment is required. In-process audits were found essential in order that project's SCM practices can be modified or even removed if needed. Because of regular in-process audits, SCM plan is up to date in every release. Functional and physical configuration audits had a positive influence to the internal quality of software releases. Overall the results show that SCM activities and tools provide essential support for XP development process and its practices.

7 Literature

[1] Maurer, F., Martel, S.: Extreme Programming: Rapid Development for Web-Based Applications. IEEE Internet computing. Vol. 6, Issue 1, 86-90 (2002)

[2] Charette, R.: The Decision Is in: Agile Versus Heavy Methodologies. Cutter Consortium e-Project Management Advisory Service, Vol. 2, no. 19 (2001)

[3] Christensen, H. B.: Tracking Change in Rapid and eXtreme Development: A Challenge to SCM-tools?. Tenth International Workshop on Software Configuration Management (2001)

[4] Compton, S. P., Conner, G. R.: Configuration Management for Software. New York: Thomson Publishing company (1994)

[5] Paulk, M. C.: Extreme Programming from a CMM Perspective. Paper of XP Universe (2001)

[6] Quality management – Guidelines for configuration management. International Standard. ISO/IEC 10007:1995. (1995)

[7] Bersoff, E., Henderson, V., Siegel, S.: Software Configuration Management - An Investment in Product Integrity, Prentice Hall (1980)

[8] Beck, K.: Extreme Programming Explained: Embrace Change. Reading, MA.: Addison Wesley Longman, Inc. (1999)

[9] Beck, K.: Embracing Change with Extreme Programming. Computer. Vol. 32, no 10, 70-77 (1999)

[10] Jeffries, R., Anderson, A., Hendrickson, C.: Extreme Programming Installed. NJ: Addison-Wesley (2001)

[11] Abrahamsson, P., Salo, O., Ronkainen, J., Warsta, J.: Agile Software Development Methods: Review and Analysis. Technical Research Centre of Finland, VTT Publications 478 (2002)

[12] Information technology - Software life cycle processes. International Standard. ISO/IEC 12207:1995. (1995)

[13] Buckley, F.: Implementing configuration management : hardware, software, and firmware. IEEE Computer Society Press, Los Alamitos (1996)

[14] Taramaa, J.: Practical devalopment of software configuration management for embedded system. Technical Research Centre of Finland. VTT Publications 366. Espoo (1998)

[15] Rahikkala, T.: Towards virtual software configuration management: a case study. Technical Research Centre of Finland. VTT Publications 409, Espoo, (2000)

[16] IEEE Standard For Software Configuration Management Plans. IEEE Std-828-1998. (1998)

[17] Leon, A.: A Guide to software configuration management. Artech House, Boston (2000)

[18] Bendix, L., Hedin, G.: Summary of the subworkshop on extreme programming. Nordic Journal of Computing 9 (2002)

[19] Succi, G., Marchesi, M.: Extreme Programming Examined. Boston: Addison-Wesley (2001)

[20] Järvinen, P.: On Research Methods. Tampereen Yliopistopaino Oy, Juvenes-Print, Tampere (2001)

[21] Yin, R., K.: Case study research: Design and methods, Sage Publications, Beverly Hills Ca (1989)

8 Author CVs

Juha Koskela

Juha Koskela has now worked since June 2002 as a research trainee and since July 2003 as a research scientist in Embedded product data management -research group at the Technical Research Centre of Finland. He has received Master of Science Degree in June 2003 in information processing science from the University of Oulu. His research interests include Software Configuration Management and agile software development methods.

Jukka Kääriäinen

Jukka Kääriäinen has now worked since 1999 as a research scientist in Embedded product data management -research group at the Technical Research Centre of Finland. He has received Master of Science Degree in 1999 in Computer Science and Bachelor of Science Degree in 1994 in Industrial Engineering and Management. His research interests include Software Configuration Management and Product Data Management (PDM).

Juha Takalo

Juha Takalo has received Master of Science Degree in computer science in 1995 from the University of Oulu. He works as a research scientist in Embedded product data management -research group at the Technical Research Centre of Finland. His current research interests cover product data management (PDM), configuration management (CM), and document management.

SPI: Comparison of Elicitation Methods to Discover Process Defects

Sigurjón Hákonarson, Ebba Þóra Hvannberg
Department of Engineering, Computer Science, University of Iceland

Abstract

The main objective of our research is to see if it is possible to develop procedures that use current models and standards as roadmaps when implementing a Software Process Improvement (SPI) Program. Our approach is developer centred, where the developer has the key role of providing information on process defects, defect causes and suggestive improvements.

Before we can institutionalise a process we need to examine the environment we wish to institutionalise that process in. The most important part of that environment is the people that work in it. Close connection and cooperation with developers in implementing a SPI program is necessary for its success.

This paper discusses the results of comparing three elicitation methods for discovering software process defects. The methods compared are Attitude Survey (AS), Post Mortem Analysis (PMA) and Initial Assessment (IA) when process defects were classified according to organizational, engineering, project management and support processes, the results showed that PMA gave similar findings as IA. The IA required more preparation than PMA but PMA has a more central focus on the developers than the IA. For the organizational processes, the attitude survey had the most defects but the initial assessment didn’t produce any defects. For all other categories the attitude survey did discover the fewest process defects.

Keywords

Software Process Defects, Software Process Improvement, Post Mortem Analysis, Software Process Assessment, Developer Centred Approach, Empirical Study, Assessment methods

1 Introduction

In the area of Software Process Improvement (SPI) and Software Process Definitions there are several models and standards one can comply with. Still we are looking for more effective ways to implement SPI in software organizations and thus ensure continuous improvements and learning. It is a challenging task to keep process definitions alive and active in an organization. Some might claim that the reason for this is that SPI is more a tool for senior management to enforce order and structure in an organization, rather than a tool for the developers to ensure stability and repeatable project success. It is our belief that SPI implementation could be very efficient if we use approaches where developers' participation is active throughout the whole development cycle of the SPI implementation.

The main objective of our research is to see if it is possible to develop procedures that use current models and standards as roadmaps to implement a SPI program, with developers as key role players. Our approach is developer centred, where the developer is the primary source of information on process defects, defect causes and suggestive improvements. It is also the purpose of this research to see how effective and efficient it is to use developer centred methods to implement SPI programs.

The results presented here are a part of a larger research project that involves the development of an effective SPI program with active participation of developers. The implementation of the SPI program will be partially based on the IDEAL life cycle model for SPI [3,4]. Figure 1 shows how the research methodologies and activities are related to the model and in what phases they will be implemented. The IDEAL development cycle describes the phases, but we have decided what methodologies to use and more specifically selected the activities for each methodology. The objective is to see if the IDEAL model is appropriate framework for developing a SPI program for small and medium-sized software organizations by using the selected methodologies and applied activities in this research.

The analysis phase (Initiating and Diagnosing) of the SPI lifecycle will be focused on analysing the environment, the transformation that will take place as a result of the improvement effort and finding the actors and stakeholders in that environment according the Soft System Methodology [5]. In the design phase of the SPI lifecycle, an experimental approach will be developed that we have named Developer Centred Approach (DeCA).

The research will be conducted with contextual inquiry, where methods and activities will be customized and reformed according to developers needs.

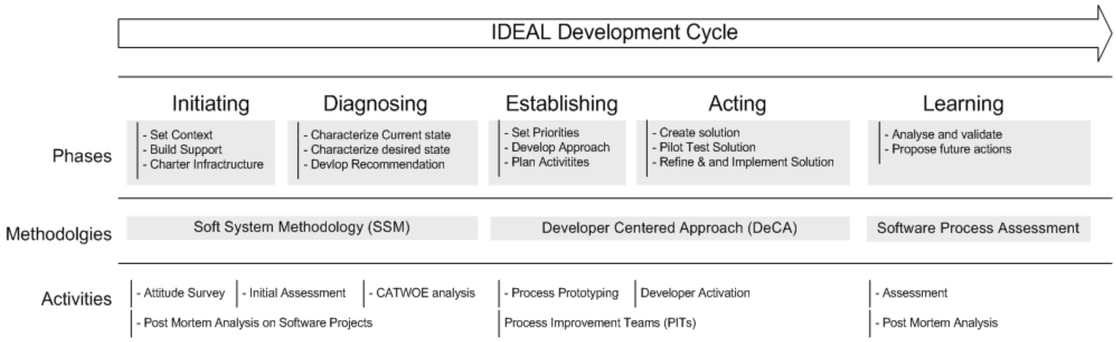


Figure 1 – Research Project Framework

The results presented in this paper are the elicitation and analysis part of SPI development cycle, and the purpose is to get a comparison of three elicitation methods we used to discover software process defects in the organization. Each elicitation method focuses on discovering different types of defects from developers and with different degree of formality.

2 Background and Related Work

2.1 Process Models and Standards

There are various models and standards that have been created to establish an appropriate framework for the ideal software process and life cycle processes. The International standard *ISO 12207 Information technology – Software Life Cycle Processes* establishes a common framework for software life cycle processes. It contains process activities and tasks that are to be applied during acquisition, development operation and maintenance of software [1].

ISO TR 15504 Information technology – Software Process Assessment has been published as a Technical Report and is the SPA standard that was initiated by the SPICE project [9]. It was an effort to create an international standard for Software Process Assessment (SPA). ISO 9001 has also been used by many software organizations, especially in Europe. Though ISO 9001 is not a standard specifically intended for software organizations it is acceptable for all organization that are developing products or delivering service of some kind [1]. The standard ISO 9000-3 is a guideline for interpreting the ISO 9001 standard for the development, supply and maintenance of software. The frameworks quagmire (http://www.software.org/quagmire/) [6] is a diagram showing the relationships between process standards, quality standards, capability models, appraisal methods and guidelines.

Many organizations have tried to establish a quality management system based on these models and standards and too many of them have failed or abandoned their effort early in the process. SEI has been in a leading role in establishing models and guidelines for SPI work, and many now consider the Capability Maturity Model Integration (CMMI formerly CMM) [2][6] the de facto standard for SPI models [1].

2.2 The Role of People in SPI

Research [8] has shown that software development in Icelandic organizations is still a long way from following standard procedures in software development. The standards, guidelines and models currently available place too much emphasis on the process instead on focusing first and foremost on the people operating the process. Before a process can be institutionalised we need to examine the environment we wish to institutionalise that process in. The most important part of that environment is the people that work in it. If we want to change the way developers work we need to work closely with them in designing and implementing the appropriate processes. We need methods that examine the environment and the work's context more closely and the practices that developers use on a daily basis, for the purpose of discovering good habits and procedures on which we can build the improvement effort. It is also necessary to realize that employees need a very good reason to change their way of working [7], and do not always agree with us that the improvement is necessary Therefore, close connection and cooperation with developers in implementing a SPI program is necessary for the success of SPI.

3 Environment of Software Process Improvement

This section describes how we selected the environment of the research study, i.e. the organization, the projects and the participants. The section serves at the same time as a description of the context, stakeholders and actors of the study as well as the support that was built before the study started and the infrastructure. This corresponds to the Initiating phase of the IDEAL Development Cycle and Clients, Actors, Owner, Worldview and Environmental constraints are parts of the **CATWOE** analysis in the Soft System Methodology [5]. The worldview defines the context in which makes the work mean ingful, which is in our case the project areas. The T stands for Transformation and is in our case the findings and improvement suggestions that will be done on the software development activities.

3.1 Research Participation

To adhere to the objective of the research we decided that the main focus groups should be developers mostly, project managers and group leaders. A visible support by senior management was necessary and was received in the beginning. The quality manager for the organization was an active supervisor of part of the organization and monitored all work. A weekly status was held with the quality manager to go over the project plan and progress.

In the currently examined organization there is a clear organizational hierarchy of authority where system developers that are at the lowest level are not so much aware of defined processes. Instead the project managers, which are in most cases technical leads, have the responsibility of performing the process. Project managers usually manage one or more project at a time, depending on the frequency of project work in each project. Group leaders have the responsibility of managing and overlooking many interrelated projects. They are the primary contact to the customer, and take care of issues regarding contract work and bids. They are also the communication point between other members of the development group. The group leader has the responsibility of communicating changes to the process model to other members of the group as well as enforcing the process performance.

Senior management are actively involved in the SPI project, they receive status reports on a regular basis, both verbally and in writing. At the beginning of the project it was obvious that senior management was ready to provide time and money to this effort and they understand the importance of continuous SPI effort in the organization.

3.2 Selected Partner and Project Area

3.2.1 Partner

When we selected a partner for the research study we found it crucial that the organization did either have an ISO 9001 certification or was currently planning to receive one in the next year. It was also important that senior management did have an idea of the meaning of CMMI. We believed that a piloted effort with a more mature organization would give us more realistic results in relation to SPI. Since there are very few software organizations in Iceland that have been ISO certified, the ISO certified organization that seemed to have the broadest development spectrum was selected.

The reason why only one partner was selected for the purpose of this research is mainly because of time constraints and the nature of the research method is not a comparative study but action research. It is also believed that a piloted effort with a more mature organization should give us more realistic results in relation to SPI.

The organization consists of about 50 software developers, with total of 80 employees working in the software department. Each development group consist of 10-20 people, and each group is organized around interrelated software projects, project types ranging from development custom applications, maintainance of legacy systems to the installation and adoption of business software.

3.2.2 Project Area

Senior management decided what projects should be investigated. The main motivation for the selection was that they believed that something had gone wrong in a particular project that should be explored further. Also the organization had not developed software in the area of web applications before. Senior management believed that the process definitions (ISO quality management system) did not support web application projects. We also suggested that it was important to investigate other projects in the same project area, at different status in the life cycle, and projects in different project area at same status. This could give us a better overall perspective on SPI status in the organization and better results for comparison.

4 Software Process Assessment Results

4.1 Elicitation Methods

The second phase of the IDEAL Development Cycle is about diagnosing the current state, characterising where we want to be based on the current state. Essentially, this means that we find areas in our assessment results that we like to improve in our organization. For the analysis phase of the project we decided to use three elicitation methods to find information on different aspects of SPI effort in the organization. Each method has its characteristics but they all have the same objective, which is to find out the capability and maturity status of the engineering, project management and supporting processes. Therefore resulting in process defects on various areas in the organization. In that context we were especially interested in following issues:

- The SPI importance in the organization.
- Employees' attitudes towards SPI and quality issues.
- The capability and maturity of key processes.

By using three different elicitation methods it was possible to get a comparison between these methods and how well they discover the information needed for further improvement. Table 1 summarizes the differences in formality and structure of these methods.

Method	Number of participants	Formality	Session length	Developer preparation
Attitude Survey Prepared questionnaire. Semi-structured interview. Interviewees were asked questions on issues regarding the quality management system and SPI in general. Discussions were not limited to the questions at hand.	2-3 in each group	Very Informal	1 hour per group	None
Post Mortem Analysis Structured Post Mortem session on one Software Project at a time, with 2-3 three developers at a time. Developers wrote down issues regarding their project experience, positive and negative. Developers discussed these issues and categorized them into groups. The primary focus was on developers.	2-3 in each group	Semi formal.	1 ½ hour per group	None
Initial Assessment Structured assessment session based on the SCAMPI [10] assessment approach and maturity questionnaire [11]. Followed by a review session that consisted of project managers and group leaders. The CMMI maturity level for the investigated organization has not been determined, but according to the information gathered at this point, it can be estimated that some process areas are largely fulfilled at level 2, but most process areas do only fulfil partially at level 1 and 2.	2-3 in each group	Very formal.	1 ½ hour per group	Developers had to read instructions on IA guidelines and SPA procedures.

Table 1 –Characteristics of Elicitation Method Structures

4.2 Software Process Defects

The results presented here are the various types of process defects found during our assessment. It is the summation of results and is the basis for further Software Process Assessment. The methods used to discover the defects have been discussed earlier in chapter 4.

The following tables list the process defects found in various process areas. It is identified with what elicitation method (AS-Attitude Survey, PMA-Post Mortem Analysis, IA-Initial Assessment) each process defect was discovered in.

Ref. No.	Process Defect	Disc. in
PM-001	**Project Classification** is to broad, there is too much difference in process responsibilities between A and C project classification.	AS
PM-002	A **quality plan** is used in every project but developers are not aware of its purpose.	PMA
PM-003	In some projects (Oracle project) the **division of labor** was unclear, developers and others were not clear what was expected of them.	PMA
PM-004	**Unrealistic Project Plans** in some projects (Oracle). Marketing People did the project plans and they seemed unrealistic to developers in many ways.	PMA
PM-005	No **formal risk analysis and assessment** is conducted on projects, no process definition exist on this matter.	IA
PM-006	**Performance measures** are informal and feedback to developers is not visible and given on regular basis, and in some cases the results are not obvious, better measurement feedback is needed.	IA
PM-007	**Understanding of senior management** was unrealistic. Regarding the scope of a web project, was considered a small web project. Better project scope analysis is needed. This resulted in use of inappropriate resources, which lead to various technical difficulties.	PMA
PM-008	Developers said that they often needed a **better perspective of the bigger picture**, to understand how their work related to other parts of a larger project, or possibly other independent projects for same customer.	PMA

Table 2 – Process Defects Related to Project Management

Ref. No.	Process Defect	Disc. in
ENG-001	No formally defined **process for managing requirements** during development only during maintainance. Requirements changes during development are only documented informally in meeting minutes.	PMA
ENG-002	**Coding standards are outdated**. No coding standards exist for JSP programming though JSP coding is extensive in the organization. It is suspected that the same situation exists for other programming language.	PMA
ENG-003	An **understanding in Unit testing** is missing, there is no clear distinction made between unit testing and debugging among developers. No process definition exist on Unit Testing.	AS
ENG-004	A **process definition for Installation and adoption projects** (Oracle) was missing.	PMA
ENG-005	Lack of experience in Development Environment, web technology and distributed environments. **No training and support was provided** during development.	PMA
ENG-006	**No technical support** individual dedicated to a project. Roles and responsibilities surrounding technical support need to be reformed and constructed.	PMA
ENG-007	Oracle Solution Provider does not have enough experience with development tool.	PMA
ENG-008	**Changing Requirements**. The change control board did not represent all the users and customers appropriately.	PMA
ENG-009	Traceability between tests and requirements is not maintained properly. No formal structure for creating test cases and no one has the role of a tester.	AS

Table 3 – Process Defects Related to Engineering Activities

Ref. No.	Process Defect	Disc. in
NT-001	Problems with **adopting new technology**. Procedures for adopting new technology need to be established.	PMA
CM-001	**Configuration Management activities** are very informal and very different from project to project. Version Control is not properly done in various projects especially Oracle related projects. For every project it is necessary to define what products go under CM. Consistency among documents under CM is lacking.	IA/PMA
SAM-001	**Hardware and systems investments** have to be considered more carefully, possibly setting up guidelines for selecting the appropriate hardware/system software for use in projects.	PMA

Table 4 – Process Defects Related to Supporting Activities

Ref. No.	Process Defect	Disc. in
QMS-001	Improvement projects do not have a clear focus and too often they are abandoned.	AS
QMS-002	Quality Management System (QMS) is distant and information in it is hard to obtain. There is no central point of beginning if you need to obtain information about the processes you are going to use.	AS
ORG-001	Limited opportunity for training and seeking knowledge during development. An educational policy has not been established, or is not visible to all. Some find that too often the wrong people are sent on seminars.	AS
QMS-003	Internal Audits are too limited to project managers, instead of programmers (developers). Also many employees find internal audits intimitating.	AS

Table 5 – Process Defects Related to Organizational Issues

Further analysis on process defects and elicitation methods is discussed in the following chapter.

5 Comparing Elicitation Methods

We decided to categorize the information discovered in all the elicitation methods to see what kind of information we were discovering with each method. First we used the four main areas in which CMMI organizes the key process areas, i.e. Organizational, Project Management, Engineering and Supporting Processes.

At the beginning we had the same objective for every method, that was to obtain information on process capability and maturity and how well that method was for doing so. After implementing all three methods we noticed a difference in the types of information we discovered. The following section explains this difference in details.

When comparing the elicitation methods we wanted to see by applying the methods if different information was discovered by different methods. We specifically searched for answers to the following questions:

- Is there a difference in the detail of information we find by applying each method?
- Do we find different types of process defects with different elicitation methods?
- How important is the structure of the method for finding reliable information on process performance?

Table 6 summarizes the findings from each method. It describes the types of process defects and the general results discovered in each method.

Method	Types of Process Defects	Results and experience
Attitude Survey	Very general information was discovered Problems with organizational structure and the Quality Management System as a whole rather then specific process areas Issues that people were frustrated about came to the surface	This method did not work to well. The information gathered here were in most cases too general and did not focus on specific process areas On the other there were some interesting remarks made on the current improvement structure which is important to look at
Post Mortem Analysis	Delivered detailed information on problems with specific process areas that particular project experienced Project specific problems Detailed information on problems with interrelated process areas Communication problems	This method worked extremely well and defects were found that could be localized to specific process areas Though the problems reported were regarding specific project, they clearly identified flaws in the general process definition (if it existed) Many ideas on better and improved ways came to

		the surface, a lot of good improvements points
Initial Assessment	Delivered detailed information on problems with specific process areas, some questions were not applicable to the people interviewed since they did not know too much about process definitions Problems with one specific process area across different projects	Did show some problems with the process definitions though developers did sometimes not realize there was a problem with it This method assumes that the person interviewed knows the process definition well and how it is implemented in details. This made a problem when interviewing developers

Table 6 – Summary of Elicitation Method Results

To find what kind of process defects were discovered with each method, we categorized the types of process defects found in the elicitation according to the following structure:

Organizational Issues: The Quality Management System (incl. Process def.), Working Environment, And Senior Management

Engineering Processes: Requirement Management and Development, Design and Construction, Verification and Validation

Project Management: Project Planning, Monitoring and Control, Contractor Management

Support: Configuration Management, Measurement Analysis

The results from each method were rated. If a person participating in the elicitation mentioned a problem with managing requirements, that specific process area received a point. In each method there were 10-12 participants. Figure 2 shows the coverage for each method applied, figure 3 summarizes the ratings for each process area.

Methods / Process Areas	Organizational	Engineering	Project Man.	Support
Attitude Survey	5	1	0	1
Post Mortem Analysis	1	5	5	4
Initial Assessment	0	5	5	5

Table 7 – Distribution of Process Area Coverage Rating

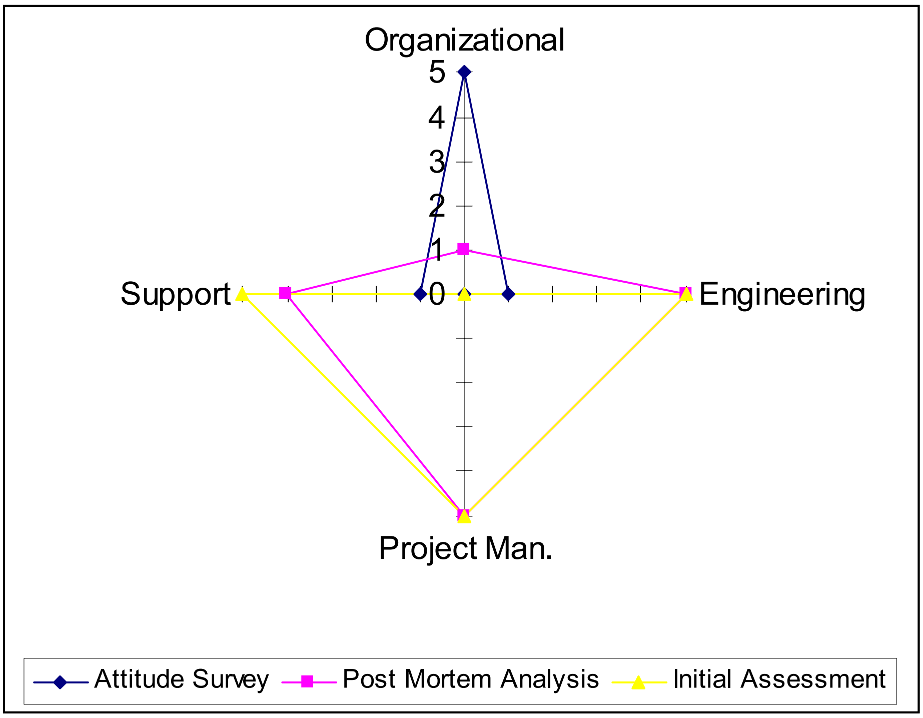


Figure 2 – Process Area Coverage by Elicitation Methods

We considered a process area fully covered (receiving the grade of 5) in a method if all of the issues in that grouped where mentioned and all of the participating groups did mention it.

6 Conclusion

In this paper we have presented the results of comparing three different elicitation methods and how they discover different types of process defects. The methods are different in formality and the emphasis they have on developers' participation. The results we found were categorized in four groups, organizational, engineering, project management and support. Those groups were then used to see what areas the methods were covering, which was represented in figure 2 and 3.

By comparing the elicitation methods we got interesting results on what type of process defects we found with each method. It is obvious that the attitude survey discovers different kind of information than the other two methods. Though the results from PMA are very similar to IA results it is still interesting to see that a semi-formal method like PMA discovers just as detailed information as IA. The IA also requires more preparation and understanding of involved participants. In cases where developers lack an understanding and knowledge of process definitions and SPI, many questions were not applicable and should be directed at the quality manager rather than developers. The PMA method did also have a more central focus on the developers and the project they had been or were currently working in. IA was focused on the process definitions primarily.

The findings and results presented in this paper are a part of a larger research project. We are currently located in the Establishing phases of the IDEAL model (see figure 1), where the results presented here are from the Initiating and Diagnosing phases.

7 References

1. Zahran, S. (1998), “Software Process Improvement– Practical Guidelines for Business Success”, Addison-Wesley, 1998.
2. Software Engineering Institute_CMMI Product Team, (2001), “Capability Maturity Model Integration, version 1.1”, Software Engineering Institute, 2001.
3. McFeely B., (1996), “IDEAL^{SM}: A User´s Guide for Software Process Improvement”, Software Engineering Institute, 1996.
4. Gremba J., Myers C., (1997), “The IDEAL^{SM} Model: A Practical Guide for Improvement”, Software Engineering Institute (http://www.sei.cmu.edu/ideal/ideal.bridge.html), 1997.
5. Checkland P., Scholes J., (1999), “Soft System Methodology in Action”, John Wiley and Sons, May 1999.
6. Ferguson J., Sheard S., Leveraging Your CMM Efforts for IEEE/EIA 12207, IEEE Software Sept/Oct. 1998.
7. Kotter, J. (1996), “Leading Change”, Harvard Business Press, 1996.
8. Hvannberg, E.T. Thorbergsson, H., Jóhannsdóttir, G. A, “A Comparison of Techniques to Support Small Companies with Software Process Improvement”. In Proceedings of EuroSPI 2000 Copenhagen, November 2000.
9. SPICE website (http://www.sqi.gu.edu.au/spice/).
10. SEI, CMMI Product Development Team, Standard CMMI Assessment Method for Process Improvement (SCAMPI): Method Description, Software Engineering Institute, 2000.
11. Zubrow D., Hayes W., Siegel J., Goldensen D., SEI, CMMI Product Development Team, Maturity Questionnaire, Software Engineering Institute, 1994.

Author CVs

Sigurjón Hákonarson

Sigurjón Hákonarson received his B. Sc. from Coastal Carolina University, South Carolina. He is currently an M. Sc. at the University of Iceland, Reykjavik. He worked in software development during the years of 1997-2000, and then became a project leader for several software projects. In the year 2001 he became the quality manager for two subsidiaries of TM Software in Iceland.

He is currently working as an adjunct in Reykjavik University, teaching Software Engineering and supervising students final projects. His main interest areas are in Software Process Improvement, Software Quality Assurance and Software Process Assessment. His website is www.ru.is/sigurjon.

Ebba Þóra Hvannberg

Dr. Ebba Thora Hvannberg is an associate professor in the Computer Science Department of the Engineering Faculty of University of Iceland. Her main research and teaching interests are in the area of software science, distributed systems, multimedia, formal methods and human computer interaction. She has a B.S. from the University of Iceland, and M.S. and a Ph.D from Rensselaer Polytechnic Institute. Ebba Thora has participated in several European, North American, Nordic and Icelandic research projects. She has been an active member of the research community, by publishing papers, giving talks at conferences and as a reviewer of papers and proposals. Ebba Thora is a member of ISTAG, Information Society Technologies Advisory Group of EU's 6th framework programme. She serves on the board of RHnet, Icelandic University and Research Network. Ebba Thora is a member of ACM and IEEE. Her web site is www.hi.is/~ebba.

CODE for SQM: A Model for Cultural and Organisational Diversity Evaluation

Kerstin V. Siakas[1], Eleni Berki[2], Elli Georgiadou[3]

[1]Technological Educational Institution of Thessaloniki,
Department of Informatics, P.O. Box 14561
GR-54101 Thessaloniki, Greece
Tel: +30 2310 791 296, Fax: +30 2310 791 290
E-mail: siaka@it.teithe.gr

[2]University of Jyväskylä, Dept. of Computer Science and Information Systems
Mattilanniemi Campus, AGORA Building, Jyväskylä, FIN – 40014, Finland
Tel.: + 358 (0) 14 260 3036, Fax: + 358 (0) 14 260 3011,
E-mail: eleni.berki@cc.jyu.fi

[3]University of Middlesex, School of Computing Science
Trent Park Campus, Bramley Road, London, N14 4YZ, UK
Tel: + 44 208 411 4331, Fax: + 44 208 411 5924,
E-mail: e.georgiadou@mdx.ac.uk

Abstract

This paper refers to the critical cultural and organisational factors that need to be considered when introducing a new or changing an existing Software Quality Management (SQM) system. This work is based on our findings from research carried out on software quality assurance issues and Software Process Improvement (SPI). The research was focused on the significance of the organisational context, including influences from the national culture in which the organisation is situated. The paper presents the *SQM-CODE model*, which was developed as part of this research work. We initially refer to the reasons of its construction and subsequently on the scientific rules of its functionality. The model investigates to what degree there is a fit between the organisational and the national culture and consequently suggests a suitable SQM system to fit the organisational needs. SQM-CODE includes two sub-models, namely the *C.HI.D.DI typology*, which initially identifies the type of the national culture and the *Authoritarian-Participative model*, which subsequently identifies the organisational culture and structure. Finally, by commenting on the model's verification and reflecting on other observations by its application, we argue that the model can be applicable in many organisations worldwide.

