



# Eur SPI'2006

European **S**oftware **P**rocess Improvement

11.-13.10.2006, SOKOS Hoetl Klmmel, Joensuu, Finland

<http://www.eurospi.net>

Eur SPI'2006

## EuroSPI 2006 Industrial Proceedings

### Partnership

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

### Supporters

University of Joensuu  
<http://www.joensuu.fi>  
FiSMA  
<http://www.fisma.fi>

Technical Report 10/06  
ISSN-NO: .....  
University of Joensuu ....

JOENSUUN  
YLIOPISTO



## Proceedings

The papers in this book comprise the industrial proceedings of the EuroSPI 2006 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), 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 13th 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.

## EuroSPI Chairs

General Chair Dr Richard Messnarz, ISCN

EuroSPI Marketing Chair Stephan Göricke, ISQI

EuroSPI 2006 Local Chair Prof. Markku Tukiainen, University of Joensuu, Finland

Scientific Program Committee Chair Ita Richardsson University of Limerick, Ireland

Scientific Program Committee Chair Prof. Per Runeson University of Lund, Sweden

Industrial Program Committee Chair Risto Nevalainen FiSMA and STTF

Industrial Program Committee Chair Jorn Johansen, DELTA

Industrial Program Committee Chair Mads Christiansen, DELTA

Industrial Program Committee Chair Nils Brede Moe, SINTEF

Tutorial Chair Dr Richard Messnarz, ISCN

Organizing Chair Adrienne Clarke BA, ISCN

## Industrial Programme Committee

Geir Amsjo, SPITIA, Norway

Beatrix Barafort, CRPHT, Luxembourg

## Session 0: Introduction

Andreas Birk, SD&M AG, Germany  
Mads Christiansen, DELTA, Denmark  
Lars-Ola Damm , Ericsson, Sweden  
Michael Ferchau, INVERSO, Germany  
Eva Feuer, MTA SZTAKI, Hungary  
Johann Flachs, Giesecke & Devrient, Germany  
Uwe Hehn, Methodpark, Germany  
Andre Heijstek, Improvementfocus, The Netherlands  
Tim Hind, Axa, UK  
Janos Ivanyos, Memolux, Hungary  
Jorn Johansen, DELTA, Denmark  
Ulrich Kolzenburg, Giesecke & Devrient, Germany  
Claudia Mathis, MagnaPowertrain, Austria  
Richard Messnarz, ISCN, Ireland & Austria  
Edda.Mikkelsen, DNV, Norway  
Nils Brede Moe, SINTEF, Norway  
Darja Smite, Riga Information Technology Institute, Latvia  
Gerhard Thüncher, ZF Friedrichshafen AG, Germany  
Tor Ulsund, BravidaGeomatikk, Norway  
Nils Jakob Villmones, Spacetec, Norway  
Hans Westerheim, SINTEF, Norway  
Bruno Wöran, DANUBE, Austria

### **EuroSPI Board Members**

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

### **Editors of the Proceedings**

Prof. Markku Tukiainen, University of Joensuu, Finland  
Dr Richard Messnarz, ISCN, Ireland  
Risto Nevailainen, FiSMA, Finland  
Sonja Koinig, ISCN, Austria

## ***Welcome Address by the EuroSPI General Chair***



**Dr Richard Messnarz**

EuroSPI is an initiative with 3 major goals ([www.eurospi.net](http://www.eurospi.net)):

1. An annual EuroSPI conference supported by Software Process Improvement Networks from different EU countries.
2. Establishing an Internet based knowledge library, newsletters, and a set of proceedings and recommended books.
3. Establishing an effective team of national representatives (in future from each EU country) growing step by step into more countries of Europe.

EuroSPI established a an experience library ([library.eurospi.net](http://library.eurospi.net)) which will be continuously extended over the next years and will be made available to all attendees. EuroSPI also established an umbrella initiative EQN (European Quality Network) which is funded by the EU Leonardo da Vinci Programme and establishes an European certification unit for T & Services professions. I therefore expect that EuroSPI partners will closely collaborate to form a group of national institutions in Europe representing a set of certified professions related with innovation and management.

Finally, keep in mind what companies stated about EuroSPI :” ... the biggest value of EuroSPI lies in its function as a European knowledge and experience exchange mechanism for SPI and innovation”.

## ***Welcome to Budapest by Prof. Markku Tukiainen***

It is a great pleasure to welcome you all to EuroSPI 2065 here in Joensuu, ....