Keywords

National Culture, Organisational Culture, Software Quality Management, Software Process Improvement, SQM-CODE, Authoritarian-Participative model, C.HI.D.DI. typology.

1 Introduction, Motivation and Perspectives

The ‘software crisis’ has been characterised by an inability to develop software on time, on budget, and within requirements [Boehm, 1993]. In 1999 Mullet stated that large software systems (with more than 50,000 lines of high-level language code), bring the software crisis to light. She argues that 75% of all large software products delivered to the customer are either not used at all, or do not meet the customer’s requirements. One of every three large projects are cancelled before even being completed and only one out of eight finished software projects are considered successful. In year 2000 it was stated by the IPSSI (Improving Professional Software Skills in Industry) Consortium that only 13% of IT projects are successful. In addition software projects are notoriously behind schedule and over budget. Research carried by the Standish Group in 1998 provides information on the cost of system errors. The evidence suggests that on average software systems over run by 89% on cost and by 122 % on time.

This statistical evidence is indeed very alarming, and shows that despite of more than twenty years of attempts to practise software development in a more predictable and controllable way, the software crisis still exists. The software industry is still many years away from becoming a mature engineering discipline and there is certainly no single solution to the crisis. Awareness of the software crisis has inspired a worldwide movement towards process improvement and many different paradigms have been created that can benefit today's software projects. The Information Systems industry has mainly turned its attention to the software development process, arguing, like other industries before it, that a high-quality process will deliver high-quality products. Thus various Software Quality Management systems have been proposed and developed with emphasis on SPI. Although all these involve technological elements (which have generally received much attention), most people now recognise that the successful development of SPI depends more heavily on social factors. After examining the best available methodologies, techniques and tools and implementing a Software Quality Management system with all its rules and regulations, if we still do not have a successful and competitive organisation then it is probably due to lack of attention to social issues. Yet these significant social factors have been comparatively neglected.

The difficulties of achieving social acceptance for Software Quality Management Systems will also be exacerbated in the future by the globalisation of the software market and the increased use of cross-cultural development teams within multinational companies. The globalisation of the software market has generated a need for the contextual boundaries of IS research and practices to include the societal context. Management that can take account of the cultural context of their endeavours deserves better success.

In general, research in IS has not considered culture when investigating the process of software development. In particular, quality related issues seem to be missing from the literature. Research on how culture influences software quality issues is still novel. Some recent attempts have been made to examine software quality in an organisational context [Biro et al., 2000] without taking national culture into consideration. Some attempts have been made to document differences amongst IS professionals from different nations [Couger et al., 1988, 1989, 1991; Holmes, 1995, Ives and Järvenpää, 1991; Keen, 1992; Kumar and Bjorn-Andersen, 1990; Palvia et al., 1992], but most of these studies have compared IS or IS related issues in only two countries mainly by using an ethnocentric approach (the same research is replicated in a second culture).

2 Research Questions and Research Methods

We argue that an integration of a disciplined approach, focusing on repeatable processes and continuous improvements, together with emphasis on the importance of people and culture within the organisational context will be of added value to the software industry.

The main objective of this research study has been to add to the knowledge of the factors that influ ence the successful implementation of a Software Quality Management system. The research in particular examined those factors, which form a cultural and organisational perspective. One of the research outcomes was the global theoretical research model incorporating key cultural variables.

The research question that the study has addressed was to what extent cultural factors influence the successful adoption and implementation of a Software Quality Management System. The analysis has considered factors from both national and organisational areas. The existence of quality oriented management procedures, (similar to the procedures identified in Capability Models), was investigated empirically, together with the awareness of quality issues amongst the workforce. The research has shown that culture is a source of competitive advantage/disadvantage and organisational issues, such as organisational structure together with national culture play a crucial role in choosing and implementing a Software Quality Management System.

The research method used was a contemporary comparative *multimethod* also called *triangulation* using *both quantitative and qualitative investigation*. The quantitative investigation was a survey collecting 'hard' data by using a postal questionnaire. In parallel, a qualitative method in form of case studies was performed in order to address different aspects of the research problem, to confirm the findings from the questionnaire and to prove the hypothesis.

The questionnaire was sent to organisations developing software in Denmark, Finland, Greece and the UK. Totally 307 questionnaires were completed. In addition field-studies were undertaken in several organisations. Totally 87 interviews were conducted in Finland, Denmark and Greece with software developers at different levels and with different positions in the organisation. In Finland three organisations took part in the interviews. Two of them were multinational and one was a big local organisation. In Greece two evolving multinational organisations took part in the interviews and in Denmark one global organisation with subsidiaries in Germany and the US. Following the initial verification phase the author also carried out observations in the Danish organisation for a period of two months [Siakas and Balstrup, 2000].

The aims of the observations were to investigate in more depth the research problem in order to comprehend both the cultural and the software quality issues related to this research, and also to verify the findings and validate the SQM-CODE model that was developed as part of this research. Consequently and by its very nature the investigation utilised the strengths of cross-national comparative studies.

2.1 Focus on Cross-national Comparative Studies

Thinking in comparative terms is inherent in sociology. Recent globalising trends have generated more cross-national studies. All the eternal and unsolved problems inherent in sociological research are unfolded when engaging in cross-national studies [Hoecklin, 1995]. Oyen [1995] found in her literature review that although some researchers disagree about goals and the theoretical framework for cross-national studies, they agree on basic rules of scientific analysis. These rules are concerned with constructing concepts and typologies, and securing ties between data and theory, as well as making use of judgement. According to Oyen, country comparisons underpin much of the empirical foundation of macro social, economic and political theory.

These country comparisons dominate how we think about political and social systems - groups and organisations - and our efforts to discipline ideas. In this sense, Oyen considers that we talk about comparative methods, referring to research about any impact of macro or more encompassing systems on micro ones, and vice versa. Social science disciplines compare countries; anthropologists compare culture and institutional change. Oyen also believes that one scientific objective is to reduce variance or variety by putting countries into general categories or groups or typologies on dimensions asserted to be theoretically significant, as for example wealth, democracy, size, culture, socialism etc. It has also been pointed out that different years and time intervals can have different meanings for different countries and activities.

3 The Inescapable Diversity of Culture

Researchers and practitioners have identified and discussed the importance of culture for organisations, but there is no common understanding or agreement on the definition of culture [Groeschl and Doherty, 2000]. The lack of clarity on definitions and meanings of different terms commonly used in a cultural context likely could be attributed to the fact that, regarding cultural studies many different academic disciplines are involved, where the same terms have different meanings; different terms are also used for the same concept.

In 1952 Kroeger and Kluckhohn identified 164 different definitions of culture. In 1983 Hofstede argued that *"there is no commonly accepted language to describe a complex thing, such as a culture"*; and in 1994 Tayeb stated that *"culture is too fundamental to be solved through tighter definition"*. However, there are similarities in the different approaches to identifying and defining culture. Researchers such as Kluckhohn [1951], Kluckhohn and Strodtbeck [1961], as well as Inkeles and Levison [1969] relate differences between cultures to *different approaches to solving common human problems*. Many more recent researchers [Hofstede 1984; 2001; Schein, 1985] seem to have adopted this approach.

3.1 The Key Elements of National Culture

Hofstede distinguished four key elements, or "dimensions", of culture [Hofstede 1994; 2001] as described below:

- *power distance*, which describes the extent to which hierarchies and unequal distribution of power is accepted;
- *uncertainty avoidance*, which indicates the extent to which a society feels threatened by ambiguous situations and tries to avoid them by providing rules, believing in absolute truths, and refusing to tolerate deviance;
- *masculinity versus femininity*, which describes the relationship between the masculine assertiveness, competitiveness and materialism opposed to the feminine concern for quality of relationships, nurturing and social well being;
- *individualism versus collectivism*, which describes the relationship between the individual independence and the collective interdependence of a group.

All the four dimensions are a continuum between two extremes and only very few national cultures, if any, are wholly at one or the other extreme. In our research we have adopted Hofstede's approach.

3.2 The Dimensions of Organisational Culture

The concept of organisational culture has been popular since the early 1980s. Schein [1985] stated that organisational culture is the key to organisational excellence. Organisational culture has mainly been created and maintained in existing frameworks by the founders and the leaders of an organisation through their own value system [Bryman, 1992, 1986; Gagliardi, 1986; Schein, 1983]. Organisational culture is often reflected as the result of management activity [Lessem, 1989, Rodrigues, 1998] or is looked at through levels of "practices", like symbols, heroes and rituals [Hodge and Anthony, 1988; Joynt and Warner, 1996; Mullins, 1989]. These characteristics enable differentiation among organisations [Mullins 1989]. Sahni and Rastogi [1995] suggest that organisational culture is defined as the integrated pattern of *human behaviour in a corporation*, which includes the way employees think, speak and act. Others have used the structure of the organisation to explain corporate culture [Lofland and Lofland, 1995].

3.3 Individual Behaviour and National vs. Organisational Culture

The basic assertion in cross-cultural studies is that national culture, expressed in terms of values and beliefs, has a direct impact on organisational culture and individual behaviour [Hofstede, 1985, 1994; Hofstede et. al. 1990; Schein, 1983]. The economical, political and legal environment imposed by governmental rules, the technical environment, such as communication networks, and the socio-cultural environment in which the organisation exists, directly affect organisational culture and functioning of organisations.

Organisational culture in turn affects directly individual behaviour by imposing guidelines and expectations for the members of the organisation. Values of other stakeholders, like employees, also create impact on organisational culture. One of the key issues for managers in international organisations is integration across geographic distance and cultural diversity.

3.4 Global organisations and Universal Culture

The management of cultural diversity is becoming a significant issue for companies of all sizes, not only for global organisations. The emergence of cross-cultural organisations has created a new awareness of the importance of understanding other cultures. The rise of global business, with an increasing number of joint ventures and cross-national partnerships, greater co-operation within the European Union, and the business need to embrace people from a variety of ethnic backgrounds and cultures, have all contributed to the need to develop a cultural sensitivity. Problems between mother organisation and subsidiaries, resistance and low motivation and productivity can arise in international operations because of cultural ignorance or insensitivity [Siakas and Georgiadou, 2003].

Being a global organisation implies having a universal culture [Joynt and Warner, 1996]. The objectives of the global organisations are to create a universal culture in the whole organisation and to integrate multi-domestic operations with individuals who hold opposed work-related values. If an organisation is going to be a truly global organisation the diverse individual work values must converge and be integrated into a common set of values to create a universal corporate culture. One of the key issues for managers in international organisations is *integration across geographic distance and cultural diversity*.

4 The SQM-CODE Model for Software Process Improvement

The SQM-CODE model (Software Quality Management: Cultural and Organisational Diversity Evaluation), which assesses the cultural fit between national culture and organisational culture. The SQM-CODE model comprises two sub-models, namely the C.HI.D.DI. typology and the Authoritarian-Participative model [Siakas, 2002a].

The findings from the C.HI.D.DI typology, which is based on Hofstede' s national cultural dimensions, defines the national culture and predicts a suitable software quality system for the organisation depending on what country the organisation is based. The findings from the Authoritarian-Participative sub-model define the organisational culture and structure.

The SQM-CODE model can be used by organisations developing software in any country. The self-assessment of the SQM-CODE will give a fast response regarding the basic underlying cultural fit or dichotomy between organisational and national culture.

4.1 The First Sub-model: The C.HI.D.DI Typology

By including cultural and organisational issues and by using two of Hofstede's dimensions, namely *Power Distance* and *Uncertainty Avoidance*, a new typology, called C.HI.D.DI. was developed [Siakas and Georgiadou, 2000b]. The C.HI.D.DI typology classifies organisations depending on organisational structure and orientation into four dimensions, called *Clan, Hierarchical, Democratic* and *DIsciplined*. It has been suggested that identification of organisational type will be indicated by considering organisational structure, orientation, leader's roles and management style usually expected in a certain national culture. Hofstede's Power Distance and Uncertainty Avoidance dimensions are used to categorise the countries belonging to a specific typology.

The C.HI.D.DI typology aims to improve the implementation of Software Quality Management systems, by offering guidelines on SPI to organisations developing software. The C.HI.D.DI typology itself proposes suitable software quality management systems for organisations in different cultures and emphasises weaknesses and strengths in the organisation together with guidelines and recommendations for process improvement and for improvements in organisational issues aiming to improve the fit between organisational and national culture. The C.HI.D.DI typology can also be used for assessing both purchasers and customers in order to identify underlying values.

4.2 The Second Sub-model: The Authoritarian-Participative model

The initial empirical results of this research revealed correlations of fundamental organisational characteristics. Based on these organisational characteristics, the Authoritarian-Participative submodel of the SQM-CODE defines the organisational culture and structure based on the attributes described below: These have been derived by results obtained in the field. Findings from a field-study in five countries have already been reported in earlier work [Siakas and Balstrup, 2000].

- ✓ Centralised / decentralised
- ✓ Tightly / loosely controlled
- ✓ Management driven / participative
- ✓ Formal / informal
- ✓ Deep / flat hierarchy
- ✓ Task / people-oriented
- ✓ Process / product-oriented

The Authoritarian-Participative submodel is used as follows: The organisation is asked to map these characteristics into a scale of one - six. The interpretation of the values are that the higher the values are the more bottom-up and participative is the organisational structure and the lower the values are the more top-down and authoritarian is the organisational structure. Since the questions above regarding the organisational characteristics measure similar items, the hypothesis was that the responses would cluster together for each country and thus the total mean value per country would give a single measure.

4.3 The Fit Between National and Organisational Culture

To demonstrate the value of a self-assessment using the previous models the following example will be used: If an organisation is situated in a country with Low Power Distance and Low Uncertainty

Avoidance it will be classified as *Democratic* according to the C.HI.D.D.I typology. A flexible in-house software quality system is predicted to be the most suitable software quality system for an organisation classified as belonging to the Democratic quadrant, because the national culture requires a participative organisational structure with flexible rules. Introducing a Software Quality Management System needs a change in the organisation and acceptance of change will only take place if decision-making is participative.

If the organisation is classified as *Participative* according to the Authoritarian-Participative model then there will be a fit between the national and the organisational culture. If there is not a fit, a deeper investigation need to be undertaken in order to find out what changes are needed either to the organisational structure and other organisational issues, like changes in the processes (SPI), responsibilities and motivation.

5 Verification of the Findings and General Observations

To verify the SQM-CODE Assessment model a field study using the model took place during February and March 2000. It was conducted in a Danish global organisation at three divisions in their software development departments in different countries. During the field-study a maturity assessment also took place in one of the divisions. Observations and interviews (56 in total) with employees in software development on different levels were conducted. Before the interviews the participants had answered the common questionnaire that was especially developed for this research. The assessors were also interviewed, as well as two employees working with a SPI project on overall corporate level. The findings provided evidence to support the national differences found in work-related values between the three countries by using Hofstede's four dimensions and their effect on the awareness. The organisational culture was also taken into consideration and analysed for the same purpose. Detailed reports on the findings and on the application of SQM-CODE can be found in earlier published work [Siakas and Georgiadou, 2000a; 2002; Siakas, 2002b].

The full assessment using SQM-CODE includes a more complex process. A more in depth field-study needs to be employed by an assessment team. Questionnaires, interviews and observations are the tools used in the full assessment. The full assessment results in recommendations and guidelines for introducing or changing an existing Software Quality Management System. It also may propose changes to organisational structure and processes in order to fill possible gap between the national and the organisational culture.

The SQM-CODE model and its two submodels can be used in parallel with maturity assessments, like CMM, SPICE or Bootstrap. Global organisations would benefit of using the C.HI.D.DI assessment model in their subsidiaries. The organisational culture in the mother organisation might not be suitable in other countries. The mother organisation has to be aware of the differences in cultures and be flexible enough to take into consideration differences between the organisational and the national culture. If the values of the employees of the mother organisation are divergent from the values of the employees in a subsidiary, conflicts and dissatisfied employees will most likely be the result.

Within a single culture and/or country certain values, attitudes and behaviours are either favoured or suppressed. A value can either be explicit or implicit. So, how democratically, participatively or hierarchically do organisations and countries design and create software?! And how does this particular development approach influence their own system of values, their working life and their life in general? Furthermore, our findings showed statistically significant evidence that if there is a fit between the organisational and the national culture, then there is a higher employee satisfaction and problems are solved more smoothly. A dichotomy is highly likely to generate dissatisfaction, conflict and ultimate failure of the quality management function in general.

In today's global business environment top quality and just in time approaches together with low-cost products and services have become core values. A critical competitive factor is culture. Emphasis is put on cultural differences not only between employees in the global organisation but also on cultural differences between clients in the global market.

6 Conclusions and Further Research and Development

The aims of this work were to investigate cultural and organisational issues in addressing software quality in general and SPI in particular. This research has shown that culture is a source of competitive advantage/disadvantage and organisational structure and national culture play a crucial role in choosing and implementing a Software Quality Management System. We presented the SQM-CODE assessment model, which assesses the cultural fit between national culture and organisational culture and based on that assessment. The model subsequently aids in choosing a suitable Software Quality Management system depending on the country in which the organisation is situated. The model can be used worldwide, especially by multinational organisations for outsourcing and other activities.

Future research and development will concentrate on searching for more evidence by applying the principles and concepts of SQM-CODE model in large scale projects and organisations. This will increase the knowledge and awareness required for further optimisation and improvisation techniques. That is, to identify which improvement areas combined with national culture, organisation culture and improvement method will cater for best practice for SPI and successful adoption of a SQMS. A network of people involved in teleworking environments and web-based communities would also be an important sample for observations and comparisons towards finding evidence to analyse the working culture generated within virtual organisations. The study of particular work relationships in many countries will assist to collect, analyse and disseminate knowledge about improvement strategies and risk within the industry and society at large.

Acknowledgements: The authors would like to express their sincere thanks to the EuroSPI 2003 reviewers for the corrective and useful remarks. Many thanks to Mr Bo Balstrup for providing the conditions for the field work.

Literature

Biro M., Sztaki, Messnarz R., Sandberg J. (2000): The Perception of Quality Based on Different Cultural Value Systems, *EuroSPI 2000*, Copenhagen

Boehm Barry W. (1973): "Software and Its Impact: A Quantitative Assessment," *Datamation* May

Bryman Alan (1992): *Charisma and leadership in organisations*, London, Sage Publications

Bryman Alan (1986): *Leadership and organizations*, London: Routledge & Kegan Paul

Couger J.D., Adelsberger H. (1988): Comparing motivation of programmers and analysts in different socio/political environments: Austria compared to the United States, *Computer Personnel*, 11(4): pp. 13-17

Couger J.D., Borovitz I., Zviran M. (1989): Comparison of motivating environments for programmer/analysts and programmers in the US, Israel and Singapore in Sprague, R/H. JR (eds.): Proceedings of the 22n^{d} annual Hawaii International Conference on Systems Sciences, *IEE Computer Society Press*, Washington, DC, 1989, pp. 316-323

Couger J. D., Halttunen V., Lyytinen K. (1991): Evaluating the motivating environment in Finland compared to the United States - a survey*, European Journal of Information Systems* Vol. 1, No.2, p.107-112

Gagliardi P. (1986): The Creation and Change of Organisational Cultures: A Conceptual Framework, *Organisation Studies*, 7, 117-34

Groeschl Stefan, Doherty Lis (2000): Conseptualising Culture, *Cross Cultural Management, An International Journal*, Vol. 7, Number 4

Hall Edward T. (1959): *The Silent Language*, Doubleday, New York

Hodge B. J., Anthony W. P. (1988): *Organisational Theory*, Allyn and Bacon, Boston.

Hoecklin Lisa (1995) *Managing Cultural Differences*, Strategies for Competitive Advantage, Addison Wesley Publishers

Hofstede Geert (2001): Culture's consequences: comparing values, behaviours, institutions, and organisations - 2nd Ed. - Thousand Oaks, Calif.; London: Sage Publications

Hofstede Geert, (1994): *Cultures and Organisations, Intercultural co-operation and its importance for survival, Software of the mind,* McGraw-Hill*,* UK

Hofstede G., Neuijen B., Ohayv D., Sanders G. (1990): Measuring Organisational Cultures, A Qualitative Study Across Twenty Cases, *Administrative Science Quarterly*, 35, pp. 286-316

Hofstede G. (1985): The interaction between national and organisational value system, *Journal of Management Studies*, 22m pp.347-57

Hofstede Geert (1984): *Culture's Consequences: International Differences in Work-related values,* Sage Publications

Hofstede G. (1983): The Cultural Relativity of Organisational Practices and Theories, *Journal of International Business Studies*, fall, pp. 75-92

Holmes Monica C. (1995): *The Relationship of Cross-cultural Differences to the Values of Information Systems Professionals within the Context of Systems Development*, PhD Dissertation, Denton, Texas

Hoppe Michael Heinz (1990): *A Comparative Study of Country Elites: International Differences in Work-related Values and Learning and their Implications for Management Training and Development,* PhD Dissertation, University of North Carolina, Chapel Hill

Inkeles A., Levinson D. (1969): National Character: The Study of Module Personality and Socio-Cultural Systems in Lindsay G., Aronson E. (eds.): the *Handbook of Social Psychology*, Volume 4. Addison-Wesley

Ives B., Järvenpää S. (1991): Applications of Global Information Technology: Key Issues for Management*, MIS Quarterly*, Vol 15 March, pp. 33-49

IPSSI Consortium, PIPSI for Managers - Personal Quality Management
Defects, EuroSPI 2000, http:// www.compapp.dcu.ie/ipssi/managers4.pdf

Joynt Pat, Warner Malcolm (1996*): Introduction: Cross-cultural perspectives Managing across Cultures: Issues and Perspectives*, by Joynt Pat, Warner Malcolm (Ed.): International Thomson Business Press

Keen P.W. (1992): Planning Globally: Practical Strategies for Information Technology in the Transnational Firm in Palvia S., Palvia R., Zigli R. (Eds) *The Global Issues of Information Technology Management*, Harrisburg Pennsylvania: Idea Group Publishing, pp. 575-607

Kluckhohn F., Strodtbeck F. (1961): *Variations on Value Orientations*, Evanstone, Illinois: Row, Peterson and Company

Kluckhohn F. (1951): Values and Value-Orientation in the Theory of Action: An Exploration in Definition and Classification in Parsons T, Shils E. A. (eds.): *Towards a General Theory of Action*, Cambridge, MA Harvard University Press

Kroeger A., Kluckhohn F. (1952*): Culture: A Critical Review of Concepts and Definitions,* Harward Business Review, Cambridge

Kumar Kuldeep, Bjorn-Andersen Niels (1990): A Cross-cultural Comparison of IS Designer Values, *Communications of the ACM*, 33.5, May 1990, pp.528-238

Lessem R. (1989): *Global Management Principles*, Prentice-Hall, London

Lofland John, Lofland Lyn H. (1995*): Analytic Social Settings, A guide of qualitative observation and analysis*, Wadsworth Publishing Company.

Mullet Diana (1999): www.unt.edu/benchmarks/archives/1999/july99/crisis.htm

Mullins Laurie J. (1989): *Management and Organisational Behaviour*, Pitman Publishing, London

Oyen Else (1992): *Comparative Methodology, Theory and practice in International social research*, Sage Publications Ltd, 1992

Palvia S., Palvia R., Zigli R. (eds.) (1992): The Global Issues of *Information Technology Management, Harrisburg Pennsylvania*: Idea Group Publishing

Rodriques Carl A. (1998): Cultural Classifications of Societies and How They Affect Cross-Cultural Management, in *Cross-Cultural Management*, Vol. 5 Number 3.

Sahni Ashweni K., Rastogi Anil K. (1995): Transforming Corporate Culture through Total Quality Commitment, *5th world congress on Total Quality*, New Delhi, Feb., pp.42 - 45

Schein E. (1985): *Organisational Culture and Leadership*, London, Jossey-Bass Ltd.

Schein E. (1983): The Role of the Founder in Creating Organisational Culture, *Organisational Dynamics,* 12,1, pp. 13-28

Senior Barbara (1997): *Organisational Change*, Financial Times Management,Pitman Publishing

Siakas Kerstin V., Georgiadou Elli (2003): The Role of Commitment in successful Software Process Improvement and Software Quality Management, *The 11th Software Quality Management Conference,* SQM 2003, 23-25 April, 2002, Glasgow, UK (accepted)

Siakas Kerstin V. (2002a): *SQM-CODE: Software Quality Management – Cultural and Organisational Diversity Evaluation*, PhD Thesis, London Metropolitan University, November 2002

Siakas Kerstin V., Georgiadou Elli (2002): Empirical Measurement of the Effects of Cultural Diversity on Software Quality Management, *Software Quality Management Journal,* Volume 10, No 2, October 2002 issue, pp. 85-95

Siakas Kerstin V. (2002b): What has Culture to do with Software Process Improvement, *Euromicro 2002 Conference*, Dortmund, Germany, Sept 4-6

Siakas Kerstin V, Georgiadou Elli (2000a): Software Quality Management from a Cross Cultural Viewpoint, *Software Quality Journal*, Volume 8, No 2, October issue, pp 85-95

Siakas Kerstin V., Georgiadou Elli (2000b): A New Typology of National and Organisational Cultures to Facilitate Software Quality Management, *The fifth INternational conference on Software Process Improvement - Research into Education and training, INSPIRE 2000,* London 7-9 September

Siakas Kerstin V., Balstrup Bo (2000): A field-study of Cultural Influences on Software Process Improvement in a Global Organisation*, European Software Process Improvement Conference, EuroSPI '00*, Copenhagen, 7-9 Nov.

Standish Group (1998): http: //www.standishgroup.com/visitor/chaos.htm

Tayeb M. (1994): Organisations and National Culture: Methodology Considered, *Organisation Studies*, Vol. 15, No 3, pp. 429-446

7 Author CVs

Kerstin V. Siakas

Kerstin works as a lecturer since 1989 at the department of Informatics at the Technological Education Institution of Thessaloniki, Greece. She is born and grown up in the Swedish part of Finland. She has an extensive industrial experience (since 1975) in software development on different levels from many European countries and mainly from multinational organizations. Because of her multicultural background and her work experience she started to do research in the UK in Software Process Improvement and how culture influences software development. She finished her PhD in November 2002. She has published around 20 papers about her research.

Eleni Berki

Eleni Berki is an Asst. Professor of Group Technologies in Jyväskylä University, Finland. She completed her PhD in Process Metamodelling and Systems Method Engineering in 2001, in United Kingdom. Her teaching and research interests include: Process Metamodelling and Information Systems Engineering, Computational Models and Multidisciplinary Approaches for Software Engineering, Knowledge Representation Frameworks and Requirements Engineering. She has worked as a software designer and consultant in industry, and has a number of academic and industrial project partners in many countries. She has been active in the development, delivery and coordination of virtual and distance learning initiatives in collaboration projects in European and in Asian countries.

Elli Georgiadou

Elli Georgiadou is a Principal Lecturer in Software Engineering at Middlesex University, London. Her teaching includes Software Metrics, Methodologies, CASE and Project Management. She is engaged in research in Software Measurement for Product and Process Improvement, Methodologies, Metamodelling , Cultural Issues and Software Quality Management. She is a member of the University's Global Campus project (developing and offering ODL). She has extensive experience in academia and industry, and has been active in organising/chairing conferences and workshops under the auspices of the British Computer Society, the ACM British Chapter and various European programmes for Technology Transfer and development of joint curricula. She established a Distance Mode Initiative between a UK University and a Hong Kong Institute developing and offering technology-based learning. She has engaged in developing the pedagogic framework as well as the development of materials. She designed and carried out evaluations of various ODL initiatives in the UK, Greece, Spain, Finland, Hong Kong and Cyprus.

Experiences With Managing Social Patterns in Defined Distributed Working Processes

Eva Feuer, Richard Messnarz, Heinz Wittenbrink

Abstract

The TEAMWORK (IST-2000-28162, Jul 2001 – Mar 2003) project trialed an e-working platform on which team-working processes could be agreed , configured, and implemented for the use by 11 organisations in 7 different countries. Finally above 50 different organisations had participated. An analysis of social patterns, and a guidance for project coordinators outlining success criteria for social team management in distributed collaboration/development was produced.

For agreeing the defined working processes we used a process improvement methodology BESTREGIT which starts from business analysis, priority analysis, and finally agrees common business processes and outcomes which can be incorporated into a jointly used e-working platform.

For analysing the social team-factors the project based on a previous study from the CREDIT project (EU MM1032 CREDIT, 1998 – 2000) in which a set of social competencies for software managers in the future were listed.

The results described in this paper are useful for distributed organisations who plan to achieve a defined capability level 3, use an e-working solution for instantiating standard processes among the sites, and need to consider social team factors for successfully managing projects.

Keywords

Defined Processes, Social Patterns, Social Competencies, Social Significances, Social Guideline

1 Introduction

The Teamwork project, IST-2000-28162, was a trial to examine the methodological, technical social and cultural aspects of distributed management in an international, multi language, multicultural environment.

The teamwork project based on a few assumptions which should further social acceptance by teams, and it was planned to analyse social feedback and patterns that would prove these assumptions or/and show further success criteria to be considered.

Assumption 1 – Mission Building and Clear Roles

Using an improvement methodology like BESTREGIT forces teams to think about joint mission statements, joint goals, and agreed roles within team structures (even if the roles act in a distributed environment). Thus we assumed that this clear role assignment and mission will positively impact the team acceptance.

Assumption 2 – Integration Ability

We assumed that by providing defined team-working scenarios and roles the integration of new partners into the project will be easier, allowing more than just the 11 organisations to participate.

Assumption 3 – System Flexibility

We assumed that by providing an environment to allow non-technical users to adapt/tailor the working processes (without programming) on the e-working platform, would increase the acceptance by the non-technical users (decision makers, politicians, etc. who participated as well).

Assumption 4 – Social Competencies

We assumed that there is a key list of social factors that in a distributed team will have to be handled differently than in a traditional management (all people in one building, etc.)

After the installation of the virtual team-working methodology and technology different scenarios were introduced to provide the users with practical applications of the methodology and technology. During these introduction and implementation processes different approaches were applied to analyse the social and cultural skills :

- Collection of Social Feedback and Statistical Analysis
- Social Patterns Analysis and Interviews
- Evaluation of Assumptions (Correlation Analysis)

2 Defined Environment – Comparable Experiments

To perform an analysis as foreseen in TEAMWORK [1],[4],[5],[6] it was necessary first to reach a basis which allowed to reach statistically comparable results. Thus it was necessary to run through steps like

- Joint mission analysis (BESTREGIT methodology)
- Joint goal analysis (BESTREGIT Methodology)
- Joint agreement on working processes (we called them working scenarios) with defined roles, results, work flows (BESTREGIT Methodology)
- Configuration of the e-working platform with the agreed processes (TEAMWORK system)

- Scenarios covered project planning and management, work package management, configuration management (enforced by the system), quality assurance (by defined e-working scenarios and system supported including planning reviews, work package reviews, deliverable reviews, effort deviation reviews, external reviews, non-conformance reporting and corrective actions). (ISO 9001, CMM 1.1 Level 2)

- The working scenarios were agreed as standards among the team, team members played the defined roles, and intergroup coordination was done through a virtual office (keeping track of data and user involvement). (ISO 9001, CMM 1.1 Level 3)

The goals was to start with an e-working environment which in terms of processes, methodologies, and technologies is defined, stable, and offers a basis for analysing social factors. If each team would not follow the same standard platform approach the measures on the social side would be impacted and a controlled experiment (with data that can be compared) would not have been possible.

2.1 Short Introduction to the Underlying Working Platform

It should be noted that what we present here is already the refined platform, as the social feedback led to two major refinements.

The wish for informal working zones (not all should be formal)

This area had to be integrated to allow team discussions, informal exchange of ideas, searching like in a knowledge base, and informal meetings (not following the defined processes). Discussion areas needed to be on different levels keeping different private zones separate.

The wish for generic systems versus wizard based generated systems

We had to withdraw from the idea of a tailorable system where a system administrator changes certain configurations for users, and to develop a wizard and e-working system generator which allowed key managers (also politicians and decision makers) to program e-working processes without having computer knowledge. (see Figure 1).

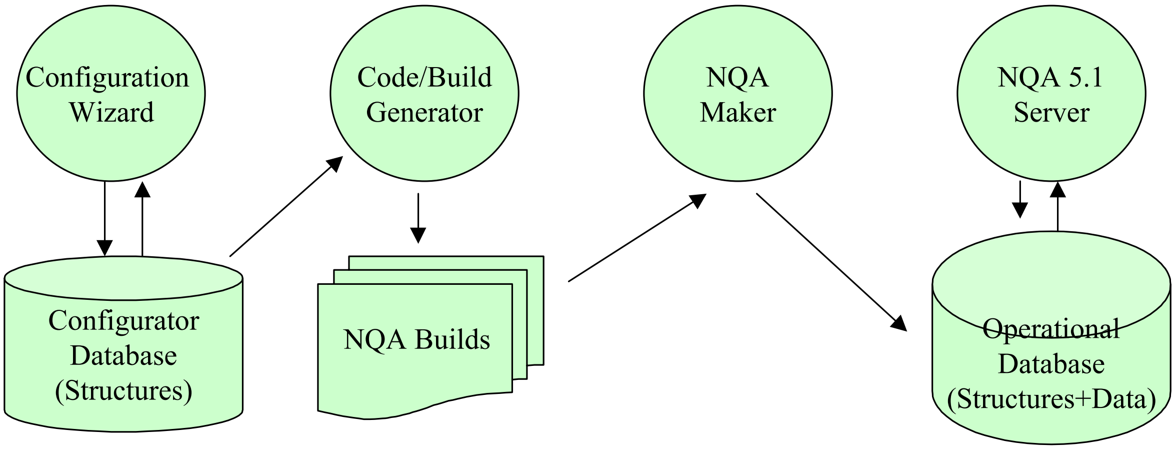


Figure 1: **The New Tool Chain (Manageable by non-computer people)**

Configuration Wizard: This is a wizard like interface developed with PHP, SQL database, and Apache server which allows to -

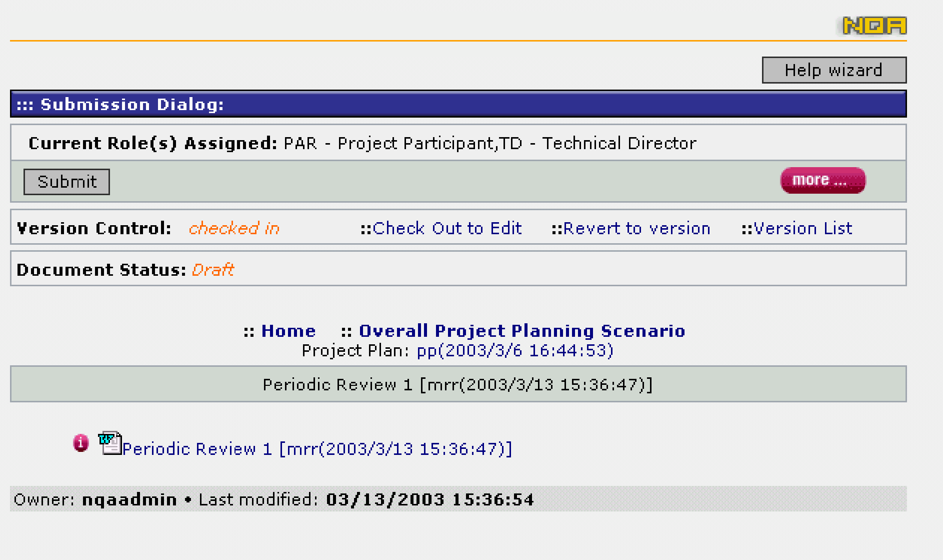
- Enter project types / e-working spaces
- Working phases in project types
- Document types, templates, and flows
- Roles, and submission flows, as well as ownership

- Links between document types

Generator: This is a tool where the user selects which project working types the e-working server should support and then generates a BUILD of NQA. This BUILD contains all regenerated user interface templates and scripts so that NQA can be adapted / tailored automatically.

Maker: This is a tool which the NQA Administrator calls and automatically the new BUILD is imported to the server. After a restart of the server all new structures are automatically supported.

NQA 5.1: This is the actual e-working server supporting the scenarios (project management, configuration management, etc.).



::: Submission Dialog:

Current Role(s) Assigned: PAR - Project Participant,TD - Technical Director

Version Control: checked in ::Check Out to Edit ::Revert to version ::Version List

Document Status: Draft

:: Home :: Overall Project Planning Scenario
Project Plan: pp(2003/3/6 16:44:53)

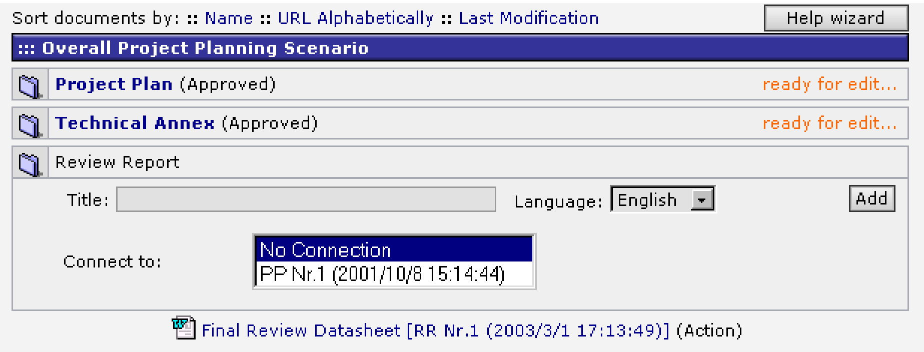
Periodic Review 1 [mrr(2003/3/13 15:36:47)]

Periodic Review 1 [mrr(2003/3/13 15:36:47)]

Owner: **nqaadmin** • Last modified: **03/13/2003 15:36:54**

Figure 2: Screen Shot of a Document Submission Dialog (Checked in State in Version Control)

...



Sort documents by: :: Name :: URL Alphabetically :: Last Modification

::: Overall Project Planning Scenario

Project Plan (Approved) ready for edit...

Technical Annex (Approved) ready for edit...

Review Report

Title: Language: English

Connect to: No Connection
PP Nr.1 (2001/10/8 15:14:44)

Final Review Datasheet [RR Nr.1 (2003/3/1 17:13:49)] (Action)

Figure 3: Screen Shot of Document types in an E-Working Project Scenario

One major technological finding was **that users are willing to think process and team-working driven but require a platform that can be generated as needed in minutes.** This is especially true in case of decision makers, politicians, etc.

3 Social Analysis

3.1 The Approach

On the one hand we performed a search in literature to find out experiences with team-working concerning social factors and team psychology aspects. This resulted in a list of key social criteria which might have an impact on the team-working in general.

Secondly the social team in the Teamwork project did a feedback analysis during the trials and statistically analysed the feedback to identify if social / human factors influence the way how people perceive team-working (especially over the Internet).

And thirdly the social team in the Teamwork project applied the social patterns analysis methodology and also performed a set of interviews which were analysed to draw a number of conclusions concerning the social factors influencing team-working.

In total **59 organisations have been involved in the trials** who each provided users to the different projects run via the system. These 59 organisations have been further split into public service, research networks, and software engineering.

We also had **42 different projects** running through the system using the defined environment and managed by a virtual team leader. The team size of each project varied between 13 and down to 2 different organisations involved.

Each organisation provided in average 9,3 users to the different projects so that a total of 550 users were involved.

3.2 Statistical Analysis and Significance Evaluation

The questionnaire consisted of 4 parts. Each part contained 10-15 questions. The questionnaire was available to all users online. The questionnaire was structured to address general data, and the three components of the TEAMwork solution:- Methodology, Technology and Social / Cultural factors [1],[4],[5],[6].

The methodology of the questionnaire is based on the hybridised grounded theory approach. There are direct and multiple choice questions. The possible answers for the multiple choice questions were not scaled on the questionnaire, but at the evaluation part a value was attach to each response.

We have done further statistical analysis to determine the connections and the importance of cultural dimensions. We have used a standard statistical package called BMDP and applied the so called Pearson Chi square test for determining the significance of the different attributes.

We have found

- age sensitive problems,
- gender sensitive problems,
- organisation type sensitive problems,
- experience sensitive problems.

But we did not find language or job role sensitiveness.

3.2.1 Age Sensitive Significance

Communication Related

We have found that the young people needed significant less face to face contact with team members, than older people just the mechanisms provided by the TEAMWORK environment. We have found that the young people think that most of the communication to the team-leader can be conducted using the environment, most of the older people think that only a small part of the communication can be conducted using the environment.

Communication Related - Possible Interpretation

This could mean that if you have a mixed team of young and older people you need to be aware that you need extra effort (despite of having all online) to involve both groups into other means of communication than technology as well. Not doing that will lead to resistance of the older team members.

Management Style Related

It is very significant that young people are happy with the more democratic management style imposed by team-working based e-working.

Management Style Related - Possible Interpretation

This could mean that if the majority in your team are younger team members you can implement flat management structures and distributed working easily, in case of a mix of old and young staff you need to establish clear hierarchies beside the flat team-working structure (based on role collaboration and communication).

3.2.2 Gender Sensitive Significance

Project Control Related

The statistical analysis shows that it is highly significant that women believe that the e-work solution could hinder the control of their work, men believe it does more for them.

Team Spirit Related

Another significant difference is how women and men feel about the effect of this tool to enhance the teamwork spirit, women found it enhancing or no effect, for men it was felt more that it negatively affects team spirit.

Possible Interpretation

This shows a quite funny situation. While women think it widens the team spirit, they still would not trust a system to take over the control. On the other site men like a system to take control but still need e.g. a coffee area to enable a team spirit.

This could mean that depending if a man or woman is the team leader the way of how the systems will be used and team spirit is built up might be different.

3.2.3 Organisation Sensitive Significance

Management Style Related

It is significant that people are happy with the more democratic management style of the TEAMWORK environment. Actually people from university, public service, and research centers are happier, than from large enterprises.

We have found, that there is a significant difference how people from university, public service, and research centers feel the effect of this tool to enhance teamwork spirit, for people from large enterprises it was felt more that it gives negative affect.

Management Style Related - Possible Interpretation

This could mean that if you set up a team and you involve a large organisation it is not sufficient to just define your agreed team-working processes, but also to integrate properly the existing hierarchies inside the collaborating large organisation.

Project Control Related

It is significant that people from university, public service, and research centers believe that e-working enhances the control of their work, people from large enterprises believe it does hinder.

Project Control Related – Possible Interpretation

This could mean that if you set up a team and you involve a large organisation it is not sufficient to just define your agreed team-working processes, but also to integrate properly the existing hierarchies inside the collaborating large organisation.

Team Communication Related

It is significant that most of the team members from large enterprises and university think that there is good cooperation in their team and from SME the team members think that the cooperation is not adequate.

Furthermore it is significant that most of the team members from large enterprises and university think that the e-working tool increases the frequency of collaboration in their team and from SME the team members think that it reduces the collaboration.

Most people from SME, large enterprises and university think that less than 50% of communication to the project leader was conducted using the e-working system, according to the university enterprise training partnership and the research centers more than 50% of communication to the project leader was conducted using the system.

Team Communication Related – Possible Interpretation

This might have been caused by the fact that the SMEs used the system mostly as an Intranet system. Users then had the feeling that this system will be seen as a substitution of the normal (e.g. coffee kitchen based) communication.

This could mean that if the system is not used in a distributed environment you need to be careful to not go into a resistance situation where some staff members will run defence mechanisms. (to defend their own social environment).

Innovation Related

It is significant that people from university, public service, and research centers believe that the Teamwork work style encourages an innovative approach to solving problems, people from large enterprises believe it does not.

It is significant that people from university, public service, and research centers believe that e-working helps them to improve the understanding of team roles.

We have found, that there is significant difference how people from university, public service, and research centers feel that there are adequate opportunities in TEAMWORK to express their views.

Innovation Related – Possible Interpretation

It is an observation (also emphasised again in a discussion after the project by a large enterprise) that larger organisations want to have more impact and decision power on projects and innovations, while the distributed team-working concepts (role based) only gave them equal weight among all team members (independent from their size).

This could mean that in case of e-working collaborations the larger organisations (who have bigger project shares) should get higher access rights than a normal user (like an area specific admin).

3.2.4 Experience Sensitive Significance

Technology Acceptance Related

It is significant that people with no or some experience believe that the Teamwork solution enhances the control of their work, many people with great experience believe it does hinder.

It is significant that people with no experience believe that the Teamwork work style encourages an innovative approach to solving problems, people with some or great experience believe it does not.

Technology Acceptance Related - Possible Interpretation

The more technology driven a user is the more the user bases the acceptance on technology matters. For instance, decision makers and managers of innovation centers scored as less experienced technology users but in fact they thought first process driven and then searched if ceratin upload function should additionally be supported. Where both groups agreed was improvements on the User Interface, not in details.

This could mean that if you mix non computer specialists (decision makers , politicians) and computer experts in one team then you need to manage process driven with the decision makers and try handling all technology issues (further technology wishes) at the same time. Be aware of that not so easy management task to get both groups into one boat.

3.2.5 Language Sensitiveness

We did not find any significance according to the languages spoken. First of all this could mean that for most of the persons who were interviewed English was the second or the first language spoken, they have experience in their work with English and secondly that the English language of the TEAMWORK solution is well defined, easy to understand.

3.2.6 Job Role Sensitiveness

We did not find any significance according to the job role. This means that the TEAMWORK solution is a very democratic approach and gives equal opportunity for all, not depending on the job role. This also means that the level of training was adequate for all job roles. The implemented job roles were:

Business Manager

Project Manager

Safety Manager

Quality Manager

Requirements Engineer

Procurement

Administrator

3.3 Social Patterns Analysis and Interviews

The methodological difficulty of describing the social requirements of a successful TEAMwork project is very similar to the typical problem of knowledge management in companies or other organisations: the transformation of "tacit" or "implicit" knowledge into explicit knowledge. In addition to the explicit TEAMwork methodology the participants in the trials used a lot of implicit knowledge, and by the combination of both, the outcome of the trials was also a "social success", the formation of a multilingual and multinational "community of praxis".

3.3.1 Design Patterns as a New Method for Transforming Implicit into Explicit Organisational Knowledge

In this paper we are using the naming and description of "organisational" design patterns [2],[3] as a technique for the transformation of implicit knowledge into explicit knowledge. It is the original purpose of design patterns to offer a language for the description of a knowledge that is already practically used and embodied by certain artifacts. "Patterns and Pattern Languages are ways to describe best practices, good designs, and capture experience in a way that it is possible for others to reuse this experience." (Patterns Homepage, http://hillside.net/patterns/). Patterns are offering “metarules” or guidelines on a level above the level of the direct application of a methodology or the technical solution of a problem. For our purposes design patterns are presenting two basic advantages:

- 1. they are described for practical purposes and can be used or applied in ordinary "working contexts" and
- 2. they are generative rules: the repetitive and sometimes recursive application of simple patterns can yield very complex results.

By identifying patterns that were applied in the course of a successful application of TEAMwork we are able to offer parts of a "toolbox" or "vademecum" to organisations who want to apply TEAMwork in the future.

3.3.2 Focussing on the Structure of a Social Space

In the qualitative interviews we did not want to collect the same information again which we collected already on methodology and technology in the user feedback. The aim of the interviews was to describe properties of the social space of the TEAMwork trials and to relate the features of this space to TEAMwork as a methodology. We were focussed on three aspects of TEAMwork as a social space:

- 1. Awareness
- 2. Collaboration and cooperation
- 3. Planning and representation of work

3.3.3 Going More in-Depth – Social Factors Checkpoints

The description of TEAMwork as a social space is related to the personal and social requirements that are expressed in chapter 2.3 of deliverable 2. The focus there are the individual skills. The focus of our analysis is the modelling of a social space. We wanted to show to what extent the social space of the TEAMwork trials is a result of the use of the TEAMwork methodology, which other factors have to be taken into account, and how these other factors can be expressed. The ultimate goal is to give future users of TEAMwork guidelines for the shaping of social spaces in connection with TEAMwork.

The selected social skills were taken from the skill set developed in the previous MM 1032 project CREDIT:

2.3.1.1	Unit SWE.010 Social Skills
2.3.1.1.1	Element SWE.010.001 - General
2.3.1.1.2	Element SWE.010.002 - Reflection
2.3.1.1.3	Element SWE.010.003 - Defence mechanisms
2.3.1.1.4	Element SWE.010.004 - Attitudes
2.3.1.1.5	Element SWE.010.005 - Co-operation
2.3.1.2	Unit SWE.011 Team Roles
2.3.1.2.1	Element SWE.011.001 - Technical team roles
2.3.1.2.2	Element SWE.011.002 - Roles connected to personal characteristics

2.3.1.2.3	Element SWE.011.003 - Leadership
2.3.1.2.4	Element SWE.011.004 - Power basis
2.3.1.3 Unit SWE.012 Group Dynamics	
2.3.1.3.1	Element SWE.012.001- Social influence
2.3.1.3.2	Element SWE.012.002- Social facilitation
2.3.1.3.3	Element SWE.012.003 - Conformity
2.3.1.3.4	Element SWE.012.004 - Groupthink
2.3.1.3.5	Element SWE.012.005 - Avoiding groupthink
2.3.1.3.6	Element SWE.012.006 - Phases in group development
2.3.1.3.7	Element SWE.012.007 - Group development
2.3.1.3.8	Element SWE.012.008 - Politics (organizational- and group-level)
2.3.1.3.9	Element SWE.012.009 - Team building

Figure 4: CREDIT MM 1032 – Required Social Skills of Future Software Managers

3.4 Selected Results of the Social Patterns Analysis

A full deliverable of 46 pages has been produced as a sub-deliverable of deliverable 5 in the IST-2000-28162 project. We include here a list of core social patterns which are important for distributed development , the study identified a many more patterns.

3.4.1 Core Organizational Design Patterns

COMMUNITY OF TRUST

„If you are building any human organization Then: you must have a foundation of trust and respect for effective communication at levels deep enough to sustain growth.“

Remark: The interviews and our experience show that a community of trust was established during the trials. The formation of a “community of trust” was the result of meetings creating positive relation between the participants, of a common interest in the success of the project, and an efficient and open management style. For applications of TEAMwork in the future it should be explicitly checked which factors in an organisation could endanger mutual trust. It is decisive to create and maintain human relationships between the participants around the formalised communication via the NQA server.

WORK FLOWS INWARD

„If you want information to flow to the producing roles in an organization Then: put the developer at the center and see that information flows toward the center, not from the center.“

Remark: This pattern was realised by the configuration of the roles of the participants. The result was, that none of the interviewed persons complained about missing information. In the future this pattern should be used explicitly when flows of information in a TEAMwork project are defined.

Note: Role based communication models lead to a fact where more important roles also receive more information flows, while in old traditional style top-down diagrams only positioned managers higher than others and defined restricted reporting paths.

TEAM PER TASK

„If A big diversion hits your team, Then: Let a subteam handle the diversion, the main team keeps going.“

Remark: This pattern was a guiding principle of the organisation of the trials with different teams working in different domains and subteams of these teams responsible for individual tasks. During the trials there was no threat of a major diversion. In future applications of TEAMwork an „emergency team“ should be defined and nominated in order to be able to handle such cases.

3.4.2 Piecemeal Growth Patterns

SCENARIOS DEFINE PROBLEM

„If you want a good characterization of customer needs Then: use scenarios to define the problem.“

Remark: This pattern is explicitly realised by TEAMwork. The role definition is always done via scenarios. It is one of the TEMwork core patterns that are implemented by every TEAMwork application.

FIRE WALLS

„If you want to keep your developers from being interrupted by extraneous influences and special interest groups, Then: impose a Fire Wall, such as a manager, who "keeps the pests away."

Remark: This pattern was implicitly realised during the role configuration for the trials. TEAMwork implements this pattern nearly in a automatic way because TEAMwork defines the circulation of information explicitly. For the trials as a whole this pattern was implemented by the overall project management handling external communication, e.g. the communication with the EU-representatives.

UNITY OF PURPOSE

„If a team is beginning to work together, Then make sure all members agree on the purpose of the team.“

Remark: For the TEAMwork trials as a whole this pattern was realised during the preparation of the project and the meetings of all participants in the first phase of the project. For the single trials it was realised by the definition of scenarios and goals. This pattern is a core pattern that is implemented by Bestregit: the definition of roles is undertaken on the base of a common mission statement.

HOLISTIC DIVERSITY

„If Development of a subsystem needs many skills, but people specialize, Then: Create a single team from multiple specialties.“

Remark: This pattern was implicitly realised by the whole TEAMwork team and the communication of people with very different skills and professional background. Here again TEAMwork seems to be especially appropriate for heterogenous teams.

The role based approach of TEAMwork makes this pattern a useful supplement to the TEAMwork methodology. The roles themselves say nothing about individual skills of the TEAM members. The combination of people with different skills in a team is especially necessary if the team has to fulfil tasks that cannot be completely formalised and represented by the configuration of the NQA-Server.

DOMAIN EXPERTISE IN ROLES

„If you need to staff all roles, it's difficult to determine how to match people to roles to optimize communication. Therefore: Match people to roles based on domain expertise, and emphasize that people play those roles in the organization.“

Remark: This pattern was implemented during the configuration process, because role were defined by a remodelling of existing practices. It should be explicitly used for TEAMwork configurations . It should be explicitly used in cases whre roles are defined for new scenarios.

ENGAGE QUALITY ASSURANCE

„If developers can't be counted on to test beyond what they already anticipate what might go wrong, Then: engage Quality Assurance as an important function."

Remark: This pattern is explicitly implemented by TEAMwork and can be regarded as another core pattern for NQA and Bestregit.

GROUP VALIDATION

If you want to avoid being blindsided in quality assurance Then:

ENGAGE CUSTOMERS and DEVELOP IN PAIRS to validate the system.

Remark: This pattern is implemented by the validation and review processes in the application of NQA. It can also be understood as a core pattern.

3.4.3 Organisational Style Patterns

STABLE ROLES

„If you have to deal with project disruptions, Then: keep people in their primary roles, and deal with disruptions as temporary tasks."

Remark: This is one of the patterns that are explicitly realised by TEAMwork with its role and scenario based method of project management and Project organisation. As well as "Engage Quality Assurance" and "Scenarios Define Problems" it can be regarded as a "core pattern" of TEAMwork.

RESPONSIBILITIES ENGAGE

„If central roles are overloaded but you don't want to take them out of the communication loop Then: intensify communication more among non-central roles to deload the central roles"

Remark: This is one of the patterns implemented by and via the usual role based TEAMwork configuration. The TEAMwork scenarios are not hierarchical and therefore distribute communication between the team members in a goal oriented way. This is one of the TEAMwork „core patterns".

DE-COUPLE STAGES

„If stages are too interleaved for the good of some high-context development where phases can be separated to increase parallelism, Then: serialize process steps, with well-defined handoffs between steps."

Remark: This is again a defining or core pattern for TEAMwork. It is a characteristic feature of TEAMwork that the phases of a work process are clearly separated.

UPSIDE DOWN MATRIX MANAGEMENT

„If the right skills and resources don't seem to be applied to a particular aspect of the work Then: go beyond corporate structures to leverage teams in other organizations (customer, partners, other internal organizations)

Remark: This pattern, also a core pattern, regards the very reason of the building of a virtual team with the TEAMwork technology. The virtual organisation is not simply a transposition of a non virtual organisation into the virtual or digital space; it can create new teams or organisations that integrate roles or persons who in the „analogous world" are clearly separated. An example is the integration of a customer into a TEAM that was practiced during the trials in the defense domain. The chances of completely new types of organisations and businesses in the virtual space are described by the „Cluetrain manifesto" (http://www.cluetrain.com/)

Note: This highlights that innovation needs sometimes a kind of revolution from the bottom driven by an innovative brain. Networked structures better enable the release of such innovative forces.

3.4.4 People and Code Patlets

ARCHITECT CONTROLS PRODUCT

„If a project has a long life, then you need someone to carry the vision forward. Therefore: The architect is an important role as long-term keeper of architectural style.“

Remark: This pattern is directly formulated for the requirements of the software industry. During the trials it was an important factor that with ISCN the architects of TEAMwork were also responsible for the implementation. For the design of social spaces around TEAMwork it will be essential that the designer or architects are also team members, capable to correct in the case of erroneous deviations from the orginal purposes and to transfer the original ideas to new situations.

Note: The job role architect stands for the responsibility to define common agreed processes and who sets these processes up for the teams as an e-working support environment. This job role then also integrates the different deliverables from the different partners who collaborate through the environment.

ARCHITECT ALSO IMPLEMENTS

„If an architect is on an ivory tower, they are out of touch; yet someone needs to take the big and long view and reconcile it with practice. Therefore: the architect is materially involved in day-to-day implementation.“

Remark: During the TEAMwork trials this pattern was successfully implemented, because the architects of TEAMwork were members of the team. The designers of future TEAMwork teams should work in the teams they created and „eat their own dogfood“, in order to be aware of the daily work requirements.

STANDARDS LINKING LOCATIONS

If you have geographically separated development, Then: use standards to link together parts of the architecture that cross geographic boundaries.

Remark: This core pattern defines a major reason for the use of Bestregit an NQA in virtual environments.

4 Evaluation and Outlook

Using the organisational design pattern language defined by Harrison, Coplien er. al. we can name a set of core patterns that define TEAMwork as a social space. These core patterns are implicitly implemented by TEAMwork. They could be used as starting points for a pattern oriented description of TEAMwork. The TEAMwork core patterns are:

1. Information flows Inwards
2. Scenarios define Problem
3. Unity of Purpose
4. Engage Quality Assurance
5. Group Validation
6. Stable Roles
7. Responsibilities engage
8. De-couple stages
9. Upside down matrix management
10. Standards linking locations

The rest of the patterns described in the TEAMwork deliverables can be used as a base for general guidelines or „rules of thumb“ for the implementation of TEAMwork with regards to its social aspects.

We can articulate a basic set of manager guidelines as follows:

1. Identify factors endangering the creation of a community of trust

2. Create and maintain positive human relations between the persons that will communicate with each other via the TEAMwork server

3. Define existing practices, that have to be remodeled, explicitly and set up a hierarchy of importance

4. Define the flow of information as centered around the productive people

5. Create emergency teams that allow the rest of the team to continue working in the case of a crisis

6. Check whether TEAMwork is adequate for the size of organisation. If necessary assemble several sub-projects to achieve the critical mass where the use of TEAMwork makes sense

7. Use customer roles or "customer proxies" in order to relate work processes to customer needs

8. Define roles to keep disturbing factors away from the productive people in an organisation

9. Avoid uniformity in team building, create teams with members of different age, gender, experience and culture, if possible

10. Define public roles in order to foster coherence for larger teams

11. Combine multiple skills in teams amd subteams

12. Make an account of the domain expertise of the team member and use this expertise for the assignment of TEAMwork roles

13. Define as few roles as possible in team or subteam

14. Make the producer roles the central roles in team

15. Create subteams to reduce the complexity of an organisation

16. When you set up a team structure, let it primarily reflect the structure of the business domain,

17. Form subteams of people at the same geographical location, if possible

18. Start the cooperation of virtual teams with a face-to-face meeting

19. Create channels and opportunities for informal communication beside the NQA-Server

20. Limit the exchange of documents and information in normal work situations to a maximum of seven persons and roles. The optimal number is between 3 and 7.