**Markku Tukiainen**

**Local Chair**

### Contents

#### Experience Session I: SPI and Assessments

Qualification of Safety-Critical Systems in TVO Nuclear Power Plants 1.1

*Juha Halminen and Risto Nevalainen*

A SPICE-based Software Supplier Qualification Mechanism in Automotive 1.11

*Fabrizio Fabbrini, Mario Fusani, Giuseppe Lami and Edoardo Sivera*

Assessment Based Learning Systems – Learning from Best Projects 1.19

*Richard Messnarz und Damjan Ekert*

#### Experience Session 2: SPI and Communication Management

Using Groupware Technologies to Facilitate Organisational Learning and Software Process Improvement – A Case Study 2.1

*Riikka Ahlgren, Mirja Pulkkinen, Eleni Berki and Marko Forsell*

Success with Improvement – Requires the Right Roles to be Enacted – in Symbiosis 2.11

*Jan Pries-Heje and Jørn Johansen*

Measurement Practices in Fidenta 2.23

*Erkki Savioja and Markku Tukiainen*

#### Experience Session 3: SPI and Requirements Management

SPI of the Requirements-Engineering-Process for Embedded Systems Using SPICE 3.1

*Alexander Poth*

Extending the Rational Unified Process with a User Experience Discipline: A Case Study 3.11

*Hans Westerheim and Geir Kjetil Hanssen*

Improving Requirements Reuse: Case Abloy 3.19

*Jukka Heinänen and Markku Tukiainen*

#### Experience Session 4: SPI and Human Factors

Exploring National Culture in Software Development Practices 4.1

*Aileen Cater-Steel and Mark Toleman*

The Human Factor Deployment for Improved Agile Quality 4.11

*Kerstin V. Siakas and Errikos Siakas*

Managing Multi-Cultural and Multi-Social Projects in SPI 4.25

*Miklos Biro, Richard Messnarz and Janos Ivanyos*

#### Experience Session 5: SPI and Learning Factors

Using Multi Dimensional Scaling to Analyse Software Engineers' De-motivators for SPI 5.1

*Nathan Baddoo, Tracy Hall and Ciaran O'Keefe*

The Craving for External Training in Small and Medium-sized Software Companies –	5.17
A Trigger Effect towards Software Process Improvement	
<i>Hanna-Miina Sihvonen, Paula Savolainen and Jarmo J. Ahonen</i>	
European Quality Network (EQN) – A European Human Resource Strategy for Improvement	5.27
<i>Richard Messnarz, Franz Piller and Bruno Woeran</i>	
<b>Experience Session 6: SPI and Knowledge Management</b>	
ImprovAbility™ at the Project Level	6.1
<i>Anna Høeberg and Klaus Olsen</i>	
ImprovAbility™ in SPI projects at PBS	6.13
<i>Arne Staal Ibsen</i>	
Mining Best Practices of Project Management as Patterns in Distributed Software Development	6.27
<i>Antti Välimäki and Kai Koskimies</i>	
<b>Experience Session 7: SPI Assessments</b>	
A Lightweight Model for the Assessment of Software Processes	7.1
<i>Francisco J. Pino, Félix García, Francisco Ruiz and Mario Piattini</i>	
Adept – A Software Process Appraisal Method for Small to Medium-sized Irish Software Development Organisations	7.13
<i>Fergal Mc Caffery, Ita Richardson and Gerry Coleman</i>	
Lighthouse: An Experimental Hyperframe for Multi-Model Software Process Improvement	7.23
<i>Semih Cetin, Ozgur Tufekci, Erman Karakoc and Burcu Buyukkagnici</i>	
<b>Experience Session 8: SPI and Model Based Processes</b>	
Model-Based Test Design	8.1
<i>Anne Kramer</i>	
Project Management Based on Effective Practices: An Alternative Framework to SMEs	8.9
<i>Gonzalo Cuevas, Jose A. Calvo-Manzano, Ivan Garcia and Tomas San Feliu</i>	
<b>Experience Session 9: SPI and Systems</b>	
Towards Tools Support for Situation – Dependent Software Process Improvement	9.1
<i>Gábor Bóka, Katalin Balla, Rob J. Kusters and Jos J.M. Trienekens</i>	
European E-Security Manager Skills Card and Learning System	9.13
<i>Alen Salamun, Richard Messnarz and Damjan Ekert</i>	
<b>Experience Session 10: SPI and Measurement</b>	
A Framework for Improving Effort Management in Software Projects	10.1
<i>Topi Haapio</i>	
Statistical Process Control for Software Development – Six Stigma for Software Revisited	10.15

*Thomas Fehlmann*

### **Experience Session 11: SPI and Management**

Generating a Work Breakdown Structure: A Case Study on the General Software Project 11.1

Activities

*Topi Haapio*

A Risk Management Process 11.13

*Josef Horstkoetter, Joachim Fleckner*

### **Experience Session 12: SPI and Development Processes**

Adoption of Agile Development in Practice: A Qualitative Inquiry 12.1

*Inga Lagerquist, Margareta Lindmark, Janis Stirna and Jaana Nyfjord*

Combining Agile Software Projects and Large-Scale Organizational Agility 12.11

*Petri Kettunen and Maarit Laanti*

Towards a {(Process Capability Profile)-Driven (Process Engineering)} as an Evolution of 12.27

Software Process Improvement

*Clenio F. Salviano and Mario Jino*

# Qualification of safety-critical systems in TVO nuclear power plants

Juha Halminen  
*Teollisuuden Voima Oy*  
*Olkiluoto, Finland*

Lic. Tech. Risto Nevalainen  
*Finnish Software Measurement Association ry FiSMA*  
*Espoo, Finland*

## Abstract

Teollisuuden Voima Oy (TVO) operates two nuclear power plant units in Finland and has started to build a third one. The current nuclear power units have continuous need to maintain and update existing instrumentation and control systems (I&C).