21. Couple the roles of team members tightly, if you want to achieve high efficiency

22. The architects or builder of a team should also be team members during the realisation of a project.

23. When TEAMwork is implemented in a large and/or distributed organisation, define an archtecture team that takes care of the structure of the organisation as a whole.

24. Do not endanger the coherence of an organisation by publicly discussing reasons for management decisions that have to be taken independently from the will of the participants

25. Confer responsibility for artefacts as a whole (products, sub-products, deliverables) to individual team members

We plan to evaluate these social patterns further and hope to find further projects or partners to exploit such data.

5 Acknowledgements

We would like to thank the European Commission for funding the IST-2000-28162 project TEAMWORK, and we would like to thank all (other than the authors) TEAMWORK partners: Brian Foley, Eugene O'Leary, Nuala O Shea, Tecnet, Ireland, Gonzalo Velasco and Nacho Sanchez, FGUVA Spain, Romana Vajde Horvath, Prof. Roszman, Univ. Maribor, Slovenia, Andrea Fenz, Gerald Pichler, APS, Austria, Bruno Wöran, Danube, Austria, Hans Staudte, Peter Luttenberger, Hyperwave , Germany, Ricardo Moyano, GMV, Spain, John Elliott, Lisa Tipping, QinetiQ, UK.

6 Literature

[1] Feuer E., Messnarz R., Best Practices in E-Commerce: Strategies, Skills, and Processes, in: Proceedings of the E2002 Conference, E-Business and E-Work, Novel solutions for a global networked economy, eds. Brian Stanford Smith, Enrica Chiozza, IOS Press, Amsterdam, Berlin, Oxford, Tokyo, Wash-ington, 2002

[2] Harrison and Coplien, The online book about organisational patterns, http://www.easycomp.org/cgi-bin/OrgPatterns?BookOutline, June 2003

[3] Project EASYCOMP (IST Project 1999-14191, homepage: http://www.easycomp.org/)

[4] Messnarz R., Stubenrauch R., Melcher M., Bernhard R., Network Based Quality Assurance, in: Proceedings of the 6th European Conference on Quality Assurance, 10-12 April 1999, Vienna , Austria

[5] Messnarz R., Nadasi G., O'Leary E., Foley B., Experience with Teamwork in Distributed Work Environments, in: Proceedings of the E2001 Conference, E-Work and E-commerce, Novel solutions for a global networked economy, eds. Brian Stanford Smith, Enrica Chiozza, IOS Press, Amsterdam, Berlin, Oxford, Tokyo, Washington, 2001

[6] M. Biro, R. Messnarz, A. Davison (2002) The Impact of National Cultures on the Ef-fectiveness of Improvement methods - The Third Dimension, in Software Quality Professional, Volume Four, Issue Four, American Society for Quality, Sep-tember 2002

7 Author CVs

Eva Feuer

Éva FEUER is Head of Quality Management, senior research associate at the Computer and Automation Research Institute. She has more than 20 years of IT and project management experience.
M.Sc. in mathematics and computer science (Jozsef Attila University, Hungary). She is certified ISO 9000 TÜV CERT auditor and BOOTSTRAP software process assessor. She managed Hungarian participation in European multinational projects and organisations (European Software Institute, Bootstrap Institute, Idealist-East Leonardo projects, Pass ESSI Pie project, Vasie ESSI project). She is head of the Information Society Technologies Liaison Office for the European Union's 5th Framework Programme in Hungary. Her main R&D fields are: information systems, software development, quality management, software process assessment, process improvement and software product quality audit. She has publications in international scientific journals and conference proceedings in the field of software quality management.

Richard Messnarz

Dr. Richard MESSNARZ (rmess@iscn.com) is the Executive Director of ISCN LTD. He studied at the University of Technology Graz and he worked as a researcher and lecturer at this University from 1991 - 1996. In 2 European mobility projects (1993 and 1994) he was involved in the foundation of ISCN, and he became the director of ISCN in 1997. He is/has been the technical director of many European projects: PICO - Process Improvement Combined Approach 1995 - 1998, Bestregit - Best Regional Technology Transfer, 1996 - 1999, TEAMWORK - Strategic Eworking Platform Development and Trial, 2001-2002, Media-ISF – e-working of media organisation for strategic collaboration on EU integration, 2001-2002 . He is the editor of a book "Better Software Practice for Business Benefit", which has been published by IEEE (www.ieee.org) in 1999 (the leading research publisher in the USA). He is the chairman of the EuroSPI initiative and chair of the programme committee of the EuroSPI conference series. He is also a lead SPICE assessor and has worked as consultant over 12 years for many automotive firms.

Heinz Wittenbrink

Heinz Wittenbrink - studied Literature, Linguistics and Philosophy in Münster, Cologne and Paris. As editor and executive editor, he was responsible for numerous encyclopedic works of the Bertelsmann Lexikon Verlag, published in print as well as on CD-ROM. He is now in charge of the product development of the encyclopedic website www.wissen.de. Heinz developed within TEAMWORK the social patterns analysis approach. heinz.wittenbrink@bisdato.com

Quality Improvement through the Identification of Controllable and Uncontrollable Factors in Software Development

Elli Georgiadou[1], Kerstin Siakas[2], Eleni Berki[3]

[1] Middlesex University, School of Computing Science, Trent Park, London, N14 4YZ, UK (e.georgiadou@mdx.ac.uk)

[2] Technological Educational Institution of Thessaloniki, Department of Informatics, P.O. Box 14561, GR-54101 Thessaloniki, Greece (siaka@it.teithe.gr)

[3] Department of Computer Science and Information Systems, P.O.Box 35 (Agora), FIN-40014 University of Jyväskylä, Finland (eleberk@cc.jyu.fi)

Abstract

The software engineering community has moved from corrective methods to preventive methods shifting the emphasis from product quality improvement to process quality improvement. Inspections at the end of the production line have been replaced by design walkthroughs and built-in quality assurance techniques throughout the development lifecycle. Process models such as the Spiral, V, W and X-Models provide the principles and techniques for process improvement which, in turn, produces product improvement.

Factors that affect the quality of software need to be identified and controlled to ensure predictable and measurable software. In this paper we identify controllable and uncontrollable factors and provide empirical results from a large industrial survey, as well as conclusions relating to the models and metamodels for the estimation, measurement and control of the totality of features and characteristics of software.

Keywords

Metamodelling, process improvement, controllable factors, uncontrollable factors, software quality, development methods, soft & formal methods, software metrics, culture

1 Introduction

Software is at the heart of most modern businesses. Business success depends on the quality, the cost and the timeliness of the software they use. A Software Quality Management System is the enabling mechanism within an organisation which co-ordinates and controls the functions needed to achieve the required quality of product or service as economically as possible. It will involve every organisational function that directly or indirectly affects a delivered product or service [DTI, 1992]. For a successful implementation of a Software Quality Management system, weak areas in the current situation have to be identified and gradually improved by introducing a formal process suitable for the specific organisation.

A Software Quality Management system requires that software engineering knowledge and discipline will be applied at all phases of the development life cycle in order to assure software quality. In Software Quality Assurance (SQA) software metrics are used to help numerically determine the quality of both the process and the products together with an independent verification and validation (V&V) usually carried out by the software quality assurance engineers. The right measures visualise different problems in the organisations and after taking action they are a means of control. Verification is the process of determining if the products of a given phase of the software development cycle fulfil the requirements for the next phase regarding consistency, correctness and completeness. Validation is the process of evaluating software throughout its development process to ensure compliance with software requirements. Thus software quality assurance engineers are required to possess sufficient domain knowledge so that they can evaluate the completeness and correctness of system requirements. They must also have the ability to determine whether the design has incorporated all requirements accurately. Finally they are responsible for advising management when or whether a product is reliable and meets quality standards.

There are essentially two approaches that can be followed to ensure product quality, one being quality assurance of the process by which a product is developed (ISO 9000:2000), and the other being the evaluation of the quality of the end product (ISO 9126). Both approaches are important and both require the presence of a Software Quality Management system. The choice of Software Quality Management system, processes, standards and metrics greatly depends on the organisational culture and the maturity level of the organisation [Siakas, 2002].

During the last two decades, there has been a change from concentrating on the product to concentrating on the development process. Software Process Improvement (SPI) has become a practical tool for companies where the quality of the software is of high value [Järvinen, 1994]. In a Technical report Herbsleb et al. [1994] showed (with results from 13 organisations) that due to SPI the products (the number of post-release defect reports were used as a measure) and business value (ROI, Return On Investment) was improved. It is generally considered that, a well-documented and a repeatable process is essential for developing software products of high quality. There is also evidence that the use of standards and process assessment models has a positive impact on the quality of the final software [Kitchenham and Pfleeger, 1996] Evidence for the emphasis on process is also hat ISO certification does not certify product quality but that a stated and documented process is followed.. Similarly, well-known and recognised assessment models like CMM [Paulk et al., 1993; Paulk, 1993; 1995], BOOTSTRAP [Haase, 1992; Haase and Messnarz, 1994; Kuvaja et al.,1994].

2 The Selection of a Suitable System Development Method

Among the most important criteria for selecting a method is its ‘suitability for modelling the application domain’ and the ‘ability to transform its characteristics in a consistent and correct manner to a computable design/software architecture’ [Jackson & Zave, 1993]. Different systems need different techniques to *capture* their properties and similar or different languages to *model* and *preserve* them. Most systems’ developers either use methods (with or without automated tools), which are technically oriented (hard methods) or try to concentrate mainly on human factors by using human activity oriented methods (soft methods).

According to Holcombe and Ipate, there is *“little empirical evidence of the superiority of one method over another in large-scale projects”* and moreover there *is “a crisis of intellectual respectability in the subject”*. They continue supporting that *“not only the evaluation of the methods used is weak, the selection of the types of system and problem to focus on is very restrictive. In order to convince, in a scientific manner, that method A is better than method B in large design projects (and that is where the problems are), we must present rigorous evidence drawn from carefully controlled experiments of suitable complexity. This is, more or less, impossible in practical terms. Is there an alternative approach? The use of theoretical models of computing systems can provide some alternative approaches ...”* [Holcombe & Ipate, 1998].

2.1 Limitations of IS development methods and associated metamodels

There are two important problems associated with IS methods currently available. Firstly, the absence of the *interconnections between hard and soft problems,* which inevitably leads to inadequate information systems with problematic functionality. Secondly, the design or software architecture of a system lacks those *computational characteristics* because they are not specified as an integral part of it. These characteristics make it reliable, easily re-engineered and maintained. We might ask whether there is any all-stakeholder-oriented method among the plethora of methods to adequately capture all the above requirements. And if so, is there any adequate automated support through an Integrated Computer Assisted Software Engineering (I-CASE) environment during all the phases of the software engineering lifecycle?

Development methodologies in order to be reliable, need to also have solid foundations in their generic structure for testing and mapping of the requirements from design to implementation in an expressible, correct and consistent way. They should also cater for systems and method re-engineering with principles that accommodate change regarding soft and hard components of the system such as people’s opinions, systems’ processes, and so on. One of the most important properties to establish quality assurance of the process models that handle the previous is the ability to provide *formal testing* which is also *absent* from existing frameworks and *metamodels* [Berki, 2001]

Moreover, considering the principles and different levels of metamodelling abstraction, that Method Engineering as a discipline caters for, none of the metamodelling environment and notations handles metadata *and* metaprocess specification aspects for method modelling considering methods as dynamic models. This is an invaluable property allowing modellers to keep control on both process metamodelling and metadata of a method because they can express it at the same time. Arguably, when we built a method, we need a general and generic process model of abstraction such as to offer us both generality and specialisation for different application domains [Berki *et al.*, 2001].

Method Engineering is of course a new division/discipline in software development and its emergence was clear while there was a need to improve process models (methods) for IS development. However, the efforts were concentrated mostly on issues such as integration of methods, flexibility of method construction, new theories on requirements management and so on, whilst emphasis was never placed on the relation of recent disasters of software development such as ‘the millennium bug’, or the space rockets and railway disasters.

The previous are examples of large software systems infrastructure that we have inherited and there is an urgent need to re-engineer, test and improve them. These and similar incidents were due to software design errors and their study and examination revealed that there is urgent need for these to be addressed at method engineering level and from the metamodelling (and possibly metametamodelling) point of view.

In the field of method metamodelling most methods’ metamodels are led by rules of modelling abstraction which concentrate on soft or hard issues for the evaluation of methods, quantitatively or qualitatively. However, many of implied goals and objectives are not addressed and remain in the wish lists, whilst it seems that many process-oriented problems in software development remain undefined or not specified at all, to say the least. Some others such as the problem of testing remain in silence.

"*There is no single metamodelling initiative or CASE-Shell tool, which claims to facilitate testing*" [Berki, 2001].

In general, the use of formal methods for developing software-based systems did not lead to quality information systems, in the past. It soon became clear that analysing and designing a system with formal methods offers some quality assurance regarding the development of unambiguous, consistent, correct and verified mathematically-proven specifications, but there were other issues raised. The most frequently mentioned problem that is associated with the use of most formal methods in software development is the unfriendly and fragmented approach, which prevents wide understanding and results in high costs for training and prototype construction and testing.

Huotari & Kaipala summarised results of scientific work within the field of Human-Computer Interaction (HCI) focusing on cognitive aspects on methods' use. Their review analyses and synthesises the main contributions and takes a critical view on how cognitive aspects are considered and what methods are used. They believe that *"Despite a trend of applying cognitive task analysis and other user-centered system design methods, issues of human cognition and human information processing still need more attention in the IS research."* [Huotari & Kaipala, 1999].

2.2 The Need for a Multidisciplinary Approach in IS Development

Exposing the issues that were examined in the previous paragraphs and having research evidence on the cultural factors that influence software development (Ref. Kerstin's PhD) we need to consider the following: The *capturing of the needs' interconnected nature in a holistic question*, which will connect them to the needs of IS. Such a question will draw examples and provide links, opinions and insights from various related contexts and contents, in order to finally present an integrated solution [Berki *et al.*, 2003a].

That being the reality, a rigorous and integrated method based on holistic communication rules must also provide appropriate syntactic and diagrammatic structures to model the semantics, pragmatics and semiotics of systems' and stakeholders' cultural requirements and thus provide the scientific ground for *usability engineering* [Berki *et al.,* 2003b].

Therefore, the interest for IS designers should be in identifying and using general and understandable, groupware-oriented structures that capture adequately the features of specification and computation. This ca be achieved in terms of specialised and sufficiently general design structures that can capture the richness and testedness of domain specifications, considering at the same time people's cognitive needs for maximum participation and, therefore, *empowerment*.

It is important for the software developer to state clearly their objectives for software product or the process improvement, and to specify the product/process response characteristics that reflect these objectives. The formulation of the problem as well as the production of a list of controlled parameters and noise variables can be achieved through brainstorming and formulated using techniques such as the Ishikawa (cause and effect) or fishbone diagram.

Measurable objectives should be chosen such as the number of bugs found during formal inspections which are conducted during the software life cycle under the specified methodology a company adopts.

2.3 Sensitisation towards human factors

Throughout its lifetime, Information Systems Engineering has failed to deliver software products at satisfactory levels of reliability, timeliness and cost. To solve this problem, the Information Systems industry has turned its attention to the software development process, arguing, like other industries before it, that a high-quality process will deliver high-quality products. Thus various Software Quality

Management Systems have been proposed and developed. Although all these involved technological elements (which have generally received much attention), most people now recognise that the successful development of a quality culture depends more heavily on social factors. Yet these social factors have been largely neglected. However, in recent years a shift has taken place in software development by taking more human factors into consideration. Cornford and Smithson [1996] for example argue that we must see beyond any technology if we are ever to understand what happens when Information Systems are built or operate. They emphasise that people, people-structures and people-processes have to be taken into consideration when adopting an Information System's perspective both in development and research. A similar approach is emphasised by Klein and Lyytinen [1985], who state that one of the reasons for an Information Systems' failure is that the design conflicts with prevailing organisational culture and attitudes.

One of the human factors is commitment, which has long been argued to play a major role in the success of software projects since it increases the odds that appropriate actions will be taken [Ginzberg, 1981; Kautz, 1999; Lucas 1981; Marcus 1981]. Most of the researchers in the field seem to agree that commitment is one of the most important human factors in determining whether a well-planned process improvement program will succeed or fail [Dahlberg and Järvinen, 1997; Diaz and Sligo, 1997; Grady, 1997; Humphrey, 1995; Rodenbach et al., 2000; Stelzer and Mellis, 1998; Zahran, 1998].

Another human factor is culture. The main objective of the study described above [Siakas, 2002] was the identification of a number of cultural factors that have a bearing on the successful adoption and implementation of a Software Quality Management system. The main contribution of the study was the development of the SQM-CODE (Software Quality Management: Cultural and Organisational Diversity Evaluation) model, which assesses the fit between national and organisational culture.

3 Case Study

A comparative study was carried out in the form of quantitative and qualitative investigation in four counties, namely Denmark, Finland, Greece and the UK. The quantitative investigation was a survey collecting hard data by using a postal questionnaire. In parallel, a qualitative method in form of case studies was performed in order to address different aspects of the research problem, to confirm the findings from the questionnaire and to prove the hypothesis.

The questionnaire was sent to organisations developing software for own use or for sale. Totally 307 questionnaires were completed. In addition field-studies were undertaken in several organisations. Totally 87 interviews were conducted in Finland, Denmark and Greece with software developers at different levels and with different positions in the organisations. Following the initial verification phase observations were carried out in a Danish organisation for a period of two month [Siakas and Balstrup, 2000]. The objective of using observations was to investigate in more depth the research problem and to verify the findings.

3.1 Are measures of the quality of software process kept?

From figure 1 we observe that amongst the organisations taking part in the study Greece is the country that keeps measures of the quality of the software development process to highest degree. The sum of the values for quite a lot or very much so is 61.9% for Greece, 44.7% for Denmark, 42.6% for Finland and 42.5% for the UK.

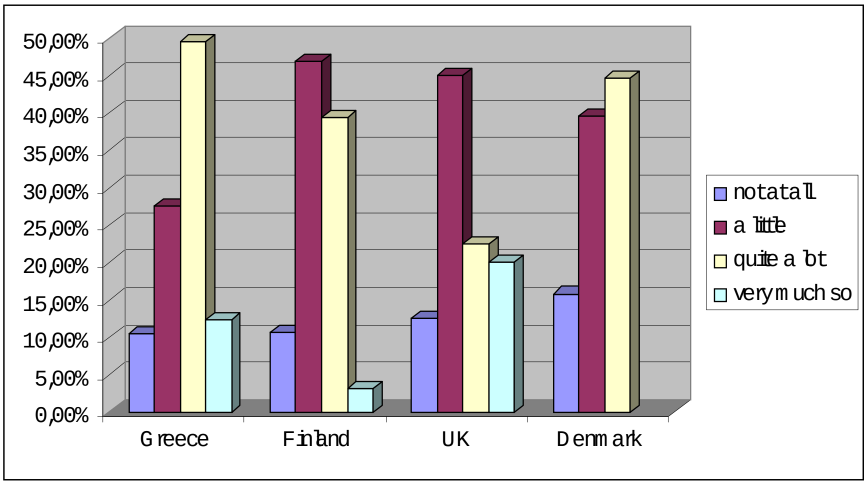


Figure 1: Measures of the quality of the software process

The significance of the Chi-square is 0.002, which indicates that the null-hypothesis, that the responses are similar for all countries, can be rejected. This means that we have statistically proved that there are significant differences in responses depending on country of origin.

3.2 Are measures of the software product kept?

From figure 2 we observe that amongst the organisations taking part in the study Greece is the country that keeps measures of the quality of the software product to highest degree. The sum of the values for quite a lot or very much so is 67.3% for Greece, 50.5% for Finland, 50% for Denmark and 47.7% for the UK.

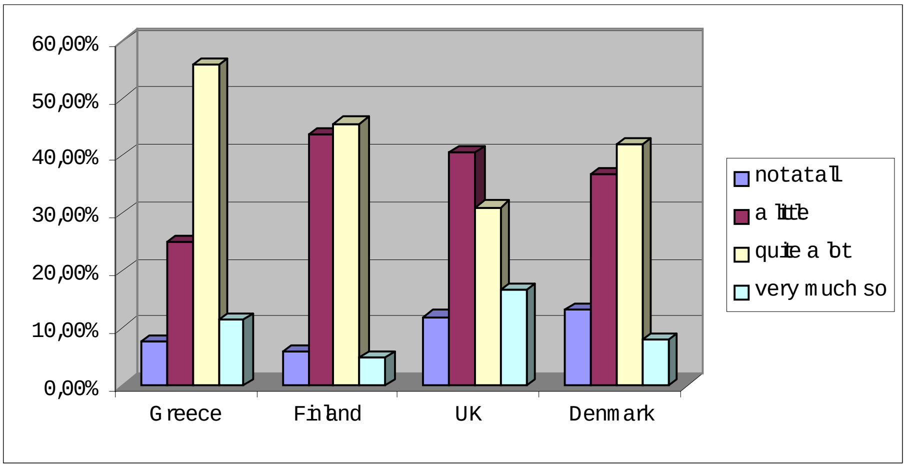


Figure 2: Measures of the quality of the software product

The significance of the Chi-square is 0.042, which indicates that the null-hypothesis, that the responses are similar for all countries, can be rejected. This means that we have statistically proved that there are significant differences in responses depending on country of origin.

In order to understand if there is a relationship between measurement of software process and measurement of software product table 1 is presented.

	Measure of quality of software product					Total
Measure of quality of software development process		not at all	a little	quite a lot	very much so	
	not at all	21	11			32
	a little	2	71	29	4	106
	quite a lot		14	93	7	114
	very much so		3	6	15	24
Total		23	99	128	26	276

Table 1: Cross-tabulation of measures of the quality of the software process vs. product

From table 1 we observe that measures the quality of the software process also tend to measure the quality of the software product. There is also a significant correlation on

The Pearson's correlation is also significant at the 0.01, which proves that there is a significant relationship between measurement of the quality of the software development process and the measurement of the quality of the software product.

3.3 What is the degree of impact of international, national or organisation specific standards and systems on measurement?

Figures 3 to 6 present the impact on the degree of measurement by quality systems, like ISO 9001 and TickIT, as well as in-house standards have any use figures 3 to 6 are presented.

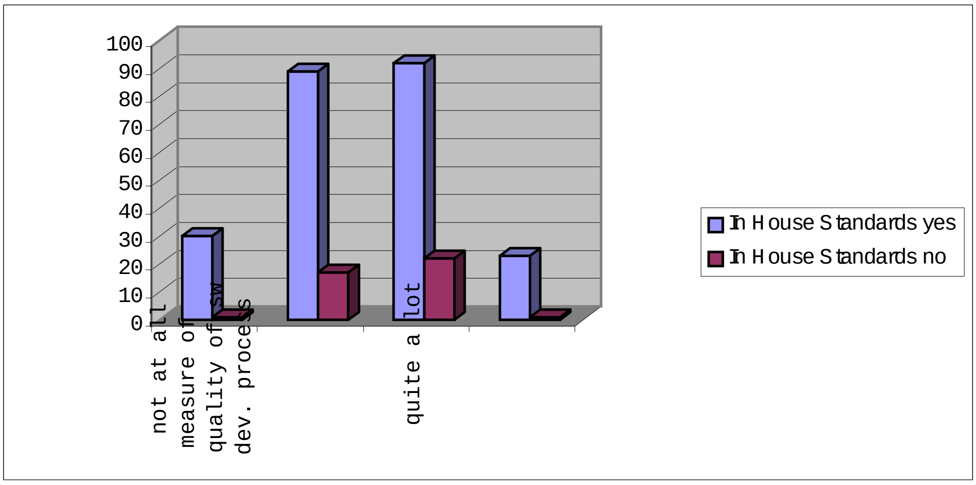


Figure 3: Measurement of quality of software development process vs. in-house standards

From figure 3, which is a cross-tabulation of measurement of the quality of the software development process and the use of in-house standards we observe that the majority of the organisations, which have in-house standards measure the quality of the software development process a little or quite a lot. The surprising element is that 30 (of 275) respondents which responded that they have an in-house quality system also answered they do not measure the quality of the software development process at all. The significance of the Chi-square is 0.061.

From figure 4, which is a cross-tabulation of measurement of software development process and third party assessment we observe that most of the organisations do not have any third assessment at all. However this does not seem to have any greater impact on if they measure the quality of the software development process or not. The significance of the Chi-square is 0.180.

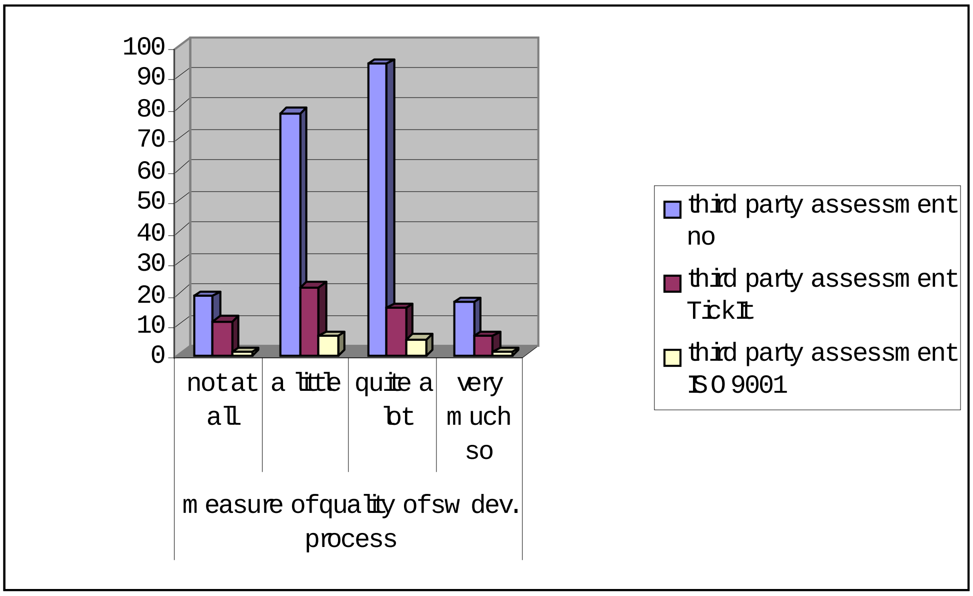


Figure 4: Measurement of quality of software development process vs. third party assessment

From figure 5, which is a cross-tabulation of measurement of the software product and in-house standards we observe similar results as with figure 3. The significance of the Chi-square is 0.012

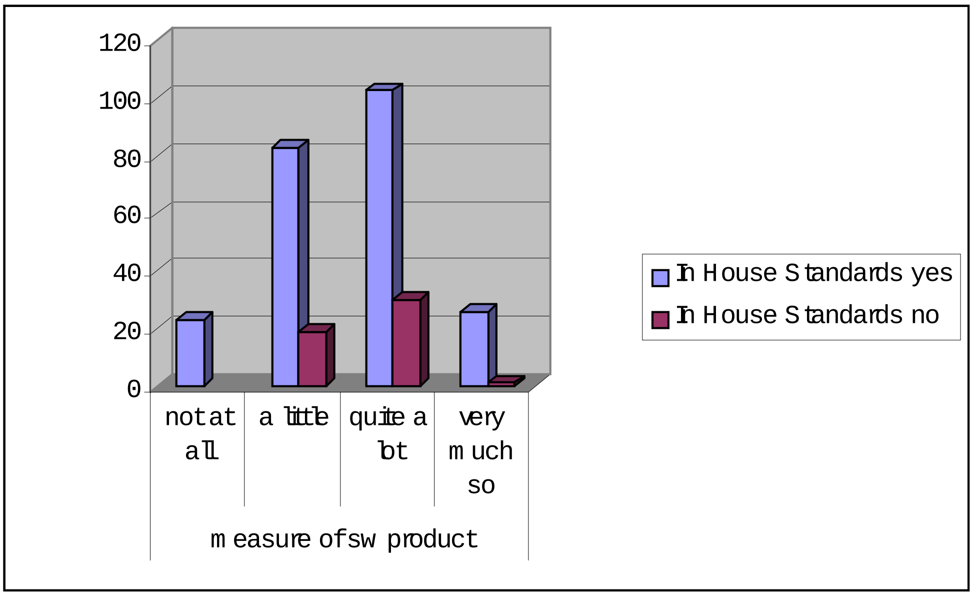


Figure 5: Measurement of quality of software product vs. In-house standards

From figure 6, which is a cross-tabulation of measurement of software product we observe similar results with figure 4. The significance of the Chi-square is 0.254.

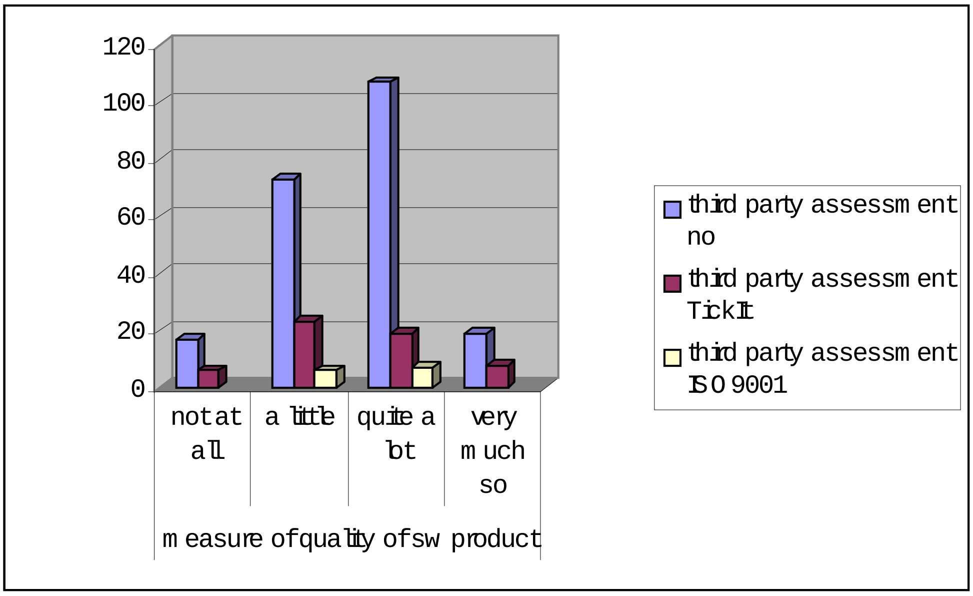


Figure 6: Measurement of quality of software product vs. Third party assessment

4 Software Measurement and Software Metrics

"To measure is to know. If you can not measure it, you can not improve it."(William Thomson (later Lord Kelvin) (1824 - 1907).

Managing the development process requires the collection of suitable metrics which will provide insights into the strengths and weaknesses of the process. What to measure, how to measure, when to measure are the fundamental questions which need to be addressed.

According to Kitchenham [1996] "Software Metrics can deliver :support for process improvement, better project and quality control and improved software estimation". *Direct measurement* of quality factors is often possible very late in the life cycle. For example *reliability*, which is concerned with how well as software system functions and meets a user's requirements, can be measured after that the software has been used for a stated period of time under stated conditions, while *indirect measurement* of quality, like number of discrepancy reports (deviations from requirements) can be obtained earlier in the life cycle. Other estimates of quality can be made by developers even earlier than the indirect measurements of quality.

According to ISO-9126 software quality may be evaluated by six characteristics, namely functionality, reliability, efficiency, usability, maintainability and portability. Each of these characteristics is defined as a "set of attributes that bear on" the relevant aspect of software and can be refined through multiple levels of sub-characteristics[6]. Definitions of sub-characteristics are given in Annex A (of the standard), which is not a part of the International Standard. Attributes at the second level of refinement are left completely undefined.

In order to achieve improvements in the software process and the software product we need to understand, measure and hence control the variability to the desirable/achievable degree of confidence. Measurement is defined as the process of assigning symbols, usually numbers, to *represent* an *attribute* of the entity of interest, by *rule* [Fenton 1991], [Shepperd 1995]. Entities of interest include objects, (e.g. code, specification, person) or processes (e.g. analysis, error identification, testing). Distinct attributes might be length of code, duration, costs. Representation is usually in numbers (or other mathematical objects e.g. vectors). Finally in order to provide objectivity we need to assign numbers (symbols) according to explicit rules: how to choose which symbol should represent the attribute. Such rules ensure that the assignment is not random.

Fenton and Pfleeger [Fenton 1997] provide a refined definition of measurement: "Measurement is the process by which numbers or symbols are assigned to attributes of entities in the real world in such a way as to characterise them according to clearly defined rules. The numeral assignment is called the measure." Hence, in order to understand the definition of measurement in the software context, we need to identify the relevant entities and attributes which we are interested in characterising numerically.

4.1 Controllable Factors

In order to understand and control the process we need measurements of both the current and the desired/new system. Internal metrics [Fenton, 1991] an be obtained in terms of the product (code) and they are counts (such as LOC, NO of Classes, McCabe Complexity) and ratios (such as No of calls Module, Average length of hierarchy). Additionally, these metrics can be generated automatically by using tools such as CANTATA, Testbed and Logiscope.

Attributes such as the morphology, architectural structure, depth of class hierarchy, size of module, maximum level of module complexity etc. can be contolled through a management mechanism and specific guidelines to the developers. *Controllable* design parameters can be found in the software development process, the software product and the software development environment [Kitchenham, Fenton, Barbor & Georgiadou].

However, external attributes [Fenton 1991], [Kitchenham, 1996], [Georgiadou 1999, 2001] which are behavioural such as understandability, maintainability are more elusive and more difficult to measure. Metrics for these attributes are both qualitative and quantitative. They are almost always obtained indirectly through the use of surrogate measures [Kitchenham, 1996], [Georgiadou 1993, 2001]. For example maintainability can be estimated, calculated and controlled through measuring the time taken for a specified maintenance task. Results obtained by Georgiadou et.al in a series of controlled experiments provided confidence (through statistical methods) in our ability to effectively use surrogate metrics [Georgiadou, 93, 94, 97, 98, 2001].

4.2 Uncontrollable Factors

Human factors are unpredictable and mostly difficult, often impossible to control. For example, performance variability in a human being, such as his/her experience and communication skills needed in a software development team. The developers' performance has an effect on producing quality software products in a similar way to the effect of machines on the manufacturing of products. It is important to maximize and properly maintain programmers' performance. The possible control factors will be conducting educational sessions within and outside a company where software developers are encouraged to learn the new techniques of their interest or polish their skills. Recreational events may help developers to get to know each other better and this will be reflected in better communication and teamwork in an office. At the extreme, the design of the office environment itself is investigated. Changing the type of chairs in current use to the ones designed to ease backpain caused by sitting all day will be welcomed by the developers (ergonomics). The temperature and humidity in the workplace also can affect the developers' performance. Therefore the suggestions made here must be investigated in the software industry.

Experimental evaluations carried out by Basili [1986]), Shepperd [1995], Georgiadou [1999, 2001] attempted to identify design parameters and hence factors, which can be controlled. According to

Taguchi in [Logothetis 1989] it is desirable to choose the set of design parameters, which are less affected by the variability of these factors. For example, developers' experience can be controlled to certain extent by years in profession and looking at the past projects involved. However every individual is unique. His/her capability, patterns of learning and cognition are likely to be different from those of others of similar experience. The health of the developers may effect on their performance at work.

5 Summary and Future Research and Development

The fundamental philosophy of this research work is that it addressed the needs for adequate expression of process models within various cultural factors and different organisational communication. In doing so, we interconnected and commented on their dynamic and computational characteristics connecting them to the coverage of testing and re-engineering.

Our multidisciplinary study reveals the strengths that the *holistic* nature of such an approach provides software development with the use of software measurement as the instrument for understanding, estimating and controlling the quality of specified factors. Bearing in mind that different stakeholders place different emphasis on software attributes we provide flexible and hence customisable quality requirements.

The efficiency of a software product such as execution time has high priority as a software quality factor. To maximize the performance of a product, the choice of machines, operating systems and programming language has to be included in the list of parameters.

Enhanced Reliability (usually achieved through testing, walkthroughs, reviews and inspections) will reduce productivity and will therefore increase costs. Both of these cause losses to the sponsor. Enhanced functionality increases costs (in the short term) and causes losses to the sponsor.

Usability is enhanced through greater understandability, which in turn is enhanced through design correctness and consistency and through training, on-line help and support all of which reduce productivity with the exception of CBD which makes extensive reuse of code and increasingly reuse of designs too.

It is important for the software developer to state clearly their objectives for software product or the process improvement, and to specify the product/process response characteristics that reflect these objectives. The formulation of the problem as well as the production of a list of controlled parameters and noise variables can be achieved through brainstorming and formulated using techniques such as the Ishikawa (cause and effect) or fishbone diagram.

Measurable and hence controllable objectives should be chosen such as the number of bugs found during formal inspections, which are conducted during the software life cycle under the specified methodology a company adopts.

Future investigations will concentrate on the development of meta-CASE tools for the specification, implementation, re-engineering and metrication of both the process and the product of software development across problem domains, national, cultural and technical boundaries.

6 Literature

Barbor, N & Georgiadou, E. [2002] Investigating the applicability of the Taguchi Method to Software Development, Proceedings of Quality Week, San Francisco. USA, July 2002

Basili, V., R. W. Selby, & D. H. Hutchins[1986] Experimentation in Software Engineering. IEEE Trans. on Software Engineering, SE-12. p. 733-743.

Berki, E., Georgiadou, E., Holcombe, M.[2003a]: "Process Metamodelling and Method Engineering as Tools for Improved Software Quality Management - *A Chronological Review and Evaluation Critique Considering the Need for a New Scientific Discipline*",In the Proc. of Ross, M., Staples, G. (Eds.) 11th International Conference on Software Quality Management, SQM 2003, Glasgow, April 2003.

Berki, E., Isomäki, H., Jäkälä, M.:[2003b] "Holistic Communication Modelling: Enhancing Human-Centred Design through Empowerment", in the Proc. of the HCI International Conference, University of Crete, Greece, June 2003

Berki, E., Lyytinen, K., Georgiadou, E., Holcombe, M., Yip, J.:[2002] "Testing, Evolution and Implementation Issues in MetaCASE and Computer Assisted Method Engineering (CAME) Environments", In the Proc. of King, G., Ross, M., Staples, G. & Twomey, T. (Eds.) Issues of Quality Management and Process Improvement, 10th International Conference on Software Quality Management, SQM 2002, Limerick, Ireland, March 2002.

Berki, E.: [2001]"Establishing a Scientific Discipline for Capturing the Entropy of Systems Process Models, CDM-FILTERS: A *Computational* and *Dynamic Metamodel* as a *Flexible* and *Integrated Language* for the *Testing, Expression* and *Re-engineering* of *Systems",* Ph.D. Thesis, Faculty of Science, Computing & Engineering, University of North London, 2001.

Burr A., Georgiadou E.[1995] *"Software development maturity - a comparison with other industries",* 5th. World Congress on Total Quality, India, New Delhi, Feb. 1995

Cornford Tony, Smithson Steve [1996]: *Project Research in Information Systems*, A Student's Guide, MacMillan Press Ltd, London

Dahlberg T., Järvinen J. [1997]: Challenges to IS Quality, *Information and Software Technology Journal*, 39 (12): 809-818

Diaz M., Sligo J. [1997]: How Software Process Improvement helped Motorola, *IEEE Software,* 1997, 14, 75-81

Dorling Alec [1993]: Spice: Software Process Improvement and Capability dEtermination, *Software Quality Journal*, 2, 93, pp. 209-224

DTI [1992]: *TickIT making a better job of software, Guide to Software Quality Management System Construction and Certification using ISO9001/EN29001/BS5750* Part 1 (1987): 29 February 1992, Issue 2.0

Fenton, N.E. and Pfleeger, S.L.[1997] 'Rigorous & Practical Approach', PWS Publishing Company

Georgiadou E., Sadler C.[1995] "*Achieving quality improvement through understanding and evaluating Information Systems Development Methodologies*", 3rd. International Conference on Software Quality Management, SQM'95, Seville, Spain

Georgiadou, E. [2003b] "Software Process and Product Improvement A Historical Perspective", International Journal of Cybernetics, January 2003

Georgiadou, E. [2003b] Towards a Customisable, Multi-layered Software Engineering Quality Model, 11th International Conference on Software Quality Management, April 2003, Dundee, Scotland

Georgiadou, E., Keramopoulos, E. [2001a]*"Measuring the Understandability of a Graphical Query Language through a Controlled Experiment*", 9th International Conference on Software Quality Management, SQM 2001, April 2001, University of Loughborough, UK.

Georgiadou, E., Milankovic-Atkinson, M. [1999] *"A formal experiment to verify Object-Oriented Metrics",* *I*NSPIRE'99, Crete, Greece

Georgiadou, E.[2001b] Software Measurement for Process and Product Improvement - Controlled Experiments and Derivation of Reengineering Metrics, Transfer Report (from M.Phil. to PhD.), chool of Informatics and

Multimedia Technology, Faculty of Science Computing and Engineering, University of North London, August 2001

Ginzberg M.J. [1981]: Key Recurrent Issues in the MIS Implementation Process, *MIS Quarterly* 5: 47-59

Grady R. B. [1997]: *Successful Software Process Improvement*, Prentice Hall PTR, Upper Saddle River, NJ

Haase Volkmar H. [1992]: Bootstrap - Measuring Software Management Capabilities, First Findings in Europe, Proceedings of *the fourth IFAC/IFIP Workshop*, Austria, May 92

Haase Volkmar, Messnarz Richard [1994]: Bootstrap: Fine-Tuning Process Assessment IEEE Software, July 1994, pp. 25-35, 1994

Herbsleb James, Carleton Anita, Rozum James, Siegel Jane, Zubrow David [1994]: Benefits of CMM-Based Software Process Improvement: Initial Results, Technical Report, CMU/SEI-94-TR-13, August 1994

Holcombe, M. and Ipate, F.:[1998] [2001] "Correct Systems - Building a Business Process Solution", Springer-Verlag, 1998

Holcombe, M., Bogdanov, K., Gheorghe, M.: “Functional test set generation for extreme programming”, In the Proc. of the 2nd International Conference on Extreme Programming and Flexible Processes in Software Engineering [XP2001], pp. 109-113, Sardinia, Italy, 20-23 May, 2001

Humphrey W. [1995]: *A Discipline for Software Engineering*, Addison Wesley, Reading, MA, USA

Huotari J. & Kaipala, J.:[1999] “Review of HCI Research - Focus on cognitive aspects and used methods”, in the Proc. of Kakola T. [ed.] IRIS 22 Conference: Enterprise Architectures for Virtual Organisations, Keurusselka, Jyvaskyla University Printing House, Finland, 1999

Jackson, M. and Zave, P.:[1993] "Domain descriptions", in Proceedings of the 1st International Symposium on Requirements Engineering, San Diego, Ca, pp. 56-64, 1993

Järvinen J. [1994]: On comparing process assessment results: BOOTSTRAP and CMM, *Software Quality Management, SQM94*, Edinburgh, pp. 247-261

Kautz K. [1999]: Software Process Improvement in Very Small Enterprises: Does It Pay Off? *Software Process – Improvement and Practices,* 4 [4]: , 209-226

Kitchenham, Barbara, and Shari Lawrence Pfleeger.[1996] "Software Quality: The Elusive Target." *IEEE Software* 13, 1, Jan. 1996: 12-21

Klein H, K., Lyytinen K. [1985]: The Poverty of Scientism in Information Systems. In Mumford E. Hirschheim R. Fitzgerald G, Wood-Harper T. [eds.]: *Research Methods in Information Systems,* Proceedings of IFIP WG 8.2 Colloquium, Manchester Business School, 1-3 Sept. 1984, pp. 131-161, North-Holland, Amsterdam

Kuvaja P., Similä J., Kranik L., Bicego A., Saukkonen S., Koch G. [1994]: *Software Process Assessment and Improvement – The BOOTSTRAP Approach*, Blackwell Publishers, Cambridge, MA

Kuvaja Pasi [1999]: New Developments in Software Process Improvement Keynote Speech in *Software Quality Management Conference* [SQM 99]: Southampton March 1999

Kuvaja, 1999] and SPICE [Dorling, 1993; Rout, 1995] concentrate on the process instead of the product by ensuring a disciplined and controlled software development process via independent evaluation.

Logothetis, N. andWynn, H.P., [1989] ‘Quality Through Design: Experimental Design, ‘Off-line Quality Control and Taguchi’s Contributions’, Oxford Science Publications, 1989.

Lucas H.C. Jr. [11981]: *Implementation: The Key to Successful Information* Systems, Columbia University Press, New York

Marcus M.L. [1981]: Implementation Politics: Top Management Support and User Involvement, *Systems, Objectives, Solutions* 1: 203-215

Paulk Mark C. [1993]: Comparing ISO 9001 and Capability Maturity Model for Software, Software Quality Journal 2, 1993, pp. 245 - 256

Paulk Mark C. [1995]: The Rational Planning of [Software]: Projects, *Proceedings of the First World Congress for Software Quality, ASQC,* San Francisco, CA, 20-22 June 1995

Paulk Mark C., Curtis Bill, Chrissis Mary Beth [1993]: Capability Maturity Model, Version 1.1, *IEEE Software*, July 1993, pp. 19-27

Rodenbach E., Van Latum F., Van Solingen R. [2000]: SPI – A Guarantee for Success? – A Reality Story from Industry, *PROFES 2000*, Oulu, Finland, Springer, 216-231

Rout Terence P. [1995]: SPICE: A Framework for Software Process Assessment, *Software Process-Improvement and Practice*, Pilot Issue, pp. 57 – 66

Shepperd, M.J.[1995] *Foundations of Software Measurement* Prentice-Hall: Hemel Hempstead, England

Siakas Kerstin [2002]: SQM-CODES; Software Quality Management – Cultural and Organisational Diversity Evaluation, PhD thesis submitted Nov. 2002 for the degree of PhD at London Metropolitan University

Siakas Kerstin V. and Balstrup Bo [2000]: A field-study of Cultural Influences on Software Process Improvement in a Global Organisation*, European Software Process Improvement Conference, EuroSPI '00*, Copenhagen 7-9 Nov. 2000

Stelzer D., Mellis W. [1998]: Success Factors of Organisational Change in Software Process Improvement, *Software Process – Improvement and Practice*, 4, 227-250

Zahran S. [1998]: *Software Process Improvement: Practical Guidelines for Business Success,* Addison-Wesley Pub. Co. Reading, Mass.

7 Author CVs

Elli Georgiadou

Elli Georgiadou is a Principal Lecturer in Software Engineering at Middlesex University, London. Her teaching includes Software Metrics, Methodologies, CASE and Project Management. She is engaged in research in Software Measurement for Product and Process Improvement, Methodologies, Metamodelling, Cultural Issues and Software Quality Management. She is a member of the University's Global Campus project (developing and offering ODL). She has extensive experience in academia and industry, and has been active in organising/chairing conferences and workshops under the auspices of the British Computer Society, the ACM British Chapter and various European programmes for Technology Transfer and development of joint curricula. She established a Distance Mode Initiative between a UK University and a Hong Kong Institute developing and offering technology-based learning. She has engaged in developing the pedagogic framework as well as the development of materials. She designed and carried out evaluations of various ODL initiatives in the UK, Greece, Spain, Finland, Hong Kong and Cyprus.

Kerstin V. Siakas

Kerstin works as a lecturer since 1989 at the department of Informatics at the Technological Education Institution of Thessaloniki, Greece. She is born and grown up in the Swedish part of Finland. She has an extensive industrial experience (since 1975) in software development on different levels from many European countries and mainly from multinational organisations. Because of her multicultural background and her work experience her research interest is in Software Process Improvement and how culture influences software development. She finished her PhD in November 2002. She has published around 20 papers about her research.

Eleni Berki

Eleni Berki is an Asst. Professor of Group Technologies in Jyväskylä University, Finland. She completed her PhD in Process Metamodelling and Systems Method Engineering in 2001, in United Kingdom. Her teaching and research interests include: Process Metamodelling and Information Systems Engineering, Computational Models and Multidisciplinary Approaches for Software Engineering, Knowledge Representation Frameworks and Requirements Engineering. She has worked as a software designer and consultant in industry, and has a number of academic and industrial project partners in many countries. She has been active in the development, delivery and coordination of virtual and distance learning initiatives in collaboration projects in European and in Asian countries.

Managing for Change when Implementing Software Process Improvement Initiatives

Ita Richardson & Timo Varkoi

Abstract

This paper presents software process improvement research carried out in small software development companies in Ireland and Finland. These software process initiatives have demonstrated alignments between the change experienced in companies studied and Willman's change management model as presented in the business discipline. This model is presented in the paper followed by actual examples in our software process improvement research, which verifies the model.

Keywords

Change management, software process improvement, small software development companies, factors for change.

1. Background

1.1 Finnish Software Economy

According to OECD statistics, the size of information and communication technologies (ICT) sector is largest in Ireland and Finland (OECD 2002). The total turnover of the ICT sector in Finland is 8.5 billion euros i.e. 6.2 % of GNP and the annual growth rate is about 8 %. The ICT sector employs over 112 000 people of which in software consultancy and supply over 21 000. There is close to 3000 software businesses in Finland of which vast majority are small and medium sized enterprises (SMEs).

1.2 Irish Software Economy

Software is a major success story in Irish industrial development. About 840 software development companies, local and foreign-owned, employ some 30,000 people and account for 8% of GDP (IDA 2001). A recent industry report (ICT Ireland 2003) predicts that indigenous employment in the broader ICT sector could double to 60,000 if the right policies, including research support, are implemented. Ireland is one of the leading exporters of software and has a leading position in some sectors such as telecommunications networks, financial systems and customer support.

1.3 Software Process Initiatives

A project to establish a software process improvement (SPI) network in the Satakunta region in Western Finland was started in August 1998 and ended December 2000. The established network is called SataSPIN and today it continues and extends the SPI related co-operation between software organizations in the region. The SataSPIN project helped SMEs in software business to develop their operations using international software process models. The project received substantial funding from the European Social Fund. (Varkoi 2002)

More than twenty small software organizations have participated in SataSPIN to improve their software processes, several more have participated in the associated training. The main source for software process assessment and improvement models in SataSPIN is ISO/IEC 15504 TR (SPICE). The work with participating companies resulted in SPI Initiation Framework (Mäkinen et al. 2000), which covers the process improvement start-up from the examination of an organizations needs, through assessments to the development and support of improvement actions. The SataSPIN companies have typically from 2 to 50 employees in software engineering related positions. Typically the personnel in the companies are professionally experienced and highly motivated in their work. A detailed description of SataSPIN is in (Varkoi & Mäkinen 1999).

During the same period in the mid-west region of Ireland, research was carried out in small companies. (Richardson et al. 2001, Richardson 2002, Richardson & Ryan 2001). The primary purpose of this research was the implementation of a software process matrix through action research with control groups in four small indigenous software development companies. A company summary is given in Figure 1. Three of the four companies demonstrated improved process and relevant findings are presented in this paper.

Company	Total Employees	S/W Dev. Employees	SPI / Quality Initiatives
Computer Craft	16	4	Implemented SPI initiative as Action Research project
DataNet	9	4	Implemented SPI initiative as Action Research project, also working towards ISO9001 certification
Software Solutions	50	10	Working towards ISO9001.
Ríomhaire	35	3	ISO9001 certified

Figure 1: Summary of Companies Researched

1.4 Why Software Process Improvement?

Humphrey (1989) defines a software process as “the set of tools, methods and practices we use to produce a software product”. Paulk et al. (1993a) expand this definition to "a set of activities, methods, practices and transformations that people use to develop and maintain software and the associated products". The process is often documented in a set of guidelines and/or procedures which are followed during software development. However, this documentation is not the process – it should reflect the process and in doing so must be actively used and followed by software developers.

Internationally, as an aid to the improvement of software quality, there is a growing interest in the improvement of software process. This initiative, if ignored, may contribute to a reduction in the advantage which Irish and Finnish companies have had in the international marketplace in recent times. Therefore, it is one which the small indigenous software sector not only should be aware of, but must become involved in. The views of Combelles and DeMarco (1998) should not be taken lightly: “the issues Software Process Improvement raises are important, if not vital, to the future of your organisations”.

Because there has been a significant move towards improving software process in recent years, this has, at times, been to the detriment of concentration on other software product improvement techniques. Humphrey (1989) did not present his software process model as a “cure-all”, in fact, he states that “two other key areas that need to be considered are people and design methods”. During this research, the authors have tried to remain conscious of this. This research is based on software process, one dimension of software quality, and while software process improvement strategies can bring improvements to organisations, it is not the only means by which this can be done.

Biro and Messnarz (1999) point out that people in different roles see process improvement differently. Business managers might not be willing to invest in SPI due to its long pay-back time, which shows the lack of commitment. Middle managers are most likely to support process improvement, because it provides them better understanding of the process and means to control it. Practitioners usually are doubtful in the beginning of SPI and see it as another way to control their performance.

So, why this emphasis on software process? The emphasis is there because companies expect that an improved software process will result in improved quality of the software product. However, “reuse, reverse engineering, rapid prototyping, the object-oriented paradigm, total quality management, and finally, process improvement each were touted, in turn, to be the solution to software problems” (Johnson and Brodman, 1992). Is improving software process just another ‘touted’ solution, or is it valid for us to expect that improving the software process will indeed improve the quality of the product?

"The quality of a software product is largely determined by the quality of the software development and maintenance processes used to build it". (Paulk, 1995a). Therefore, software process improvements should result in an improvement in the software product: improved product quality, reduced cost and / or faster time-to-market. Gillies (1992) agrees that "the link between the quality of the development process and the final product is well established", while Paulk (1996) states that “the true value of software process improvement lies in the impact on quality, productivity, predictability, and time to market of software products”. Zahran (1998) states that “it is a widely accepted fact that the quality of a software product is largely determined by the quality of the process used to develop and maintain it". In the IBM Survey Results (1996), where over 360 European organisations were surveyed, one of the outcomes was that "not one of the organisations that have good practices is performing badly. Equally, none of the organisations that have poor practices is performing well". Performance was measured on development productivity, removal of defects before delivery and accurate estimation of cost and duration of projects. In Humphrey (1998), 13 organisations that worked on software process for three and a half-years all gained "substantial cost, schedule, and quality benefits". In a study carried out by Brodman and Johnson (1995), results from twenty questionnaires distributed to American software companies are presented. These companies saw a range of tangible benefits from their software process improvement efforts including:

- Increase in Productivity
- Reduction in Defects
- Reduction in Error Rates
- Decrease in Costs
- Increase in On-time Deliverables and Project Completion
- Reduction in Rework
- Savings in Test time.

Strader et al. (1995) link the "high quality software" produced by the Space Shuttle Onboard Software project to the high maturity level of their software process. All of the aforementioned have been able to improve their product by improving the quality, decreasing the time-to-market and / or reducing the cost. Management in Irish software companies also support this view. The managing director of Blackbird Data Systems commented to the author:

> *"Bad quality and bad installations resulted in sales representatives avoiding the customers. We had irate customers. We had to support a product and could not charge for it. And we were spending more money trying to get the next contract. Now we have good quality, happy customers, and ** per annum per installation coming in. This has been because we improved our process".*

These results are what would be expected when good quality product is produced: "It is much more expensive to gain a new customer than to keep one who is satisfied, or, even easier, delighted with our product" (Bergman and Klefsjo, 1994).

There are dissenting voices from this view. Fenton et al. (1995) state that "there is, as yet, no convincing evidence that higher rated companies produce better quality software". The difficulty expressed by Jones (1996) is that there is not a large amount of empirical data available. Humphrey (1998a) gives results from 13 organisations; Brodman and Johnson (1995) have results from twenty, most other cases are individual case studies, some quoted in multiple papers.

However, prior to and during this research, these authors have seen enough evidence of poor processes in companies, to be convinced that improvements to their processes could only cause improvements to the software product. It is likely that, when a company is not in 'chaos' and has reached a certain threshold in the formality of their processes, further improvements will not have a significant, if any, effect. This could occur when companies have a more mature software process.

The small software sector is a major driver for both the Irish and Finnish economies and, consequently, it is important for it to continue to grow in the global marketplace. To do this, it must keep up with the software sector worldwide. Increased interest in software quality, particularly in the United States of America, has caused many software companies to strive for an improved software process. Although not beyond dispute, it is generally accepted that software process improvement will improve product quality. It can also be shown that when an organisation improves process, time-to-market will be reduced. Consequently, cost will be reduced. This interest in software process is often being driven by larger organisations who sub-contract to smaller companies. This trend is likely to continue and grow.

2 Model for Change Management

Willman’s framework for change management (Willman, 1996) identifies five factors, which must exist before there is successful implementation of change, regardless of the functional area in which this change is taking place. If any one of these is missing, the outcome changes (see Figure 2).

These are:

- Pressure for change;
- Leadership and vision;
- Capable people;
- Actionable first steps;
- Effective rewards.

Pressure for change	+	Leadership and vision	+	Capable people	+	Actionable first steps	+	Effective rewards	= Successful implementation
		Leadership and vision	+	Capable people	+	Actionable first steps	+	Effective rewards	= Disinterest
Pressure for change	+	Leadership and vision	+	Capable people	+	Actionable first steps			= Evaporation
Pressure for change	+	Leadership and vision	+	Capable people			+	Effective rewards	= Frustration
Pressure for change	+	Leadership and vision			+	Actionable first steps	+	Effective rewards	= Disengagement
Pressure for change			+	Capable people	+	Actionable first steps	+	Effective rewards	= Disillusionment

Figure 2: Framework for Change Management

Three of these requirements for successful implementation of change, *leadership and vision, capable people* and *effective rewards* belong to the human resource discipline. While normally outside the scope of software assessment and improvement models, without these, any initiative to improve an organisation’s software process, regardless of the size of the company, is likely to fail. If we consider changing the software process, *pressure for change* would correspond to the reason for improving the software process. This may be because of external or internal influences. It is possible that management have decided that processes need to be improved for business purposes, that a customer requires a higher level of process maturity, or because the software engineers are unhappy with processes as they exist. The improvement strategy should also provide *actionable first steps* as identified by Willman. This strategy should list actions to be undertaken within the company to improve the software process.

3 SPI Initiation Framework

The SPI Initiation Framework, which was developed during the SataSPIN project, is an approach to SPI implementation in especially small software organisations. The framework supports process improvement establishment from analysis of an organization's needs to assessments and development and support of the SPI program. In this case, the framework is used to discover various change management issues related to SPI.

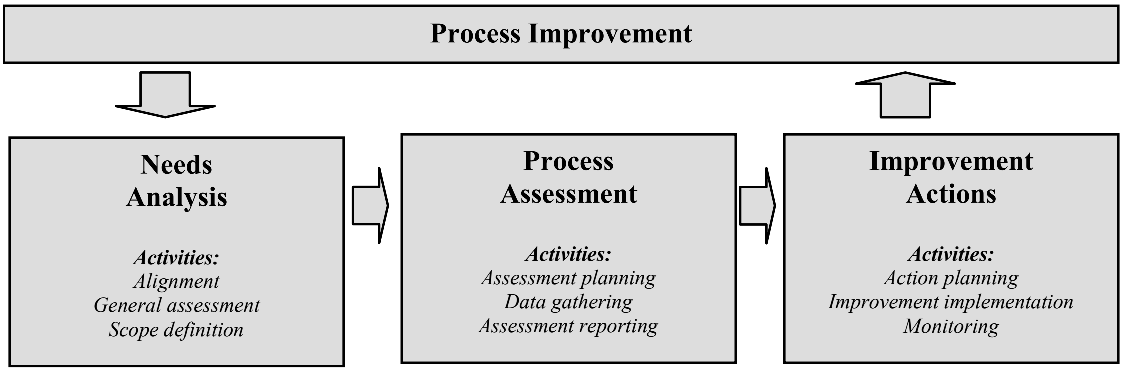


Figure 3: SPI Initiation Framework

The framework consists of three steps (Figure 3). First, the organization needs to understand the possibilities of SPI in achieving its business goals and to direct the improvement efforts. In the second step software processes are assessed in a reliable manner to find out the improvement ideas that will benefit the company. Next, appropriate improvement actions are planned and supported, and monitoring the results provides input for further improvements. Each step of the framework consists of activities to facilitate detailed planning and to ensure that the improvement work proceeds as desired.