Each new device shall be classified and qualified according to its safety requirements. Using modern technology means in practice that more and more components have programmable features. The reliability of such components has proven to be difficult to demonstrate due to the nature of flaws in software. Standards and rules given by authorities set the acceptance criteria for the components used in the safety systems of nuclear power plants.

As a result of this trend, there is a clear need for an integrated and effective method to qualify software intensive I&C systems in nuclear power plant units. Integration has three major areas: 1) definition and harmonization of requirements for software intensive systems at different safety classes, 2) integration of several approaches like SPICE (Software Process Improvement and Capability dEtermination) and FMECA (Failure Mode, Effects and Criticality Analysis method) to improve confidence in qualification and 3) integration of the system acquisition and qualification processes to improve total effectiveness of the acquisition, delivery and deployment processes.

The integrated qualification method is called TVO SWEP (SoftWare Evaluation Procedure). It consists of detailed qualification process and related methods for safety category B and C (IEC 61226) and Finnish safety class 3 qualifications. TVO will use the TVO SWEP method to evaluate suppliers and the conformance of their products/systems against requirements. It has been used in several cases, and it seems to save a lot of qualification resources compared to traditional methods.

## Keywords

Safety-Critical systems, Instrumentation and Control, qualification, SPICE, FMECA



### 1 Introduction

When the first versions of nuclear specific system and software standards were written some 20 – 25 years ago, no generic software and quality standards like ISO15504 or IEC/EN ISO61508 existed or were not at least commonly known. So, each party developed their own criteria and terminology for their own needs.

Quite typically, nuclear power instrumentation and control (I&C) systems are industrial products and are not designed and manufactured uniquely for each application. At least their platform is based on standard solutions and may be developed for many different purposes. Some subsystems may be old and not being able to qualify during the system delivery. The only evidence may be their historical development process and current operational history. They may have many versions and variants and new changes to come. Only some minor part of the whole delivered system may consist of customer-specific application. When the system will be delivered into the nuclear power unit, it may require platform oriented pre-qualification and application-oriented qualification during the delivery process. As a whole, complete qualification can be very time-consuming and expensive.

As a result of the described development, there is clear need for an integrated and effective method to qualify software intensive systems in nuclear power units. Integration has three major areas: 1) Definition and harmonisation of requirements for software intensive systems at their different safety levels 2) Integration of several approaches like SPICE and FMECA to improve confidence of qualification 3) Integration of the system acquisition and qualification processes to improve total effectiveness of the acquisition, delivery and deployment processes.

The objective of SPICE method is to evaluate the process capability. SPICE is a brand name for ISO15504 Process Assessment standard. The capability measurement system is based on ordinal 5-point capability level scale. Basically any process can be evaluated using the measurement system. In most cases, some predefined process reference model is used. Most known models are defined by ISO itself. ISO12207 is the standard for software life-cycle processes. ISO15288 is similar model for systems engineering. In most cases I&C systems for nuclear power plants are developed using a combination of software and systems engineering processes.

Failure mode effects and criticality analysis method (FMECA) has been used in some cases in Finland to bring evidence to the qualification process of safety-critical system. FMECA is effective to focus in most critical parts of the system, which have highest potential to cause failures. In hardware components, many well-defined methods can be used to show evidence about reliability and potential to failures. Redundancy can be used to reach required reliability and failure prevention level. For software-intensive components standard FMECA is less applicable, because software failure statistics is typically incomplete. Software reliability and probability of failures to occur may be difficult to predict, even to calculate. Software failures can be common failures, making them even more critical than separately occurring hardware failures.

TVO SWEP has been developed as a joint effort of TVO, Technical Research Centre VTT and Finnish Software Measurement Association FiSMA. TVO had the project management role, and it validated the method in real-life pilots. VTT is the primary research party and are responsible of the safety analysis method refinements. FiSMA is the national body responsible of software and systems engineering standards, including ISO12207, ISO15288 and ISO15504. FiSMA is responsible of modification and extension of SPICE to fulfill qualification requirements of safety-critical systems.

STUK (Säteilyturvallisuuskeskus, The Finnish Radiation and Nuclear Safety Authority) has defined four safety class levels for nuclear power unit (SC1 ... SC4, SC1 being the highest). IEC/EN61508 defines four safety integrity levels (SIL1 ... SIL 4, SIL4 being the highest). Some other standards have defined for example safety classes 1, 2 and 3 for systems, and safety categories A, B and C for functions. There is no clear mutual compatibility between various nuclear specific standards

and their safety classifications. Also criteria and requirements to validate achievement of the defined safety class can be different. National regulators as STUK want and even must to define their own requirements for the qualification process, to be able to carry out their monitoring and regulatory role.