The activities in the first step are alignment, general assessment and scope definition. Alignment activity provides common understanding within the organization of process thinking and related concepts. General assessment is used to study the overall situation of the organization's software processes. Organization's needs and present situation are considered and the scope for improvement is defined including the processes that will be assessed next. The outcome of the first step is a feasible software process development plan also known as a SPI program plan.

The second step concentrates on a planned, detailed assessment of the selected processes. Assessment planning determines timetables, participants, source material and restrictions to the assessment. Data gathering consists of studying the source material and interviewing the personnel during the assessment sessions. The outcome of the second step is an assessment report that records the findings, process ratings and profiles in detail.

Action planning, improvement implementation and monitoring are the activities of the third step. Improvement actions are prioritised, planned and scheduled, and resources for the work are appointed. Improvements are implemented and achievements are controlled to ensure desired results. Based on the monitoring of the processes and improvements new improvement ideas will come up.

Comparing with Willman's framework the factors *pressure for change* and *leadership and vision* would be emphasized in the first step, Needs Analysis. The factor *actionable first steps* can be applicable in two cases: first steps in SPI implementation or in improvement of the selected processes. The SPI Initiation Framework provides aid in the first case and analyzed assessment results supports the second case. Setting rules for *effective rewards* should be part of the Alignment activity and Monitoring activity should provide the basis for rewarding. *Capable people* are equally important in each step of the improvement work.

Willman's framework presents the complexity of change management, which is familiar in SPI improvement efforts, too. The change management factors could be used to enhance SPI models, for instance in risk management throughout the steps of the SPI Initiation Framework. Successful implementation is the underlying goal of each SPI program and consideration of Willman's factors would increase our chances to reach that goal.

4 Researched Companies

During our research with small software development companies, we noted alignments between Willman's model (from the business management discipline) and the software process improvement initiatives in which we were involved. We identified that all factors in the model were needed for successful implementation, and where factors were missing, Willman's model was verified. Discussion on each factor, illustrated with examples (using pseudonyms), is given below.

4.1 Successful Companies

The research undertaken in Computer Craft and DataNet, provided *actionable first steps* and *pressure for change* resulting in the successful implementation of the SPI initiative. Considering Willman's framework, this fits into the "successful implementation" category when combined with the other factors previously existing in the companies: *leadership and vision*, *capable people* and *effective rewards*. In DataNet, *pressure for change* had also arisen from the focus to become ISO9001 certified. Positive improvements were also evident within Software Solutions, which also belongs to the "successful implementation" category. In this case, *pressure for change* was there because the company wanted to become ISO9001 certified. The project manager 'championed' much of this change, and devised those actions which were implemented within the organisation.

The most successful SataSPIN company, ReTech, has incorporated SPI as an essential part of their business strategy. The company had entered a demanding market and was committed to improve their quality both in products and processes. ReTech had also appointed capable resources for this work. The only thing missing was the *actionable first steps*, which was solved when they joined the SataSPIN. Results of their SPI efforts can clearly be seen in the positive customer feedback, expanding market share and solid economic result. ReTech is now considering obtaining an ISO9000 certification.

4.2 Disinterest – lack of *Pressure for Change*

In Ríomhaire, there was little evidence of improvement within the organisation, mainly because the company was already ISO9001 certified, and a goal in the current climate was not improving the current process, but rather ensuring that the process would continue to attain this level during audits. The software developers themselves did not feel part of the ISO9001 process and as a group, they did not have any pressure for change. The researchers concluded that, for changing the software process, they belong to the "disinterest" category as identified in the framework above. However, in the future, this lack of change in the software process could cause difficulties in Ríomhaire, particularly if customers require a software-based standard such as the Capability Maturity Model or SPICE, as "ISO9001 describes only the minimum criteria for an adequate quality-management system, rather than addressing the entire continuum of process improvement" (Paulk, 1995b).

4.3 Evaporation – lack of *Effective Rewards*

In LiftUp the pressure for change came from rapid growth of the company. In two years the company grew from 6 to 12 persons. The personnel were young, very well motivated and eager to adapt SPI. The management lead their SPI efforts and the SPI Initiation Framework was applied. The main problem was that the company invested somewhat in external assessment and training services but did not reward their own personnel. On the contrary, it was expected that the key people carry out SPI related activities in addition to their normal duties in customer projects. LiftUp has gained from SPI but the advancement has been relatively slow and there have been multiple start-ups to improve the same processes. Finally the company hired a new person solely to support SPI.

4.4 Frustration – lack of *Actionable First Steps*

Pike Software was started as an outsourced IT service company. In this case the pressure came from the management's vision to change the company's operations from leasing people to project based subcontracting. A quality manager had been appointed but, within the company, there were many conflicting views about what was needed. Frustration was already obvious when we started with a preliminary assessment of their processes. Management interpreted the results, and eventually, there was too much work to setup the processes. No commitment to proceed could be obtained and the frustration grew. Nevertheless, the company received later an ISO9000 certificate.

4.5 Disillusionment – lack of *Leadership and Vision*

In Codeboys the *pressure for change* came mainly from the software engineers in the company. The product they produced was very well accepted in the market and they had recently started exports. The outcome was increasing support and maintenance needs. They had a common view that something needed to be done and that SPI as an approach would fit their needs. In the beginning the engineers worked hard to find out improvement ideas and the work seemed to proceed well according to their plans. However, the managing director wanted to avoid the related responsibility and asked the outside consultants to "act as a police to ensure that everyone follows the process". This lack of *leadership and vision* hit the engineers motivation very hard and they became disillusioned with their efforts.

4.6 Disengagement – lack of *Capable People*

Willman also mentions *capable people* as a factor in the success of change management. In the context of the small indigenous software development company, these are the developers on whom the success of the software process is dependant. While studying software development companies, we noted that there was rarely an occasion where a developer was checked as to whether they carried out a process correctly. While this has its faults – processes can become chaotic very quickly – there must also be a trust built between the software development manager and the employee. Otherwise, in such a small group the overhead of checking and re-checking would increase development costs considerably. *Capable people* are indeed fundamental to success within a small company.

5 Conclusions

There are often opportunities for one discipline to learn from another. The different effects as stated in Willman’s model come from the study of business management. Those of us who are interested in software process improvement must remember that it too involves management of change. With this in mind, we can focus on the five factors required. Without them, as seen in our examples, our software process improvement initiative may fail. The positive aspect is that once we realise what factor is missing, it can easily be rectified, and in a complex and dynamic situation, each of these factors can and do change over time. What is ultimately important is that when there is successful implementation of software process improvement initiatives there are tangible software quality benefits to be gained by the small software development company.

6 Literature

Bergman, Bo and Bengt Klefsjo, 1994. Quality from Customer Needs to Customer Satisfaction, Studentlitteratur, Sweden.ICT Ireland. 2003. "Creating a World Class Environment for ICT Entrepreneurs."*ICT Ireland*.

Biro, M. & Messnarz, R.: Key Success Factors for Business Based Improvement, Proceedings of the EuroSPI´99, Pori, Finland, 1999.

Brodman, Judith G and Donna L. Johnson, 1995. "Return on Investment (ROI) from Software Process Improvement as Measured by US Industry", Software Process - Improvement and Practice, Pilot Issue, August, pp 35-47.IDA Ireland. 2001. "Achieve European Competitive Advantage in Software."

Combelles, Anne and DeMarco, Tom, 1998. "Viewpoint", Software in Focus, published by the ESSI News Project Team, Issue 1, March.

Fenton, Norman, Robin Whitty and Yoshinori Iizuka, Editors, 1995. Software Quality Assurance and Measurement - A Worldwide Perspective, International Thomson Computer Press, U.K.

Gillies, Alan C., 1992. Software Quality, Theory and Management, Chapman & Hall, U.K.

Humphrey, Watts S., 1989. Managing the Software Process, Addison-Wesley, Reading, M.A., U.S.A.

Humphrey, Watts S., 1998. "Three Dimensions of Process Improvement, Part I: Process Maturity" in CROSSTALK The Journal of Defense Software Engineering, February.

IBM Survey results, 1996. Ensuring Profitable Investment in Software Process Improvement, U.K.

Johnson, Donna L. and Judith Brodman, 1992. "Software Process Rigors Yield Stress, Efficiency", Signal, August, pp 55-57.

Jones, Capers, 1996. "Our Worst Current Development Practices" IEEE Software, March, pp 102-104.

Mäkinen, T., Varkoi, T. & Lepasaar, M.: Detailed Process Assessment Method For Software SMEs, Proceedings of the the EuroSPI 2000 Conference, Copenhagen, Denmark, 2000.

OECD: Measuring the Information Economy 2002 (www.oecd.org/sti/measuring-infoeconomy), 2002.

Paulk, Mark C., 1995a. "The Evolution of the SEI's Capability Maturity Model for Software", Software Process - Improvement and Practice, Volume I, Pilot Issue, August, pp. 3-15.

Paulk, Mark C., 1995b. "How ISO9001 Compares with the CMM", IEEE Software, January, pp 74-83.

Paulk, Mark C., 1996. "Guest Editorial for the Special Issue of Papers from the 29th Hawaii International Conference on Systems Science", Software Process - Improvement and Practice, Volume 2, Issue 3, September, pp. 165-166.

Paulk, Mark C., Bill Curtis, Mary Beth Chrissis and Charles V. Weber, 1993a, "The Capability Maturity Model for Software, Version 1.1", Technical Report SEI-93-TR-24, Software Engineering Institute, Carnegie Mellon University, U.S.A.

Richardson, Ita, Ryan, Kevin and Murphy, Eamonn, "Development of a Generic Quality Function Deployment Matrix", Quality Management Journal, Volume 9, Issue 2, March 2002.

Richardson, Ita, "Software Processes in Very Small Companies", Software Quality Professional, Volume 4, Issue 2, March 2002.

Richardson, Ita and Ryan, Kevin, "Software Process Improvements in a Very Small Company", Software Quality Professional, Vol. 3, No. 2, March 2001, pp 23-35.

Strader, L.B., M.A. Beim, and J.A. Rodgers, 1995. "The Motivation and Development of the Space Shuttle Onboard Software (OBS) Project", Software Process – Improvement and Practice, Volume1 Issue 2, December, pp 107-113.

Varkoi T.: Management of Continuous Software Process Improvement, Proceedings of the IEMC 2002, Cambridge, UK, 2002.

Varkoi T. & Mäkinen T.: Software process improvement network in the Satakunta region - SataSPIN, Proceedings of the EuroSPI´99, Pori, Finland, 1999.

Willman, Paul, 1996. "Organisational Change in the 1990s", lecture at University of Limerick, 10th October.

Zahran, Sami, 1998. Software Process Improvement, Practical Guidelines for Business Success, Addison-Wesley, U.K.

7 Author CVs

Ita Richardson, Ph.D.
Department of Computer Science and Information Systems
Computer Science Building, University of Limerick, Limerick, Ireland

Dr. Ita Richardson is Senior Lecturer and Head of Department of Computer Science and Information Systems at the University of Limerick where she has responsibility for 35 academic staff and over 50 research students. Her research work has focuses on Software Process Improvement in Small Software Development Companies, and on how software process can be affected by various business models. Her most recent study involves an investigation of the effect of takeover on software processes within small organisations. Dr. Richardson is a researcher on B4-STEP, Science Foundation Ireland Investigator Program which is examining processes, methods and tools in software development. Recently, she was project leader on a project funded by Nokia Mobile Phones, where the project team assessed the NMP new product development process.

Timo Varkoi, Lic. Tech.
Tampere University of Technology, Information Technology, Pori,
P.O. Box 300, FI-28101 Pori, Finland

Mr. Varkoi has studied Computer Science at Tampere University of Technology and graduated Master of Science in 1989 and Licentiate of Technology in 2000. His working experience includes both software development and management responsibilities in industrial software organisations. Recently he has been working as a project manager in an ESF-funded project to improve software processes with small software companies. He is a competent SPICE assessor and the chairman of the SPICE-section of Finnish Software Measurement Association (FiSMA). His current interests include software process assessment and improvement related research, expertise and training combined with development of software intensive organisations. At present he is working on his dissertation as a research project manager at Centre of Software Expertise in Tampere University of Technology, Pori.

Validating the MODIST Approach to Managing Uncertainty for Distributed Software Development

Lisa Tipping, David Milledge and John Elliott

Abstract

MODIST is producing a method (and associated toolkit) to improve the way software systems are developed and managed within a distributed development environment. In particular, it aims to demonstrate the impact that distributed development techniques are having on software quality and associated uncertainties. Previous software quality studies have considered a project as a monolithic work entity rather than as distributed workgroups. The MODIST project aims to fill this important gap by using Bayesian Belief Networks (BBNs) to capture and measure the causes of variation and uncertainty about software quality in distributed development environments. The project is part of the European Commission's Framework 5 Information Society Technologies programme (Project IST-2000-28749). MODIST provides novel challenges in implementation and validation because technically complex and innovative technology needs to be implemented in a manner that is transparent to end-users. This paper presents the work of QinetiQ, the lead partner of the MODIST project, in formulating a validation strategy that can be applied to novel technology in a software process improvement environment. The validation strategy operates at two levels. At the macro level, the strategy addresses the evaluation of the underlying assumptions, rules and information contained within the MODIST model against industry best practice and empirical data. At a more detailed level, the validation strategy examines the effectiveness of the model at a working or project level from empirical project data and feedback.

Keywords

Validation, Trial, MODIST, Distributed Software Development, Bayesian Belief Networks, Risk, Software Quality, Prototypes, Software Process Improvement

1 Introduction

The paper describes a strategy that is used by the MODIST project to validate the introduction of novel technology and a new methodology into a Software Process Improvement (SPI) environment; and that can in principle be applied as a generic approach by software practitioners. The new technology is in the form of a prototype or technology demonstrator. The paper is based on the experience of validating the MODIST method and associated toolkit. The purpose of the MODIST method is to improve the way the development of software is managed within a Distributed Software Development (DSD) environment.

In the context of this paper, the term validation means to evaluate whether the technology and methodology fulfils its intended purpose when placed in its intended environment. Validation occurs during a trial, where the term 'trial' is defined as the specific environment and set of activities associated with performing the validation. The validation strategy, as in the MODIST project, may encompass a programme of trials. The purpose of the validation strategy is to define the overall approach for evaluating the new technology and methodology against the stated objectives and aims. The strategy enables the communication of the goals, objectives, scope and design of the trial(s) to all relevant stakeholders. This ensures that all stakeholders and trial participants understand the validation process, their role and the requirements of the trial. Additionally, each trial is conducted in a manner that is coherent and consistent with the other trials in the programme.

The objectives of validation are to identify means to improve and exploit the technology, to facilitate the roll out of new technology and processes in a DSD environment (if appropriate), and to reduce the risk of the technology or methodology failing to meet its objectives after implementation.

2 Background about MODIST

This paper presents experience gained during the MODIST project from the perspective of the lead partner, QinetiQ. The purpose of the MODIST project is to produce a technology demonstrator in the form of a method, and associated toolkit. This is designed to help improve the way software systems are developed and managed within a Distributed Software Development (DSD) environment. The demonstrator is a knowledge based decision support tool that assists in managing project risk and assessing software quality. It is important that the complex technology of the demonstrator be 'hidden' from the user, and that the tool is as 'user-friendly' as possible. This is because the final product is targeted at a wide variety of users, and must be intuitive to use. Users include project, software, and quality managers, and independent assessors.

The MODIST project is part of the European Commission's Framework 5 Information Society Technologies programme (Project IST-2000-28749). The project is a 30-month programme of research that commenced in October 2001. The partners in the MODIST consortium are QinetiQ, Agena, Philips Digital Systems Laboratories, Philips Innovative Applications, and Israel Aircraft Industries (IAI). The consortium of partners collectively has expertise in the DSD arena, and specialises in Bayesian probability and decision support. The consortium represents trial participants from multiple sites, organisations and countries across the world. All partners contributed to the requirements for the MODIST method and tool; Agena led the tool development; QinetiQ, Philips, and IAI are performing the validation activities.

2.1 Why MODIST is novel

Rapid changes in the world of software technology are making distributed development an increasingly feasible, cost-effective and popular paradigm. This is because the most appropriate resources are exploited irrespective of geographical location. A DSD environment is typically characterised by development teams that are located across different company sites, organisations, sectors and nations. There are specific risks associated with DSD environments, in addition to the normal risks of software development. Different working practices have to be reconciled, and effective communication is crucial.

Good support for decision-making implies good support for risk management. Yet traditional metrics approaches, often driven by regression-based models for cost estimation and defect prediction, provide little support for managers wishing to use measurement to analyse and minimise risk. In addition, previous software quality studies have considered a software development project as a monolithic work entity rather than as distributed workgroups.

Consequently, new and innovative decision support methods and tools are required if consistent quality is to be achieved, and of particular interest is the use of measurement and modelling techniques. MODIST has addressed this need by using Bayesian Belief Networks (BBNs) to capture the impact that distributed development techniques are having on software quality and associated uncertainties. MODIST uses BBNs to augment sparse data with expert judgement. MODIST describes this approach as Bayesian Process Control (BPC). BPC overcomes the difficulties of applying traditional Statistical Process Control (SPC) by including qualitative measures to reflect the diversity of factors in DSD projects. MODIST combines different life cycle models, stages and activities with hard and soft evidence that impact on the end-software product. Hard evidence is factual, objective, and data orientated; soft evidence is subjective in nature, and is opinion and knowledge based.

2.2 MODIST Application Areas

MODIST aims to support decision making within a DSD environment in the following application areas:

- **Project development**: helps the project to achieve goals concerning cost, time, quality and performance, and functionality.
- **Acquisition**: helps in monitoring the performance of the supplier's work packages, and in assessing the evolving products of the allocated work packages.
- **Assessment**: helps independent auditors, reviewers and assessors to assess the status of the evolving project and products against requirements and plans, and the status of their attributes (e.g. supplier performance, safety, security, risk, cost and quality).
- **Process improvement**: helps business management and quality/process review boards to identify project trends in programme performance and evaluate corrective actions, and therefore improve their processes.

3 Validation Strategy

This section outlines the MODIST approach to validation of the MODIST demonstrator, and the trial process architecture used by the MODIST project.

3.1 Approach to the Validation of MODIST

The MODIST validation strategy encompasses three validation approaches: a programme of structured trials, structured reviews, and informal feedback. The validation strategy documents the validation approaches and provides techniques and guidelines for each approach. Each individual trial has a trial plan, and the strategy provides a template for constructing the trial plan. This helps the trial coordinators assess the scope and coverage of trial objectives across the trial programme. The performance of each trial is in a different DSD environment, and the results collectively contribute to the evaluation of the MODIST Method against its objectives. Participants in the validation activities include the MODIST project team members, users, and external contacts in the MODIST User Group.

The evaluation activities range from informal, general and unstructured feedback from the MODIST partners and the MODIST user group, to formal structured trial participation by the MODIST partners and external trial participants.

On MODIST, validation occurs at two levels: concept and project (see Figure 1).

.

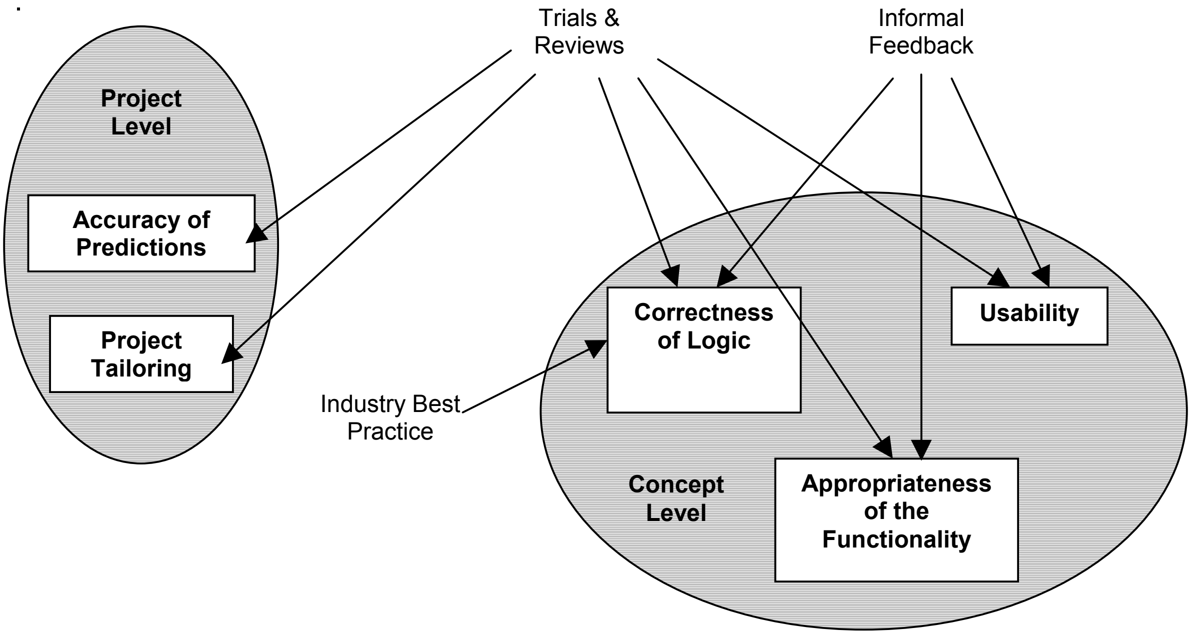


Figure 1: Levels of validation

The concept level addresses the evaluation of the correctness of the logic of the model, the appropriateness of the functionality, and the usability of the demonstrator tool. The trials, structured reviews and informal feedback provide qualitative information about all of these characteristics. Industry best practice is in the form of an analysis of why software projects fail, and the expert derived data from the CMMI [CMU 2002] model. It provides qualitative feedback about the logic of the model. In addition, individual project trials assess two characteristics of the demonstrator at the project level. Firstly, the accuracy of the predictions that the model makes is evaluated. Secondly, the ease and effectiveness of tailoring the demonstrator to the characteristics of individual projects is determined.

3.2 MODIST Trial Process

It is important to plan the activities of the structured trials. This section describes the process architecture for validation, modified from [McAndrews 1993], and illustrated in Figure 2. This process enables data to be simultaneously collected and analysed to help make decisions and determine trial results with respect to the trial goals. Additionally, the process allows feedback to be obtained about how to improve the validation process itself.

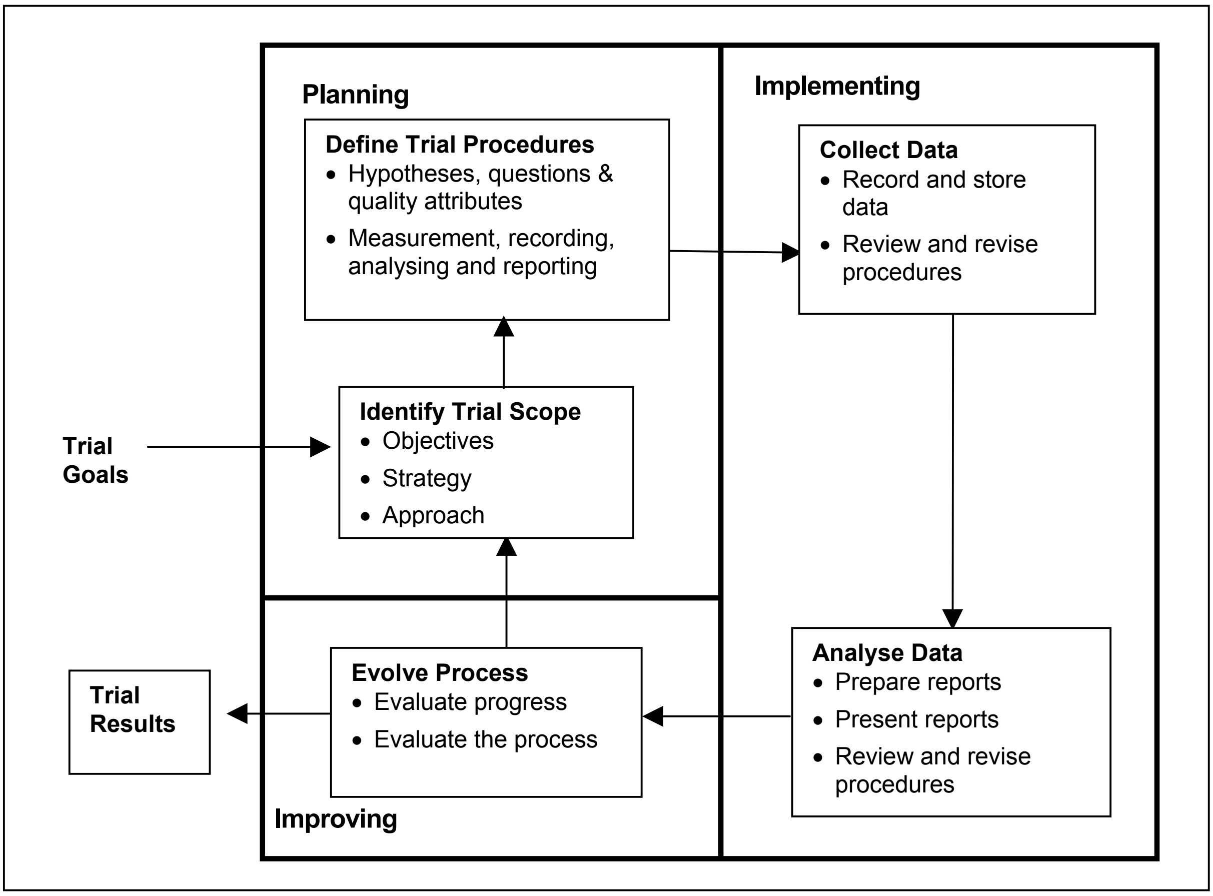


Figure 2: MODIST Trial Process Architecture

4 Investigation Techniques

This section outlines investigation approaches and techniques available to validate a new technology and methodology.

4.1 Investigation Approaches

Two investigation approaches are considered: the scientific method and the systems approach. The scientific method is the process by which scientists, collectively and over time, endeavour to construct an accurate (that is, reliable, consistent and non-arbitrary) representation of the world [Wolfs 2003]. The systems approach concerns acquiring first-hand knowledge of the subject under investigation by comparing the system under use with the perceived values, expectations and objectives of the individual. The scientific method has been very successful in advancing the natural sciences. However, it has not been as successful in investigating complex human-activity based systems, the fundamental controversy being whether the adoption of such an approach is at all appropriate for dealing with complex issues arising in social systems [Tipping 1999].

Software development is a human design activity. Hence, the scientific method is not valid in its truest sense because the results are not repeatable; different software engineers design code in different ways and the interactions of software teams are dynamic and spontaneous. In terms of the software projects participating in the MODIST trial, some software projects are similar but they are not identical in terms of team composition or the product being produced. Pragmatically, a scientific type approach is only appropriate for software projects where the same team of individuals develops very similar products. The systems approach is therefore the most appropriate form of investigation for evaluating the perceptions of users and trial participants.

4.2 Types of Investigation Techniques

The investigation technique used by the scientific method is the formal experiment. A formal experiment is rigorous, planned and the variables are controlled and measured.

There are a number of investigation techniques available when using the systems approach. Fenton and Pfleeger (1997) outline a number of these techniques. The techniques available include surveys, historical data, case studies and prototyping.

A survey is a retrospective study of a situation to try to document relationships and outcomes. When performing a survey, there is no control over the situation and it is not possible to manipulate variables. The use of historical data from past projects to retrospectively validate whether the model accurately predicts the outcome of a software project is a particularly relevant form of investigation for the MODIST project. A case study is a research technique where variables are identified that may affect the outcome of an activity. The activity is documented in terms of its inputs, constraints, resources and outputs.

Prototyping is a technique specifically designed for evaluating software. Lichter et al (1994) define their understanding of the term ’prototyping’ and the principles underlying the terminology in three parts. Firstly, prototyping is an approach based on an evolutionary view of software development, affecting the development process as a whole. Secondly, prototyping involves producing early working versions (“prototypes”) of the future application system and experimenting with them. Thirdly, prototyping provides a basis for discussions among groups involved in the development process, especially between users and developers. In addition, prototyping enables us to adopt an approach to software construction based on experiment and experience.

The MODIST trials use a number of techniques: surveys, historical data, case study and prototyping. However, the most pertinent approach is to treat the MODIST demonstrator as a prototype and evaluate MODIST as such. MODIST was initially evaluated against the software quality attributes defined in the standard: Information technology – Software product quality [BS ISO/IEC 9126-1:2001], which defines a system for measuring the quality of a software product. However, the conclusion from the exercise was that this approach had limited applicability while MODIST is still a technology demonstrator rather than a software product.

5 Collecting Data

This section describes ways of collecting data, and presents an outline of the MODIST trial template.

5.1 Education and Training

The attitudes of all people involved in collecting data contribute to the success of the evaluation activity. Sympathetic attitudes are encouraged through a carefully designed and executed education and training programme [Mayhew and Worsley 1990]. The users must appreciate that the aim of prototyping is to enable the participants to learn about aspects of the required system. They must not expect prototypes, especially early exploratory ones, to be perfect operational systems. Conversely, they must understand that what appears to be a highly suitable system, may in reality be a rather fragile, incomplete prototype under its professional exterior.

It is important to explain that a prototype (or technology demonstrator) has a number of strengths: it is a better representation of what a system does than is a description of the behaviour [Schneider 1996]. Conversely, design rationale (why the system was built and why it was built in a certain way) is easier to describe in text or figures. Documenting judgements about the prototype are often the most important type of information a prototype elicits from users. Schneider (1996) suggests that users need to understand why they are using a prototype, and what the demonstrator is proving, falsifying or demonstrating. He suggests documenting the answers to the following questions, and communicating them to trial participants (or users). Why was the prototype built? What is the prototype supposed to do? And what does it actually do? How does the prototype do it? Why does it do it in this way? What are the concepts used by the prototype?

5.2 Methods of Data Collection

This section outlines two methods, being used by the MODIST project, for collecting data about prototypes.

5.2.1 User and Developer Evaluation Sessions

Smith and Dunckley (2002) propose the following approach to prototype evaluation. Two distinct activities are described for the identification of usability problems linked to design factors: Developer – User Contextual Evaluation (DUCE) sessions and Team Evidence Analysis (TEA) sessions. In DUCE sessions, users provide evidence relating to the overall usability of the system by interacting with an exploratory prototype and verbalising their experience. In these sessions, designers take a subordinate role. In TEA sessions, the evidence captured in a series of DUCE sessions is refined by a team of developers into a number of design factors and possible solutions. This approach was appropriate for the MODIST trial for evaluating the usability of the MODIST demonstrator for specific scenarios.

The objective of the DUCE session is to make the user explain their normal working practice in relation to the prototype while they are interacting with it. In order to assist the user in verbalising their experience, the developer/evaluator leading the session is required to ask the user a number of open questions as interaction progresses. Questions are provided and structured as Norman's Seven Stages of Action [Norman 1986]. These can be asked at appropriate points within each discrete user action. On completion of the required tasks, further general questions are asked of the user about the system as a whole.

After the DUCE session the developer/evaluator is required to analyse the records of different individual users by transcribing what are considered to be relevant comments pertaining to usability. The evidence is documented and grouped by each subtask, for examination in the TEA session. It is important that developers are presented with actual user comments rather than an interpretation of them. The purpose of the TEA sessions is to refine the evidence captured in the DUCE sessions into a number of key design decisions. The objective of the sessions is to identify solutions to user problems, and prioritise these in terms of importance, difficulty of repair and a cost benefit analysis. User importance is based on the number of users who encounter the problem and its significance to the user. This involves the design team recognising the factor causing the problem and alternative solutions.

5.2.2 Product Evaluation

Heaton (1992) suggests that users make purchasing decisions based on how well a product performs, according to the following criteria.

- The degree to which it supports a user's tasks, and the amount of functionality that can be exploited
- The effort a user has to expend to release a product's functionality (e.g., time to learn, number of errors made and speed of getting a job done)

- The perceived cost of a product (both financial cost and other costs – such as opportunity lost/gained)
- The perceived benefit a user derives from use.

These procurement requirements reflect the criteria of: effectiveness, learnability, flexibility and attitude. This approach was appropriate for MODIST in evaluating the exploitability of the product.

5.3 MODIST Trial Plan Template

This section outlines extracts from the MODIST trial plan template. It is composed of some suggested questions that may be asked of trial participants. The questions chosen are tailored to the goals of the individual trial participant, and it is not suggested that all of the suggested questions are relevant in all cases. In addition, trial participants can add their own pertinent questions.

User questions can be broken down into generic high-level questions about MODIST, questions about how users perform individual tasks using the MODIST demonstrator, and questions relating to the factors affecting whether users might purchase MODIST as a product.

5.3.1 User Questions relating to MODIST Objectives

The systems approach in the trial concerns the evaluation of the trial participants' perceptions about MODIST. These questions seek to evaluate the MODIST demonstrator against the objectives of the MODIST project, and can be broken down into a number of criteria. Examples of these criteria include: the external perception of whether MODIST is credible and accepted within the user community; whether the BBN models capture a complete range of distributed development scenarios; and whether they capture the essence of distributed development scenarios.

The scientific approach addresses the following hypotheses. The first hypothesis states that the cost of collecting, analysing and reporting metrics on distributed software development projects that use MODIST is less than on distributed software development projects that do not use MODIST. The second hypothesis states that the code produced on distributed software development projects using MODIST has a lower defect density than code produced on distributed software development projects that do not use MODIST.

5.3.2 Task Based User Questions

This part of the template is in the form of a DUCE checklist, and is based on Norman's Seven Stage model.

Norman's Stage	**Potential Questions**
1. Form a goal	How does the screen help you to decide how to achieve your task?
2. Form an intention	How does the screen suggest what actions you need to take next to perform the task?
3. Specify the action sequence	How does the system let you know you are making progress?
4. Execute the action	
5. Perceive the resultant system state	What is the most important part of the information visible now?
6. Interpret the resultant state	How has the screen changed to show what you have achieved?
7. Evaluate the outcome	How do you know that what you have done was correct? How would you recognise any mistakes?

The following summary questions are asked: Taking the system/your tasks as a whole.....

System Needs	In which way did it meet the needs of the tasks which you were asked to perform?
System Problems	Were there any general problems – or things you would like to see changed?
Interface Qualities	In what ways did the interface show simplicity/difficulty? In what ways did the interface show consistency/lack of consistency? In what ways did the interface let you feel in control/ out of control? In what ways did it give feedback so that you know what you were doing?
Screen Qualities	In what way was the screen layout helpful for you to achieve the tasks? In what way was the choice of objects helpful for you to achieve the tasks? In what way was the order of actions helpful for you to achieve this task?

5.3.3 Product-Based User Questions

Product-based user questions address the factors affecting whether a user might purchase MODIST as a final product. Examples of product-based questions include the following: To what degree does MODIST support decision-making on software projects? How much would users be willing to pay for a product like MODIST? What is the perceived benefit that a user would derive from using MODIST?

6 Conclusions

The MODIST project has produced a novel and technically complex demonstrator that is designed to be used on DSD projects. Indeed, the MODIST trial is itself a DSD project because trial participants from multiple sites, organisations and countries across the world are involved.

The Validation Strategy describes the MODIST approach to validation, provides a description of some available investigation and data collection techniques, and outlines the MODIST trial plan template. It is important to have a coherent strategy to ensure that each trial participant, in the trials programme, approaches the trial in a consistent and coherent manner. It also ensures that each trial participant understands the trial process, their role and the requirements of the trial. Finally, it helps the MODIST project team to consolidate the results from all the trials.

7 Acknowledgements

We wish to thank all the MODIST Partners for their support and assistance during the course of the work and reviewing this paper. The partners in the MODIST consortium are QinetiQ, Agena, Philips Digital Systems Laboratories, Philips Innovative Applications, and Israel Aircraft Industries (IAI).

8 Literature

BS ISO/IEC 9126-1:2001 Information technology – Software product quality – Part 1: Quality Model

CMU 2002 Carnegie Mellon University, Capability Maturity Model Integration, V 1.1, Staged Representation, CMU/SEI/-2002-TR-012/March 2002.

Fenton & Pfleeger 1997 Software Metrics: A Rigorous and Practical Approach, Norman E Fenton and Shari Lawrence Pfleeger, Thomson Computer Press, 1997

Heaton 1992 What’s wrong with the user interface? How rapid prototyping can help. Human Applications, Elms Grove, Loughborough, 1992.

Lichter et al 1994 Prototyping in Industrial Software Projects – Bridging the Gap Between Theory and Practice, Lichter Horst, Schneider-Hufschmidt Matthias, and Zullighoven Heinz, IEEE Transactions of Software Engineering, Vol 20, No 11, November 1994

Mayhew & Worsley 1990 Software Prototyping: Management Implications, University of East Anglia, England 1990

Norman 1986 Cognitive Engineering In: User Centred Systems Designs: New Perspectives on Human-Computer Interaction 7, 347-374, Norman D.A., Lawrence Erlbaum, Hillsdale, NJ

McAndrews 1993 Establishing a Software Measurement Process, Technical Report CMU/SEI-93-TR-16, Software Engineering Institute, 1993

Schneider 1996 Prototypes as Assets, not Toys. Why and How to Extract Knowledge from Prototypes, Schneider Kurt, IEEE Experience Report, Proceedings of ICSE-18, 1996

Smith & Dunckley 2002 Prototype evaluation and redesign: structuring the design space through contextual techniques, Smith Andy and Dunckley Lynne, Interacting with Computers 14, 2002, 821-843

Tipping 1999 Facilitating the Development of a Modern Electronic Design Process Using Soft Systems Methodology, Lisa Tipping, MPhil Thesis, University of Hertfordshire, 1999

Wolfs 2003 Introduction to the Scientific Method, Teaching Notes, University of Rochester, US, 2003

9 Author CVs

Lisa Tipping currently works as a Consultant for the Consulting Department at QinetiQ. Lisa has a BSc in Applied Physics, a MSc in Software Engineering and a MPhil. The title of the MPhil is ‘Facilitating the Development of a Modern Electronic Design Process using Soft Systems Methodology (SSM)’. She is a Chartered Physicist and a Member of the Institute of Physics, and a Chartered Engineer and a Member of the Institute of Electrical Engineers. Lisa has experience of a variety of process improvement activities including: the development of Capability Maturity Model capability, advising software projects about software quality assurance and performing internal audits.

David Milledge is a Consultant within QinetiQ Consulting and manages the MODIST project. David has a BSc degree in Physics. He has over 30 years of software engineering experience, with particular expertise in processes, methods and tools. He has performed a wide range of technical management roles, mainly in support of large real-time systems, and subsequently simulation and training systems. As the Software Process Improvement Manager of a large, multi-site software division, David successfully introduced a number of new or modified processes, as recognised by local awards and the results of formal assessments against the SW-CMM.

John Elliott is a Consultant within QinetiQ Consulting. John has a MSc in Information Systems Engineering, and is a Member of the British Computer Society and the Institute of Statisticians. He is QinetiQ’s leader and board member within the EuroSPI consortium. John has spent over 25 years specialising in advanced system and software engineering, in particular, where quality and integrity is vital. John has experience as a manager, developer, consultant and assessor of software-based systems.

For further information about the MODIST project, please contact David Milledge at dmilledge@qinetiq.com.

Improving Software Organizations: An Analysis of Diverse Normative Models

Jan Pries-Heje, The IT University of Copenhagen, jph@itu.dk
Richard Baskerville, Georgia State University, baskerville@acm.org

Abstract

The purpose of this paper is to develop a framework for comparing and contrasting alternative normative models for improving software organizations. The framework provides an analytical classification of the activities and scope of improvement schemes. Building on this classification, we can characterize normative models according to four dimensions: Agenda, observation mode, analysis mode, and synthesis mode. The paper explains in detail how we arrived at each of the four dimensions and give selected examples of models that falls within each of the characterisations. Finally the paper discusses how the framework can be used in practice in an organization and concludes that you have to discuss your own values and beliefs in the organization to select the best normative model for your purpose. But the framework presented in this paper will provide valuable guidance to your selection.

Keywords

Software process Improvement, organizational process improvement, normative models, CMM, Bootstrap, Balanced Scorecard, Six Sigma, Juran, Business Excellence, EFQM, ISO 9000

1 Introduction

Every day consultants all over the world recommend that software organizations take this step or that step in order to improve their performance. Sometimes, however, the basis for these recommendations can be highly dubious such as the Five-Minute Rule (Weinberg 1985, p. 67): “Clients always know how to solve their problems, and always tell the solution in the first five minutes.”

Many consultants, however, use normative models to derive recommendations from the observations they are making. But what is it that they are doing? Is it a sound process? Or is it pure magic? Or are the model just hiding what we coined dubious above?

Thus we set out to study normative models for improving software organizations documented in the literature. We know off course that many consultants and consulting companies have models of their own that are never published. Never the less we believed that a study of normative models in the literature could reveal interesting patterns in the process of going from observation to recommendation.

1.1 Software Organization Improvement

A number of authors have surveyed models for software organization improvement. Thomson & Mayhew (1997) surveyed several maturity models and identified a number of different approaches. Austin & Paulish (1993) described a number of commonly applied methods for improving the software development process, seen in the context of one normative model, namely the Capability Maturity Model developed at the Software Engineering Institute in Pittsburgh (Paulk et al. 1995). The European PICO project resulted in a book (Messnarz & Tully, 1999) that included a comprehensive catalogue of 34 models and standards (Tully et al., 1999).

More recently Aaen et al. (2001, 2002) have advocated a 3 by 3 categorization of more than 50 software process improvement approaches. The three main categories are concerns: (1) How should the intervention be managed, (2) Approaches to the intervention process, and (3) Perspectives in the intervention process. Management of the improvement - or intervention as it is called - covers how you organize?, how you plan?, and how you get feedback? The main distinction in relation to approaches is whether you follow a norm, builds on commitment, or focus on (slow) evolution? Finally the 3 perspectives is called process, competence and context.

1.2 Examples of Software Organization Improvement

The Capability Maturity Model - or just CMM - is a framework characterizing a 5-step path for software process improvement (Paulk et al. 1995). The path describes key processes at each of five levels. The description includes a number of goals at each level. An organization has to meet the goals at one level to reach the next. I.e. to go from the basic level 1 where behavior is characterized by being adhoc and intuitive to level 2 you need to achieve the goals incorporated in six key process areas: requirements management, sub-contractor management, project planning, project tracking, quality assurance, and configuration management.

The CMM became so popular that a large number of other models using the same 5-step path were invented. For example People-CMM, Integrated Product Development CMM, Systems Acquisition CMM (Cooper & Fisher 2002), Testing Maturity Model (cf. Burnstein 2002) and several others. Finally a large number of the CMM-models were summoned in CMM-integrated - or just CMM-I. The major difference between CMM and CMM-I is that Systems Engineering is included. That means that there are more key processes making it more useful in companies that are not only developing software but combining software with for example mechanics, electronics or other kinds of hardware (cf. Ahern et al. 2001, Chrissis et al. 2002).

Within the world of software development the issue of improving software development took two directions around 1990. In Europe software-developing companies took the ISO 9000 standard and adapted it to be used in a software world. In this adaptation companies could use the ISO 9000-3 on how to use the ISO 9001 standard for software. In USA software-developing companies at the same time took the CMM in use. Bootstrap was developed in a European research project (Kuvaja et al., 1994). Bootstrap combined the features of both ISO 9000 and the CMM. Furthermore the BOOTSTRAP model was extended and adapted to include guidelines from European Space Agency (ESA)'s PSS-05 software development standard. The major difference between Bootstrap and CMM - besides having other processes in the model - was that the step-wise improvement of CMM was replaced by a continuous model where each process could be measured and improved on a 1-to-5 scale.

In 1993 work started on an international standard for Software Process Improvement and Capability dEtermination (SPICE). Today SPICE is recognised as a series of Technical Reports under ISO called ISO 15504. SPICE takes the maturity levels of CMM and extends them in two ways. First a level zero is added meaning that a process is not carried out at all. Second, SPICE also apply the continuous improvement model that Bootstrap used meaning that the processes don't have to move in groups to a given level – like the six key process areas between level 2 and 3 in CMM. Instead an organization can improve the processes where it is needed the most seen from a business perspective.

1.3 Organizational Process Improvement

There is additionally a large body of management literature dealing with ways in which any organization can improve the quality of their activities. These improvement schemes also apply to software organizations. Thus when aiming at improving software organizations – as is the title of this paper – it is relevant to look at normative models for organizational process improvement.

1.4 Examples of Organizational Process Improvement

In the 1950s and 60s problems in the delivery of defense products led to the invention of defense standards. This then led to the launch of a British Standard 5750 in 1979 which then became the foundation for the international standard for quality systems, ISO 9000, in 1987. The standard was revised considerably in 1994 and 2000. The standard applies to an organization providing products to a customer. In the 1994 version of standard there were 20 clauses in chapter 20 that an organization could be audited against (Mirams & McElheron 1995).

Six Sigma is an organizational quality system associated with its highly publicized success in Motorola and GE. Six Sigma originated as a model to improve the quality of processes already under control. The name originates from the assumption that defects in controlled quality processes will average ± 3 sigma (standard deviation) from the mean in a normal distribution. Six Sigma forces designed service level tolerances that seek to eliminate defects out to ± 6 sigma. Such service levels correspond to 3.5 defective parts per million produced. Today, six sigma aims for "business success" through an "understanding of customer needs, disciplined use of facts, data, and statistical analysis, and diligent attention to managing, improving, and reinventing business processes" (Pande et al. 2000, p xi) It uses a 5-step roadmap that strongly values measurement and customer involvement.

Balanced Scorecard (BSC) was originated by Kaplan and Norton (1996) mainly as a reaction to the weaknesses of traditional financial measures. BSC is not just a measurement model but a model that enables an organization to clarify their vision and strategy and translate them into action, which is both iterative and rational. Process improvement is related because it is framed and fostered by BSC. The concept of a scorecard is related to each of four key perspectives on the organization: (1) learning and growth, (2) business process, (3) customer, and (4) financial. Learning and growth is a perspective on the knowledge and skills of the people in the organization. It is foundational. If the organization advances in this area, it will lead its people to innovate and improve from the other perspectives. They will improve the business processes. Better business processes will bring more satisfied customers. Happy customers will improve the organization's financial performance. The improvement process begins to cycle or spiral upward as better financial performance permits more latitude for further learning and growth.

The scorecard for each of these perspectives includes setting clear objectives in each of the four perspectives. Objective performance measures are set that reveal performance relative to the objective. Targets are set for future performance levels. Then initiatives are defined and implemented to drive performance toward the targets. The scorecard for each perspective relates how each objective has defined clear measures, targets and initiatives. In this way the approach is similar to Deming's "Plan, Do, Check, Act" Total Quality Management cycle. The objectives in all perspectives are driven by an express vision and strategy set that should advance the organization from all four perspectives simultaneously, yet interdependently.

Juran - often mentioned as one of quality gurus - developed a 3-phase model for quality control (cf. Juran & Gryna 1988): (1) quality planning, (2) quality control, and (3) quality improvement. Juran build on work from the 30s on statistical process control where control charts were invented. A control chart is a graph where the thing being measured is shown. If measures exceed the upper or lower control limit something has to be done to bring the process back into statistical control. Juran's idea of improvement is to "tie in" the difference between the upper and lower level.

The Business Excellence Model – also called EFQM (2003a) - was founded in 1988 by 14 major European companies as a kind of European answer to quality improvement along the lines of the Malcolm Baldridge Model in the USA and the Deming Prize in Japan. The EFQM Business Excellence Model was developed with the objective of describing the principles and practices that create high performing organisations. The criteria could then be used by organisations to assess their performance and drive continuous and sustainable improvement. The EFQM Business Excellence Model can be used as a framework for organisational self assessment, but is also used as the basis for judging entrants to the European Quality Award.

2 Our Study of normative models

Our study was started by curiosity. Numerous times we have meet normative models being used in practice. But often there was no good explanation why a specific model was used in a specific situation? In many cases the only explanatory factor was that the person using the normative model only was knowledgeable about or certified to use that specific model – and not any of the others available.

We then set out to gather information on normative models. And we found an impressive set of models. Quite early we realised that we had to limit ourselves to some models. The first choice we made was to eliminate models that did not improve the software developing organization as a whole but only part of it. For example this led us to eliminate the normative portfolio approach by Applegate et al. (1999) that can be used by an organization to select which projects to start.

We then analyzed the remaining models trying to identify dimensions that could be used to distinguish and characterize different type of models. Our first finding was what we have called the „Agenda". That is the perspective on outcomes (of improving the software organization) that drives the entire process improvement effort. Second, we realised that many normative process improvement models has a lot in common with research methods. From this realisation we then derived three dimensions that can be used to characterise and understand normative models in-depth, namely „Observation", „Analysis" and „Synthesis".

Through an iterative analytic process we have been able to distil three 2-by-2 matrices for the four dimensions. In the following sections we discuss and provide examples of each of the four dimensions that together form our framework.

3 The Agenda in process improvement

The agenda is the perspective on outcomes that drives the entire process improvement project. The strategic mode defines what kind of recommendations can be derived from the project. It is equivalent to the set of research questions that drive a research project. The strategic mode determines to a certain extent, a predefined set of possible recommendations that can be developed by the improvement project.

It refers to the goal or aim of the process in terms of specific possible outcomes. The set of possible values of the recommendations may be closed in terms of its scope, ambit, compass or program. There may be a closely framed set of possible recommendations, or there may be an open unstructured set invented in whatever way the experts choose.

3.1 Agenda examples

ISO 9000 has a very simple agenda: OK or not-OK. The organization is measured against a standard. Deviations are recorded and documented. The deviations then have to be corrected before an OK (that can lead to an ISO 9000 certification) is issued. In principle you can view ISO 9000 as a binary process with two steps: Zero and OK. However, when you have reached OK then you are not done. In the most recent version of ISO from 2000 the focus in on continuous improvement. This means that you go from a direction – from not-OK to OK – and to a kind of balance where you continuously improve what is found to most urgent.

The agenda in CMM is to improve your processes by moving up to the next step on a 5-step scale. If you fulfil the goals of all the six key process areas at level 2 then you are recommended to undertake activities that will lead to fulfilment of the 7 key process areas at level 3. To support this agenda a number of studies have been undertaken. An overview can be found in Emam & Briand (1999).

The agenda in Bootstrap is partly the same as for CMM; the organization is measured against a number of processes. But rather that just recommending that the lowest scoring processes should be improved it is recommended to take business goals into account. “Business goals can be kept as the main push for process improvement” (Kuvaja, 1994, p. 106).

The Six Sigma agenda is focused on five sets of deliverables that correspond to each of the five steps in the six sigma roadmap. On the six sigma agenda :

- An inventory of value-delivering activities.
- Output and service specifications or requirements that closely tract the factors that drive customer satisfaction.
- Specific measurements that describe the current organizational performance baseline, the capability of each process, and descriptions of new or enhanced measurement methods and resources needed to support continuation of organizational process improvement
- An unstructured set of new designs or redesigns of processes in need of improvement. These process improvements regard new demands, new technologies, or increase speed, accuracy or cost performance.
- A continuing organizational improvement program. There are process controls, clear statements of process ownership, response plans to adapt strategies, and an organizational culture for improvement

The exact nature of each of these deliverable sets, and consequently the agenda for Six Sigma is context-driven. Those involved in the improvement project have considerable leeway to define these deliverables according to the nature of the organization and its stakeholders.

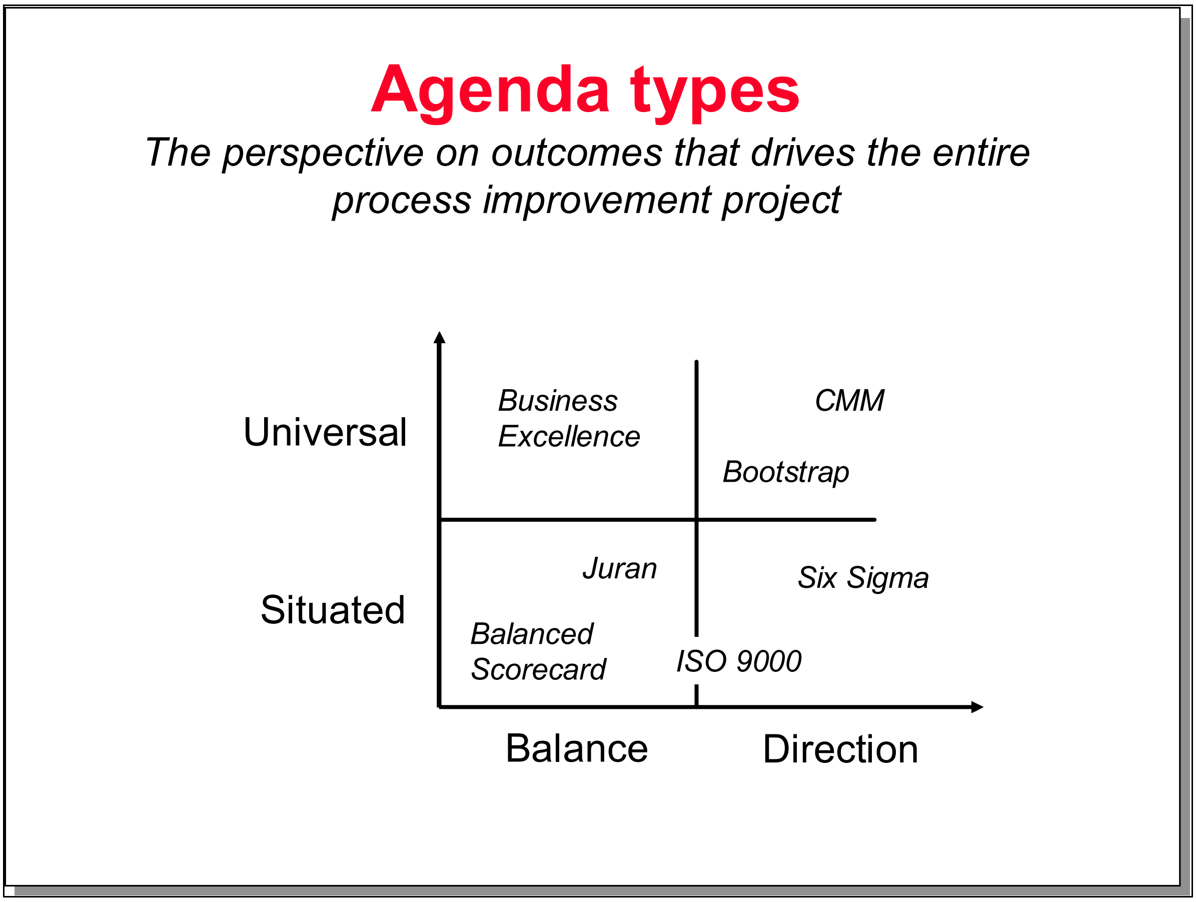
The agenda in EFQM is to strive for Business Excellence (EFQM 2003a, 2003b). The EFQM framework has nine criteria. Five of them are so-called enablers: (1) Leadership driving (2) Policy and Strategy, (3) People, (4) Partnerships and (5) Resources and Processes. The other four criteria are in common called results: (6) Customers, (7) People and (8) Society are leading to (9) key performance results. Excellence is defined as outstanding practice in managing the organisation and achieving results, all based on a set of 8 fundamental concepts (1 to 8 above) leading to results (numbered 9 above) The Model recognises there are many approaches to achieving sustainable excellence in all aspects of performance. But at the core the belief in the model is that the 8-9 criteria should be balanced.

The agenda in BSC embraces a pronounced vision, mission and strategy for the organization. Beyond this the BSC process seeks to set concerted objectives, measures, targets and initiatives in each of the four perspectives: learning and growth, business process, customer, and financial.

Finally the agenda in Juran is to bring processes under statistical control. There is no specific direction, and it is contingent what processes that are selected for control

3.2 Agenda types

The various improvement agenda can be analyzed along two dimensions. The first dimension is based on the perception of the ideal outcome of the process. Some improvement models focus on an outcome of balance in the organizational activities and resources for an optimum performance. Alternatively, another part of the models aim to provide a direction for the organization, a path to a future, desirable state. The second dimension regards the viewpoint taken by the agenda. One viewpoint assumes that software organizations are quite similar and a set of universal solutions can be applied in most organizations. This viewpoint sees the improvement agenda as rather universal across organizations. Another viewpoint sees organizations as highly unique instances, intersections of very particular resources and people. The improvement solutions from this viewpoint must be equally unique, or at least carefully adapted. This viewpoint sees the improvement agenda as particularly situated within each particular organization. In figure 1 we have shown seven selected normative models along the two dimensions identified.



Agenda types

The perspective on outcomes that drives the entire process improvement project

Figure 1: Agenda types in normative models for improvement

4 Observation

The mode of observation is defined by the assumptions and techniques used to collect data about the organization and its processes. This data collection mode may include observations in anticipation of the process improvement project, observations during the improvement project, and observations following the improvement project. Assumptions can include such philosophical aspects as the objective-subjective dimension.

Borrowing from the study of research design, we can identify two main types of observation:

- Detached Observer. This is the traditional form of objective data collection. It assumes that one or more observers can stand more-or-less detached from the organizational setting and capture data that is generally independent of any serious observer bias. One valued aspect of consultants is that they can bring unbiased observations into consideration. The detached observer mode is frequently enhanced by various kinds of instrumentation. These instruments can include questionnaire surveys, interview guides, assessor response forms, subject response forms, and focus groups.
- Participant Observer. When people in a position to change or influence the organizational setting collect data, they are considered one of the study subjects and therefore participants in the research setting. Their observations are recognized to have bias, but achieve additional value because of their deep understanding of the problems at hand. Participant observers are often considered to be in a "learning mode" rather than a "data collection mode." Organizational development often follows this style of data collection and is sometimes call action research or action learning. There are fewer forms of instrumentation, but these can include self-assessment and diary or journal keeping.

Besides the two main types of observation a number of normative models also use secondary sources. Secondary sources are data extracted from existing repositories of information about the organizational improvement setting. Valuable information about problems and solution can be found in memos, correspondence, mail, system and organizational documentation. Intranets, websites, databases, and knowledge bases are also good examples of secondary data sources.

4.1 Observation Examples

CMM recommends a number of different observation strategies. Master & Bothwell (1995, page 32) writes in the so-called Appraisal Framework: "Observations should come from multiple and independent sources to ensure that they do not reflect just one person's perspective concerning the appraised entity's processes. Participants in presentations and interviews may be biased by the comments others make; therefore, observations should also be supported by data from multiple data collection sessions". And to ensure validity Dunaway et al. (1996, page 9) also set up three rules for the validation (quality) of data: "(1) Observations are based on data from at least two independent sources, e.g., two separate people or a person and a document. (2) Observations are based on data obtained during at least two different data gathering sessions. (3) Observations are confirmed by at least one data source reflecting work actually being done, e.g., an implementation level document or an interview with a person who is performing the work."

So CMM employs both detached observation mode and secondary sources. Furthermore CMM requires at least one person from the organization being appraised to be part of the appraisal team (Dunaway 1996, page 7-8): "The team shall consist of a minimum of 4 and a maximum of 10 team members. At least one team member must be from the organization being assessed." Thus CMM also employs the participant observer mode to a certain degree.