## 2 Overview of the qualification process by SPICE and FMECA

The main phases of the qualification are **pre-qualification** and **application qualification**. SPICE is used mainly in the pre-qualification phase, together with relevant nuclear specific standards. If needed, also application qualification is done, partly with the same methods. As a starting point, preliminary hazard analysis (PHA) is done as part of the user requirements definition step. FMECA or HAZOP method is used after PHA, and is maintained and completed during all qualification steps. Figure 1 shows the main steps of qualification and how it is integrated to main steps of system acquisition and development.

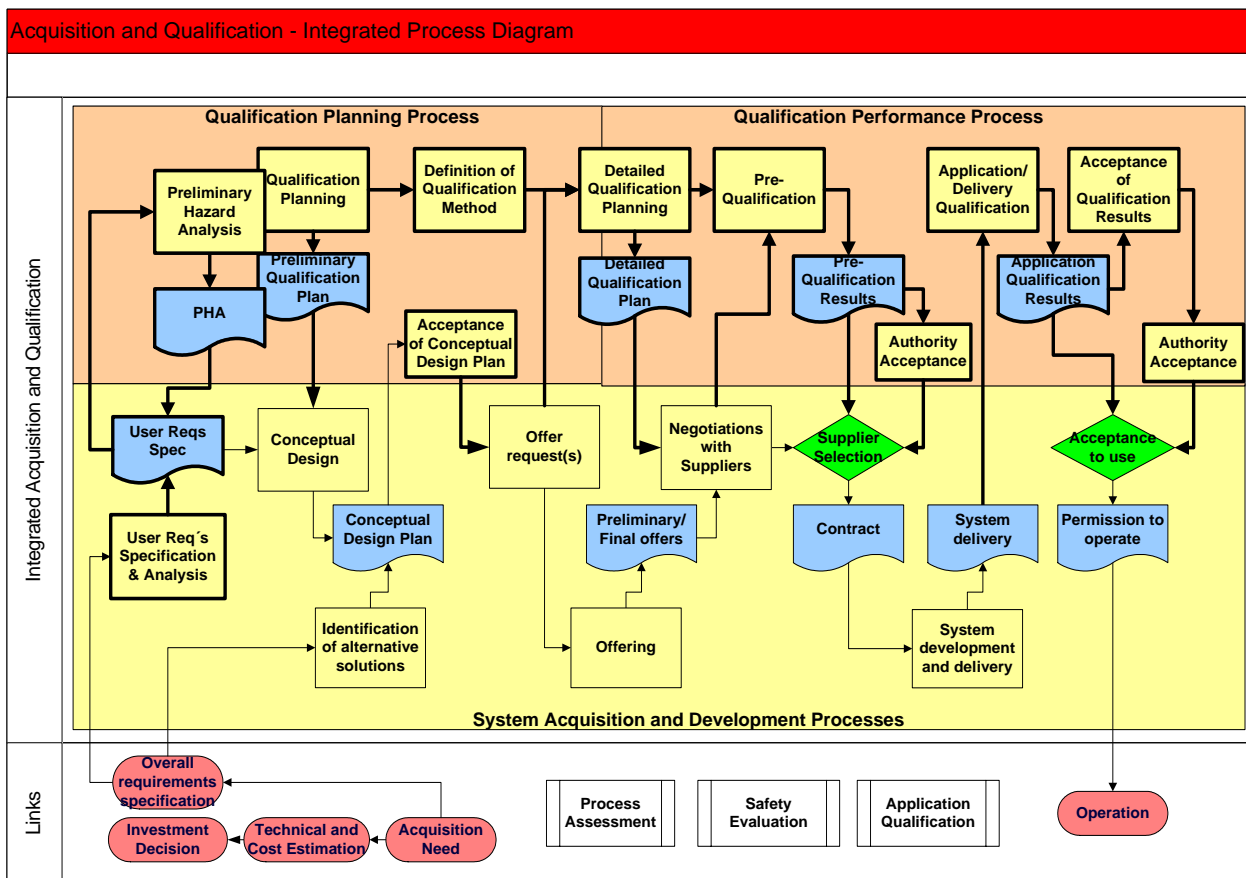


Figure 1. The qualification process, integrated with I&C system acquisition

As figure 1 illustrates, qualification is based on a detailed qualification plan. A typical input is PHA based on user requirements. It is very important to define safety requirements early in the acquisition process for each safety-related function. When that is defined, the detailed qualification plan and tailoring of questionnaires can be done according to requirements.

Typically, the qualification needs a lot of technical data from system suppliers. Therefore the pre-qualification phase and necessary negotiations with system suppliers is in parallel with qualification planning. The suppliers are informed about the qualification, and are prepared to participate if needed.