Bootstrap is a little different from CMM. Detached observation mode is used as the main mechanism for observation as typically 2 assessors go into the organization and use group interviewing and 2 questionnaires as their main data collection mechanism (Kuvaja et al. 1994, chapter 7)

Six Sigma – Six Sigma is strongly anchored to objective measurements. The data that is used to determine the improvement deliverables is expected to be repeatable and non-subjective. Consequently, Six Sigma is tightly anchored to the idea of instruments. Observations are preliminaries necessary to discover good measurement instruments. Six Sigma assumes that observers who discover and operate these instruments are detached even though they may actually be participants. Participant bias would interfere with the objective measurements required to prove six sigma improvements. The affinity for Six Sigma is toward detached observation and secondary data. Examples of such six sigma observations include hold time per incoming call, minutes to board an airplane, units exceeding target cost, etc. While the focus of six sigma improvement is quantitative, it does incorporate some less structured qualitative data in the form of customer contacts and interviews for the purposes of establishing customer needs. Objectivity still prevails as an assumption behind the collection of this data.

ISO 9000 typically is carried out by certified auditors coming from outside. They observe the organization for a period of time, and based on their interpretation of what is happening they certify or fail the organization under study.

BSC – Like Six Sigma, BSC aims at highly objective measurements. The improvement strategy is a matrix of four sets of objectives, and each must be associated with clear performance measures. The assumption is one of objective proof; the improvement setting is shown to have improved by these clear measures. The measures must also support a clear target value for the measures: A goal for the improvement efforts. Questionnaires and secondary data are prevalent among the ideal observations. Thus the normal mode of observation would be detached.

The Business Excellence Model EFQM encourages the Participant Observer Mode in the form of Self Assessments (2003a): “Adoption of the process of Self-Assessment is EFQM’s recommended strategy for improving performance … applied rigorously, Self-Assessment will help organisations, large and small, in the private and public sectors, work more effectively”. Self-Assessment is a comprehensive, systematic and regular review of an organisation’s activities and results referenced against the EFQM Excellence Model. So the idea is that you take the EFQM Excellence Model, answer a number of questions on your organization’s activities and results, and then you know what to do come closer to or achieve excellence - You know where you are un-balanced today.

EFQM, however, also is used as part of National Quality Awards. A company can ask to be taken into consideration for an award. And then a couple of outside-assessors will visit the company and use the EFQM-model to assess the organisation. So in this case the Detached Observation Mode is applied.

Finally Juran focuses on measurable data that can be used for statistical analysis. Typically the data will be collected by people in the organization aiming at improving quality.

5 Analysis

The mode of analysis is the perspective and techniques used to plan the sense-making activities that follow from the collection of data. This analysis might take place as a phase of activity following observation, or it might be continuous or iterative during the project. The analysis mode is the way in which the process improvement practitioners structure, organize, compile and summarize the often-voluminous data collected during the project. Analysis usually aims to discover partitions or categorical classes in the improvement setting.

1. Partition – This technique uses the data to divides the improvement setting into component parts. It involves analyzing the organization into parts according to each part’s functions, behavior or outputs.
2. Classification – This technique uses the data to discover categories of events, activities, or other phenomena in the improvement setting. Classification is central to systematics, the study and building of taxonomies.

There are many ways to approach such analyses. Here are some of the common techniques from the natural and social sciences:

1. Quantitative Statistics – These techniques encompass the various mathematical transformations used to compile numeric data into descriptive or inferential abstractions of the improvement setting.
2. Subjective Interpretation – Interpretive analysis springs from the knowledge and experience of the analysts and is usually applied to unstructured qualitative or textual data. Analysts can use logical argumentation in discovering patterns in the data. Alternative interpretive approaches might be more historic in nature, seeking stories or scenarios. There are also cyclical or iterative approaches to interpretation, such as hermeneutics or critical analysis, which recognize the emergent nature of interpretation.
3. Content Analysis – This is one of a number of techniques for converting qualitative data to quantitative data. Conceptual representations in the form of words or phrases are identified, located and counted for further statistical analysis using either descriptive (frequency counts) or analytical (collocation analysis) arithmetic manipulation.
4. Change Analysis – This technique involves isolating and identifying the changes in an improvement setting over a period of time. Commonly used in participative approaches such as action research, change analysis reduces and structures qualitative data by separating organizational consistency from organizational variance.

Improvement models may incorporate an explicit toolkit of analytical techniques. Tools in this toolkit may range widely across techniques such as those above, or they could focus in a central area, such as statistical analysis. In cases where a model incorporates a toolkit concept, there still may be a preferred tool, or a subset of preferred tools, that are most clearly consistent with the forms of observation promoted by the model.

5.1 Analysis examples

In CMM the analysis activity is mainly about consolidating information into a manageable set of observations that then can be categorized with reference to the KPAs of the CMM. Dunaway et al. (1996, p. 20) tells: “The team must reach consensus on the validity of observations and whether sufficient data in the areas being investigated have been collected. It is the team’s responsibility to obtain sufficient information to cover the organization, the software development life cycle, and the CMM components within the assessment scope before any rating can be done…”. When that is done each KPA is rated satisfied or unsatisfied. All the KPAs at one level must be satisfied for the organization to achieve a given level on the 1-5 CMM scale.

Bootstrap is very much alike CMM in that the assessors should agree on validity of observations. But then the likeliness tops. In Bootstrap each process is rated on a 1 to 5 scale using not only whole numbers but also quartiles. I.e. the outcome of the project management process can be 2.25. A fairly complicated algorithm is used to go from the approximately 200 answers in the Bootstrap questionnaire to the rating (cf. Kuvaja et al., 1994, page 83-84).

Six Sigma – Consistently with the focus on objective, quantitative data, six sigma values statistical forms of analysis. The preliminaries to these statistics, such as an interpretive analysis to define categories or classifications for statistical data analysis, seem to be regarded as somewhat obvious and intuitive. While little attention is paid to the development of classifications from the data, the analysis of these classes is pre-eminent. Classifications might be weighted for priority ordering or analysts apply statistical techniques to the data to develop Pareto diagrams, histograms, time-series plots, correlation diagrams, and other forms of quantitative analysis.

BSC – Similarly, high-quality quantitative measures are valued. Mathematical and statistical analyses are most valued. Performance ratios and descriptive or analytical statistics from survey analysis are seen as good examples of BSC analysis.

For the Juran model, ISO 9000 and EFQM see text under “Observation examples”.

5.2 Modes of Observation and Analysis

If we want to distinguish between modes of analysis the obvious distinction comes from the use of mathematical forms of model, like statistics. Some improvement models focus on such quantitative measures. Alternatively, other models are oriented toward highly unstructured forms of analysis. These models are sometimes oriented toward learning through interpretive or logical analysis of interviews, studies of organizational memoranda, texts, etc.

The modes of observation and analysis can be shown together in a 2-by-2 matrix. Interestingly, it seems that modes of observation and modes of analysis is not correlated. Thus we can identify normative models in all the 4 quadrants as shown in figure 2.

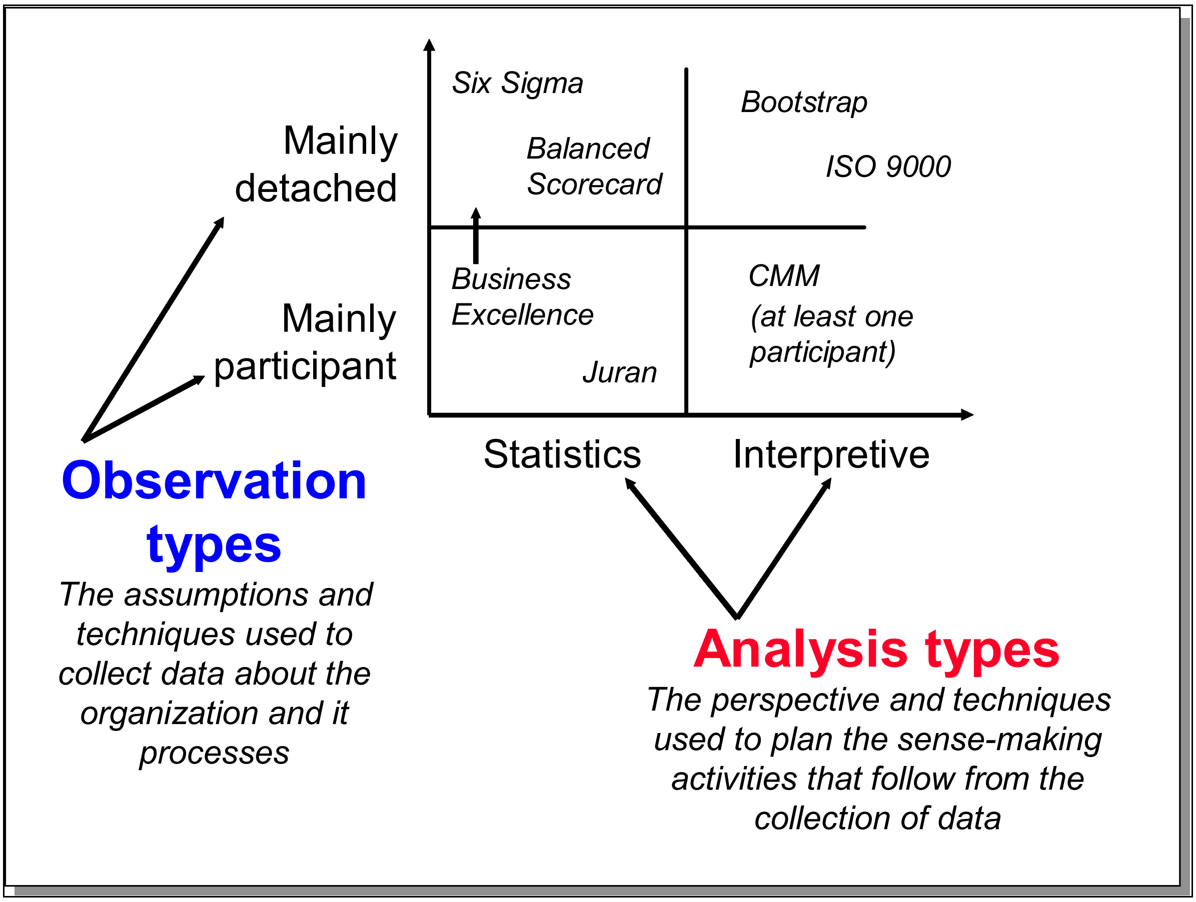

Mainly detached
Mainly participant
Six Sigma
Balanced Scorecard
Bootstrap
ISO 9000
Business Excellence
Juran
CMM (at least one participant)
Statistics
Interpretive
Observation types
The assumptions and techniques used to collect data about the organization and it processes
Analysis types
The perspective and techniques used to plan the sense-making activities that follow from the collection of data

Figure 2: Observation and Analysis types combined for normative models of improvement

6 Synthesis

The synthesis mode is the perspective and techniques used to draw conclusions from the analyzed data. It regards the way in which recommendations are synthesized from the analytical compilations of the observations. The synthesis process links the analysis back to the agenda of the improvement model by developing the recommendations that are the goals of the entire process. The social and natural sciences incorporate a wide variety of such synthesis forms, except that these are seen as operations at the theoretical level rather than the empirical level.

Some authorities do not distinguish the analysis and synthesis modes. When the synthesis of recommendations from analytical structures becomes conflated into a single process authors will regard both activities as analysis. It is not difficult, however, to logically separate the two activities *post hoc*. Since consistency is important for the purposes of comparing the improvement models, we will consistently distinguish these separate modes.

Examples of how synthesis can be carried out are:

- Modelling – Explicit or implicit models are the most common distillation of analytical data. Stage, phase, or cycle models describe the processes for improvement. Causal models describe antecedents of desirable or undesirable states or outcomes. Theoretical models describe relationships between concepts discovered in the analysis. Next to the creative leap, modelling is the most common form of synthetic process where objective assumptions temper the reliance on the experience and knowledge of the analysts.
- Causal Maps – There are many different forms of modelling, but causal modelling is certainly important among them. Causal modelling can take such forms as influence diagrams, causal flow diagrams, and causal maps. Causal maps provide a good example of how causal modelling applies to interpretive analysis. These maps usually illustrate the interconnected nature of events and activities, and show how these are dependent on one another as precedents.
- Solution Checklist – Because the synthesis mode must sometimes map the analysis onto a predefined agenda of improvement solutions, the synthesis may be seen as a simple process of selecting the recommendations from a predefined inventory of known solutions. Some improvement models implement the “Law of the Hammer,” i.e., if the only allowable tool is a hammer, then somehow every problem must be converted into a nail (Weinberg, 1985, p. 53). In this case, the synthesis mode provides a means by which to deduce one of the checklisted solutions from the analysis of the improvement setting.
- Action Research – Synthesis of recommendations can also arises from practical activity within the improvement setting. Tightly related to participant observation and change analysis, action researchers experiment with improvement recommendations in small steps. Such synthesis embraces a “learn-by-doing” that seeks to anchor its recommendations in the experience of the particular improvement setting at-hand. The synthesis emerges from the improvement process in an unstructured, but obvious way.
- Creative Leap – Many approaches will regard synthesis as a taken-for-granted human activity that will not benefit from formal structuring. With the data analysis at-hand, the indications for recommended improvement strategies will be somewhat obvious to the experienced practitioner.

In general these examples of synthesizing form two different types. The first one we could call „Synthesis Model“ meaning that there is a distinct model that we use for synthesis. The first three examples above would fall in this category. The second one we could call „Know how“ where it is more or less tacit how the synthesis happens. The two last examples above would often fall under this type.

6.1 Synthesis examples

At the end of a CMM appraisal a conclusion is synthesized that is then presented to the sponsors Dunaway et al. (1996, p. 11): “A final findings briefing must be given to the sponsor that presents the strengths and weaknesses of each KPA within the assessment scope, as well as a KPA profile that indicates whether KPAs are satisfied, unsatisfied, not rated, or not applicable.”

In Bootstrap the synthesis is made from a matrix of maturity and business goals. A process that is rated important in business terms and at the same measured to be weak using the 1 to 5 scale will be a top priority for improvement. However, the description of how to rate business importance is not rigorously described but mainly left to the knowledge of the assessors.

Six Sigma – The analytical concepts get refined into causal hypotheses as part of a cyclical process called a “root cause hypothesis/analysis” cycle. These hypotheses are partly represented in a causal model called a cause-and-effect diagram. The synthesis also includes process mapping, a form of stage model. The synthesis culminates with a three-stage “solution statement” creation process. Stage one is idea generation, stage two is the generation of a variety of solutions, and stage three is a solution selection process. Six Sigma offers a bit of variety in the way it structures synthesis, and recognizes the importance of causal modelling as an outcome of the objective analysis. However, the solution statement structure values the creative leap by knowledgeable and experienced professionals to provide the final link between the analysis and the agenda of recommendations.

For the synthesis Juran is close to Six Sigma. Synthesis is done using a statistical model of variation. But the recommendation is not given. It depends.

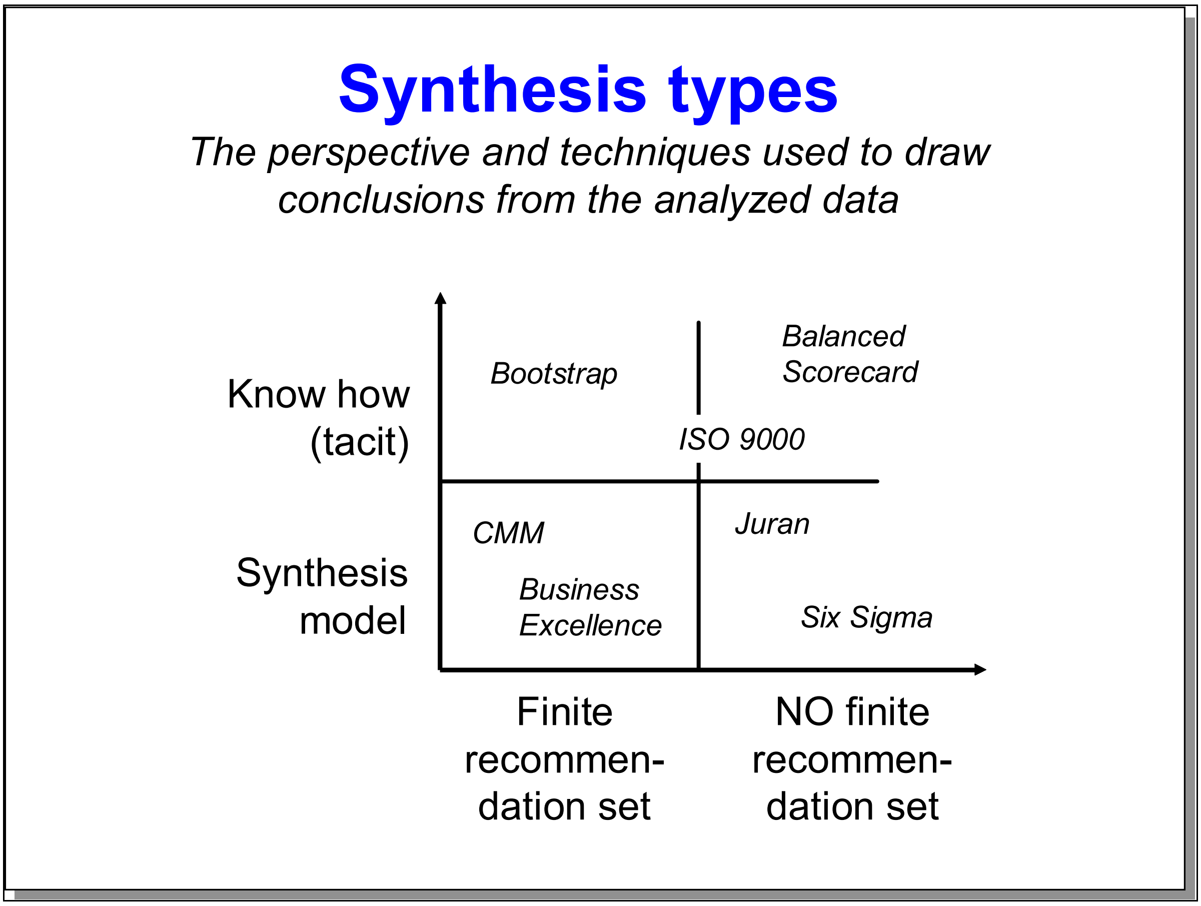
BSC – The synthesis activities are framed by the need to set strategies, objectives and targets. In addition clear performance measures must be discovered. In the early BSC cycles, benchmarking organizational performance measures against external standards (such as competitors) is particularly important. As the process matures, focus shifts toward a process of “strategic learning,” an unstructured advance of strategies, objectives and targets as the organization improves. Synthesis also has a focus on strategic alignment, the need to align organizational resources in order to foster the various initiatives and achieve the targets.

In the ISO 9000 model before 2000 there was a typical recommendation set, namely that one or more of the 20 clauses was not fulfilled and had to be corrected. With the new 2000-version of ISO 9000 and its focus on continuously improving there is no finite set of recommendations. It depends. And it is left to the interpretation of the auditors to decide on what.

Finally the Business Excellence model EFQM has a finite set of recommendations, namely to improve what is missing in relation to the “correct” balance described in a model that also is used for deriving the synthesis.

6.2 Modes of Synthesis

The modes of synthesis are analyzed along two dimensions in Figure 3. The first dimension is based on whether the recommendation set is predefined by the improvement approach. Some approaches have a tightly structured menu of possible recommendations. Other approaches only loosely describe the kinds of recommendations that are possible from the process. The second dimension – that we discussed earlier in this section - distinguishes the degree to which the recommendations leap from the analysis in a highly creative fashion. Some approaches assume that an experienced practitioner will just somehow know how to translate the analysis into concrete recommendations for improving the software processes. Others provide a model of the possible analytical outcomes and the meanings of these outcomes in terms of possible recommendations. This latter form of synthesis provides formulas for distilling recommendations from the analyses.



Synthesis types

The perspective and techniques used to draw conclusions from the analyzed data

Know how (tacit)

Synthesis model

Bootstrap

Balanced Scorecard

ISO 9000

CMM

Business Excellence

Juran

Six Sigma

Finite recommendation set

NO finite recommendation set

Figure 3: Synthesis types combined for normative models of improvement

7 Discussion and conclusion

There are clearly a number of different models for improving software organizations. Each model has distinct activities regarding its agenda, observation mode, analysis mode and synthesis modes. These characteristic dimensions can be combined and used to classify the approaches. But how can the framework be used in practice?

7.1 Using our framework in practice

We have developed a framework that can help organizations make sense of different normative models, and link them to their improvement goals. Our framework distinguishes between four significant dimensions: agenda, observation, analysis and synthesis. These dimensions can be used to help you examine your organization’s needs and then select an appropriate improvement model in an informed and systematic way.

In the concrete you should try to place your own organization, its values and beliefs in the framework. Let us take agenda as an example. A *universal perspective* puts its faith in maturity models such as CMM and Bootstrap. More generally, the universal perspective focuses on models of best practices and, consequently, models of general software-process problems. A *situated perspective* on the other hand focuses on what software practitioners and their managers perceive as problems in the software process.

Furthermore you should consider what end goal you have in mind? Do you for example believe that CMM level 5 is an attractive state-of-the-practice in your organization? In general *directed normative models* provide an organizational direction toward better and better software development. Such models assume that the organization is in a software development "state" and has the opportunity to change to an improved state. Usually, this goal also assumes that there will be further opportunities to improve so the improvement process is seen as progressive On the other hand *balanced normative models* have a clear aim toward building a value system for software quality among other important organizational values. A balancing strategy assumes that something is missing in the software development organization: some activity, value, or element that must be added or restored in order to improve the software process.

You should then continue to observation. At one end you find the *detached observation* that relies on some kind of assistance from outside actors, such as process or measurement consultants, to appraise the organization. The assistance from outside is often organized as a specific project with a start date and an estimated completion date. At the other end you have the *participant observation* where you as part of your daily work in organization implements appraisals of the routine software work carried out by the employees in the organization.

Again you should ask yourself: Which type of observation suits my values and beliefs best? Next you go to analysis and synthesis modes. Here the questions are: Will I rely on statistics or interpretation? Do I need a synthesis model, or will I rely on people being knowledgeable enough to tacitly synthesize results? And do I prefer a normative model with a given outcome – in the form of a predefined finite set of recommendations?

After having answered all these questions you are probably in a situation where the four dimensions in our framework point to different normative models. In this situation our recommendation would be to follow the advice from the agenda dimension simply because that is the most fundamental of the four dimensions.

7.2 Conclusion

We have now presented a framework for comparing and contrasting alternative models for improving software organizations. The framework provides an analytical classification of the activities and scope of improvement schemes. Building on this classification, we can characterize the agenda in normative models according to two dimensions. One dimension distinguishes the kinds of improvement goals: a goal of balancing the organization as distinguished from a goal of directing the organization. A second dimension distinguishes the assumptions about the improvement venue: a venue in which a universal model defines the improvement setting as distinguished from an idiographic or situated problem that defines the improvement setting.
Furthermore we can characterize normative models in relation to how they observe, analyze and synthesize.

8 Acknowledgements

The authors wish to thank Ivan Aaen, Bo Hansen, Karl Kautz, Peter Axel Nielsen and Jacob Nørbjerg and an anonymous reviewer for valuable comments and discussions.

9 Literature

Aaen, Ivan, Jesper Arent, Lars Mathiassen & Ojelanki Ngwenyama (2001). A Conceptual MAP of Software Process Improvement. Scandinavian Journal of Information Systems, 2001, 13: 81-101.

Aaen, Ivan, Jesper Arent, Lars Mathiassen & Ojelanki Ngwenyama (2002). Mapping SPI Ideas and Practices. Chapter 2 in: Mathiassen, Lars, Jan Pries-Heje & Ojelanki Ngwenyama (2002). Improving Software Organizations. From Principles to Practice. The Agile Software development Series. Addison-Wesley, Boston, MA, USA.

Ahern, Dennis M. Aaron Clouse & Richard Turner (2001). CMMI Distilled: A practical introduction to integrated process improvement. SEI Series in Software Engineering, Addison-Wesley, Boston, MA, USA.

Applegate, L. M., F. W. McFarlan & J. L. Mckenney (1999): Corporate Information Systems Management: The Challenges of Managing in an Information Age. 5th Edition. Irwin McGraw-Hill, Boston, MA, USA.

Austin, Rob & Dan Paulish (1993). A Survey of Commonly Applied Methods for Software process Improvement. Technical report CMU/SEI-93-TR-027, Software Engineering Institute, Pittsburgh, PE.

Burnstein, Ilene, Ariya Homyen, Taratip Suwanassart, Gary Saxena & Rob Grom (2002). A testing Maturity Model for Software Test Process Assessment and Improvement. Chapter 6.2 in Ted Daughtrey (Ed.) (2002). Fundamental concepts for the software quality engineer. American Society for Quality, Quality Press, Milwaukee, Wisconsin.

Chrissis, Mary Beth, Mike Konrad & Sandy Shrum (2002???). CMMI - Guidelines for process integration and product improvement. SEI Series in Software Engineering, Addison-Wesley, Boston, MA, USA.

Cooper, J. and M. Fisher (2002). Software Capability Maturity Model (SA-CMM) version 1.03. Pittsburgh, CarnegieMellon . Software Engineering Institute.

Crosby, P. B. (1979) Quality is Free: The art of making quality certain, Mentor, New American Library, New York.

Deming, W. E. (1982) Out of the Crisis, MIT Center for Advanced Engineering Study, Cambridge, Mass.

Dunaway, Donna K., and Masters, Steve. Technical Report CMU/SEI-96-TR-007 ESC-TR-96-007, CMM^{SM}-Based Appraisal for Internal Process Improvement (CBA-IPI): Method Description, pp. 7-8, 14, 38.

EFQM (2003a). Introducing Excellence. Brochure. ISBN 90-5236-072-3. Downloaded 31 May 2003 from http://www.efqm.org/.

EFQM (2003b). Eight essentials of excellence - The Fundamental Concepts and their Benefits. Updated in 2003. Downloaded 31 May 2003 from http://www.efqm.org/.

Emam, Khaled El & Lionel Briand (1999). Costs and benefits of Software Process Improvement. Chapter 7 in: Richard Messnarz & Colin Tully (Eds.) (1999). Better Software Practice for Business benefit: Principles and Experience. IEEE Computer Society, Loa Alamitos, California.

Juran, J. M., and Gryna, F. M. (1988) Juran's Quality Control Handbook 4th Edition, McGraw-Hill, New York.

Kaplan, R. & D. Norton (1992). The balanced scorecard: measures that drive performance. Harvard Business Review, January-February 1992, pp. 71-79.

Kuvaja, Pasi, Jouni Similä, Lech Krzanik, Adriana Bicego, Samuli Saukkonen, Günter Koch (1994). “Software Process Assessment & Improvement. The BOOTSTRAP Approach.” Blackwell Publishers.

Masters, Steve & Bothwell, Carol (1995). CMM Appraisal Framework, Version 1.0. (CMU/SEI-95-TR-001, ADA 293300). Pittsburgh, Pa.: Software Engineering Institute, Carnegie Mellon University, 1995.

Messnarz, Richard & Colin Tully (Eds.) (1999). Better Software Practice for Business benefit: Principles and Experience. IEEE Computer Society, Loa Alamitos, California.

Mirams, Mike & Paul McElheron (1995). Gaining and maintaining the new international quality standard: The BS EN ISO 9000 toolkit. Finacial Times, Pitman Publishing, London.

Pande, P. S., Neuman, R. P., & Cavanagh, R. R. (2000). *The Six Sigma Way: How GE, Motorola, and Other Top Companies Are Honing Their Performance*. New York: McGraw-Hill.

Paulk, M. C., Weber, C., Curtis, B., and Chrissis, M. B. (1995) The Capability Maturity Model: Guidelines for Improving the Software Process, Addison-Wesley, Reading, Mass.

Robey, D. & Boudreau, M.-C. (1999). Accounting for the Contradictory Organizational Consequences in Information Technology: Theoretical Directions and Methodological Implications. *Information Systems Research, 10*(2), 167-185.

Targett, David, David J. Grimshaw & Philip Powell (Eds.) (1999). IT in Business. A manager's casebook. Butterworth Heinemann, Oxford, U.K.

Thomson, H.E. & P. Mayhew (1997) Approaches to Software Process Improvement. Software Process - Improvement and Practice, 3, 1: 3-17.

Tully, Colin, Pasi Kuvaja & Richard Messnarz (1999). Software Process Analysis and Improvement: A Catalogue and Comparison of Models. In: Richard Messnarz & Colin Tully (Eds.) (1999). Better Software Practice for Business benefit: Principles and Experience. IEEE Computer Society, Loa Alamitos, California.

Weinberg, Gerald (1985). The secrets of consulting: a guide to giving & getting advice successfully. Dorset House Publishing, New York, N.Y.

Willcocks, Leslie, David Feeny & Gerd Islei (1997). Managing IT as a Strategic Resource. McGraw-Hill.

10 Author CVs

Jan Pries-Heje

has a Ph.D. from Copenhagen Business School. He now works at The IT University of Copenhagen. . His main research interests are information systems development, software engineering, and software process improvement. In particular he has carried out action research with industry on specific topics such as high-speed software development, IT project management, requirements specification, and successful organizational change with IT. He has published in these areas in journals like Journal of Accounting Management and Information Technology, IEEE Computer, The Data Base for Advances in Information Systems, European Journal of Information Systems, and Annals of Software Engineering.

Richard Baskerville

is professor of information systems and chairman in the Department of Computer Information Systems, College of Business Administration, Georgia State University. His research and authored works regard security of information systems, methods of information systems design and development, and the interaction of information systems and organizations. He is a Chartered Engineer, holds a B.S. *summa cum laude*, from The University of Maryland, and the M.Sc. and Ph.D. degrees from The London School of Economics, University of London.