## Session 1: SPI and Assessments

Pre-qualification is meaningful to perform in full scale, if the system platform and the application are quite large systems and have typically several safety-related functions. For small systems some less effort-intensive methods are used if possible. The pre-qualification is mainly a combination of detailed and evolved PHA, process assessment and conformance checks against necessary nuclear specific standards. Necessary documents are reviewed as part of assessment. Also verification and validation of technical documents and their safety functions are an essential part of the pre-qualification. See figure 2 about the typical work flow of pre-qualifications.

Qualification during application and system development is done when needed. As a process, it is quite similar as the pre-qualification. In most cases it includes further checks of system and application details. Also some additional requirements may evolve from selected normative standards. They may be identified during pre-qualification, but need more attention and evidences. Some typical topics are control of tests and their coverage during application development, and handling of system changes for each application.

### 3 Process assessment elements of the method

ISO15504 Part 5 (known as the SPICE model) is used in as the main source of process assessment. The latest published ISO standard version ISO15504 Part 5 is used as the baseline. Part 5 has all ISO12207 processes and not all of them are relevant for qualification purposes. Many nuclear specific standards include quite similar concepts of processes as ISO15504 Part 5, and they are also used as normative sources. The primary sources are listed in table 1.

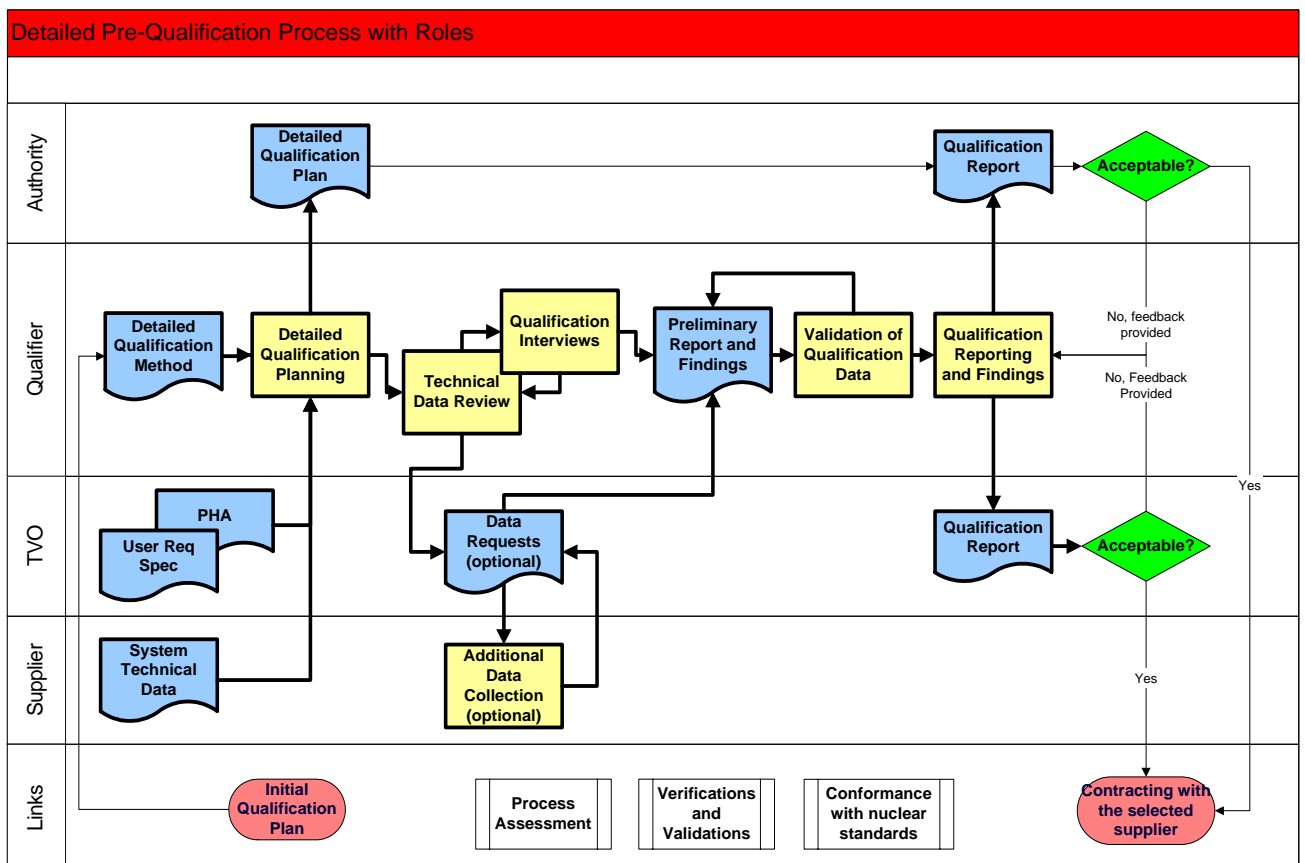


Figure 2. A typical work flow of pre-qualification

Table 1. Main normative sources for safety class 3 requirements

<b>YVL 5.5</b>	Instrumentation systems and components at nuclear facilities. STUK 2002.
<b>IEC 61513</b>	Nuclear power plants – Instrumentation and control for systems important to safety – General Requirements for Systems. 2001.
<b>IEC 62138</b>	Nuclear Power Plants Instrumentation and Control Computer-based systems important for safety Software for I&C systems of safety classes 2 and 3. 2001.
<b>ISO15504 Part 5</b>	An Exemplar Process Assessment Model. Published as an ISO/IEC standard in 2006.
<b>IAEA Safety Guide NS-G-1.1</b>	Software for Computer Based Systems Important to Safety in Nuclear Power Plants. 2000.
<b>IAEA 384</b>	Verification and Validation of Software Related to Nuclear Power Plant Instrumentation and Control. IAEA 1999.
<b>IEC/EN 61508</b>	Functional safety of electrical/electronic/ programmable electronic safety-related systems. 1998.
<b>Common Position</b>	European Nuclear regulators guideline for the licensing of safety critical software for nuclear reactors, EUR 19265, 2000.

The list of relevant SPICE processes selected for qualification is presented in table 2. Not all SPICE processes are as relevant as others, and also the cost-effectiveness of process assessment indicates rather a short than complete list. The criterion for process selection has been alignment and integration of ISO12207 processes and related nuclear specific standards in table 2.

**Table 2. List of SPICE processes used in qualification**

<b>Process</b>	<b>Name</b>	<b>Main areas of integration with nuclear specific standards</b>
ENG.1	Requirements elicitation	Detailed specification of safety functions and their SIL type according to PHA analysis results. Requirements for system testing.
ENG.2	System requirements analysis	Validation of each requirement, separate handling of safety requirements. Traceability.
ENG.3	System Architecture design	Allocation of each safety function. Overall architecture of the system. System validation planning.
ENG.4	Software requirements analysis	Specification and independent validation of each software function related to safety
ENG.5	Software design	Similarly as ENG.4. Planning of software verification tests.
ENG.6	Software construction	Module testing and documentation. Avoidance of unnecessary code.
ENG.7	Software integration	Test records. Validation of integration test results.
ENG.8	Software testing	Test records. Validation of software testing results.
ENG.9	System integration	Test records. Validation of system integration test results.
ENG.10	System testing	Test records. Validation of system test results.
ENG.11	Software installation	Installation test. Correct technical environment.
SUP:1	Quality assurance	Quality planning. Reviews and inspections at project level.
SUP.2	Verification	Independent tests and technical reviews.
SUP.3	Validation	Independent FAT and SAT tests.
SUP.7	Documentation	Done according to supplier's process and safety requirements.
SUP:8	Configuration management	Full traceability. Change control.
SUP:9	Problem resolution management	Full audit trail. Analysis of each defect and it's impacts. Common causes of failures.
SUP.10	Change request management	Full change records. Analysis of each change.
MAN.3	Project management	Quality planning. Verification and validation planning.
MAN.4	Quality management	Quality management activities according to supplier's process.
MAN.5	Risk management	Avoidance of product related risks.
MAN.6	Measurement	Measurement-based testing and validation, if possible.

Each process is assessed up to capability level 3, if possible. Level 3 is considered as the highest required capability level, because only standard processes and an organisation-wide quality system is required for safety class 3 systems. Some processes need a lot of refinements and elaborations to comply with safety-critical system context, and that is done as part of integration of SPICE and nuclear specific standards. In most cases, an interpretation of each SPICE element is not enough, also extension of the processes with additional practices or alternative checklists are needed.

The result of SPICE assessment is a capability level for each process and a number of evidences. They can be used as “load evidence” for more detailed safety analysis. SPICE capability level is not always the best way to express the real capability of each process. Therefore also an “capability index” is calculated as a ratio of evaluated practices and their sum compared to target level of the process.

Conformance against nuclear specific standards and their safety requirements is done mainly in parallel with SPICE assessment. Most requirements are used as interpretation rules of base and generic practices of each SPICE process. Also complementary methods and evaluations are needed, especially in software and system validation.

### 4 Safety evaluation elements of the method

The starting point for safety analysis is PHA (Preliminary Hazard Analysis). It defines the need for evidences to achieve required detection level for potential failures. In FMECA (Failure Mode Effects and Criticality Analysis), three main components of each potential failure are severity, occurrence and detection.

Based on original safety requirements, each potential failure is classified according to its severity, using 1 – 5 scale. Similarly, also the probability of occurrence and required detection rate are classified 1 – 5. SIL levels from IEC/EN61508 are used as a reference to map each safety function and its required detection rate. As a result, these factors are multiplied into APN number (Action Priority Number). The calculated APN for each potential failure indicates, how much evidence is needed to achieve acceptable level of failure detection and so the APN as a whole. Figures 3 and 4 explain in more details most important method features.

SOD-tables for safety class 3 13.3.2004						Severity of safety effect				
						Severe radiation damage (several persons)	Radiation damage (one person)	Radiation exposition	Fuel damage	No impact on safety
Severity										
Occurrence										
Detection										
Occurrence and detection table for a I&C system										
Probability of failure		SIL	Occ Det	Det	Rank	5	4	3	2	1
Continuous	Low demand		Reduct.	%	Target					
$10^{-4} \dots 10^{-5}$	$\dots 10^{-1}$		0	0	5					
$10^{-5} \dots 10^{-6}$	$10^{-1} \dots 10^{-2}$	SIL 1	-1	60	4					
$10^{-6} \dots 10^{-7}$	$10^{-2} \dots 10^{-3}$	SIL 2	-2	75	3					
$10^{-7} \dots 10^{-8}$	$10^{-3} \dots 10^{-4}$	SIL 3	-3	90	2					
$10^{-8} \dots 10^{-9}$	$10^{-4} \dots 10^{-5}$	SIL 4	-4	99	1					
Potential consequence of a Safety Function					Rank	5	4	3	2	1
Loss of main mission of a SF (latent faults, no fail safe)					5	5	4	3	2	1
Minor loss of a SF					4	4	3	2	1	
Degraded performance of a SF					3	3	2	1		
Fail safe operation					2	2	1			
No appreciable effect					1	1				

Figure 3. Integration of severity, occurrence and detection rate of each potential failure

Det column in Figure 3 shows minimum values for detection rate reduction. For example to achieve SIL 1 you have to achieve at least 60 % detection rate. See figure 5 for more details how to count detection rate.

Figure 4 shows two selected failure modes as an example. The system was in this case the gamma wagon. It is used to lift fuel bars up and down in the reactor pool, for example to maintain the fuel bar.

Evaluation of Functional Description of the Gamma Wagon							Existing conditions						Existing conditions			
Identity: P 03-044, rev.2																
1 Ref.	2 Entry code	3 Potential failure mode (FM)	4 Effect on	5 Potential effect (E)	6 Potential cause (C)	7 Risk controls (RC)	8 Sev	9 Occ	10 Det	11 APN	12 Recommended action	13 Action taken	14 Sev	15 Occ	16 Det	17 APN
General	0	The irradiated fuel is too near the water surface of the pool	Safety	Danger of radiation for people at the pool	1. Operator drives the wagon too near the surface 2. Risk Controls are failed	1. Failure detection by operator at the pool, DET-1 2. Manual emergency stop, OCC-½ 3. Hardwired emergency stop, OCC-1 4. Mechanical stop, OCC-1 5. Position alarm of the mechanical stop, DET-½ 6. Brake system 7. Radiation alarm, DET-½ 8. Lock keys	5	2,5	3	37,5	1. Position detection of the mechanical stop 2. Radiation protection 3. Programmable safety stop 1-3: OCC-1	For further SW error detection evaluation, DET-1	5	1,5	2	15
2.1.1.1 2.1.2.2	1	Programmable safety stop (Entry Code 1-7/8) function does not be actuated in mode A	Entry code 0-RC8	The movement of the wagon will continue rising	1. Electro mechanical limit switches are failed 2. Inputs fail 3. Logic fails 4. Outputs fail 5. Control unit fails	EC 0-7 RC	5	2,5	3	37,5	V&V for SW	For further SW error detection evaluation: SW causes 2-4	5	1,5	2	15

Figure 4. Use of severity, occurrence and detection to calculate APN. An example with two potential failure modes is presented just for illustration.

### 5 Evidence collection to the Detection table

Figure 5 is a sample from a so called detection table of the TVO SWEP method. It summarises qualification findings in a composite index called “Total detection number DET”. The idea is to use DET index as load or backing evidence in FMECA sheet (Column 10 – Column 16 in Figure 4). The table is separate for Pre-Qualification and Qualification phases. In most cases it is also separate for Platform and Application. Evidence in each phase is gathered by process evaluation (SPICE) and product safety evaluation (FMECA). Also compliance with nuclear specific standards is taken into account here.

SPICE capability level is converted to so called capability index, which summarises detailed practice ratings at levels 1 – 3 for each process. It is normalised to get values between 0 – 1 for each capability level. Other evidences are detailed requirements from V&V processes, as defined in IAEA report no. 384.

In the example of Figure 5 we present two columns for DET and its inputs. The right column is for the pre-qualification phase and the left column for the additional nuclear specific verification of the application. The pre-qualified system was in this case a radiation monitoring equipment. As seen in the example, also NA (Not Applicable) rating is allowed and used, if relevant input data or rating result is not available.

In most cases the Severity and Occurrence parameters remain the same and only failure detection rate can be improved. As a result, Action Priority Number APN may reach an acceptable level. In our example (see columns 11 and 17 in figure 4) the goal was that each potential failure has APN value 25 or less. In our example that is the case, and the qualification has been successful.

## Session 1: SPI and Assessments

Finally, the aim is to determine the Reduction Detection Number RDN. RDN is typically a difference between DET number at FMECA table (see columns 10 and 16 in Figure 4 as an example). RDN can get a value 0 – 4. Detection table calculates the value of RDN automatically, based on the SPICE and V&V evidences. Calculated RND value is used in FMECA table to reduce Detection rate.

Verification of SW change process		0.33	0.20
SPICE	Pre-Q of the process, CFG.3 Problem resolution management	NA	NA
SPICE	Pre-Q of the process, CFG.4 Change request management	NA	NA
r	Review of impact of change analysis (Walk or Insp)	0.66	NA
r	Review of life cycle tasks (Walk or Insp)	1	1
r	Review of non-regression analysis (Walk or Insp)	NA	NA
o	Review of impact of change analysis (FMECA)		
o	Review of impact of change analysis (FTA)		
Configuration and quality		0.57	0.57
SPICE	Pre-Q of the process, CFG.1 Documentation	1	1
SPICE	Pre-Q of the process, CFG.2 Configuration management	0.67	0.67
SPICE	Pre-Q of the process, QUA.1 Quality assurance	NA	NA
SPICE	Pre-Q of the process, QUA.2 Verification	0.61	0.61
SPICE	Pre-Q of the process, QUA.3 Validation	0.36	0.36
Management		0.24	0.24
SPICE	Pre-Q of the process, MAN.3 Project management	0.94	0.94
SPICE	Pre-Q of the process, MAN.4 Quality management	NA	NA
SPICE	Pre-Q of the process, MAN.5 Risk management	NA	NA
SPICE	Pre-Q of the process, MAN.6 Measurement	NA	NA
Operating experience		0.33	0.33
	Collected with a well specified method	NA	NA
	Operational profiles	NA	NA
	Representative for the application	NA	NA
	HW & SW versions	NA	NA
	Long collection time	1	1
	Results of experience	1	1

<b>DET index</b>	0.689	0.68
DET after SW requirements phase	0.64	0.59
DET after SW coding phase	0.73	0.62
DET after I&C integration phase	0.772	0.679
DET after FAT	0.767	0.688
DET after SAT	0.759	0.691

Figure 5. A sample from an Excel based checklist to calculate detection index DET. An example is presented just for illustration.

## 6 Conclusions and future developments

Methods and techniques in TVO SWEP consist of process safety evaluation, and product safety evaluation. These same methods and techniques are used by the second or third party inspectors, who assess explicitly or/and implicitly the artefacts or processes.

Several real-life qualifications are already done by using TVO SWEP method. The goals of the method are achieved well, and the pre-qualification is effective. It is evident that TVO SWEP needs still refinements and additional validation. Main difficulty is to collect evidences so systematically, that the necessary calculations and reports could be done as automatically as possible. Then the main

focus can be in professional topics and may lead to useful win-win findings between TVO and the supplier.

Other industries may have quite similar needs for qualification as nuclear power plants. For example control of railway and metro networks and traffic, electro medical devices like patient control systems and many military systems could be users of our method. In fact, any business sector and company who has pre-defined safety requirements and can express it by using SIL levels could be potential user.

Method is implemented partially in FiSMA Assessment System GNOSIS and in Excel sheets. Also paper-based version is available. It is quite possible that the method will have several technical implementations in the future.

## 7 References

1. **YVL 5.5**, Instrumentation systems and components at nuclear facilities. STUK 2002. **IEC 61513**, Nuclear power plants – Instrumentation and control for systems important to safety – General Requirements for Systems. 2001. **IEC 62138**, Nuclear Power Plants Instrumentation and Control Computer-based systems important for safety Software for I&C systems of safety classes 2 and 3. 2001.
4. **ISO15504 Part 5**, An Exemplar Process Assessment Model. 2006.
5. **Safety Guide**, Software for Computer Based Systems Important to Safety in Nuclear Power Plants. 2000.
6. **IAEA 384**, Verification and Validation of Software Related to Nuclear Power Plant Instrumentation and Control. 1999.
7. **IEC/EN 61508**, Functional safety of electrical/electronic/ programmable electronic safety-related systems. 1998.
8. **Common Position** of European Nuclear regulators for the licensing of safety critical software for nuclear reactors, EUR 19265, 2000.
9. **Harju, H.** Ohjelmiston luotettavuuden kvalitatiivinen arviointi (The qualitative assessment of software dependability). VTT Technical Research Centre of Finland. VTT Research Notes 2066. 2000 (In Finnish).

## 8 Author CVs

### Juha Halminen

Automation engineer in TVO

Mr. Juha Halminen has 5 years experience in nuclear equipment quality topics. His working experience includes qualification of new equipment with or without software to nuclear power plant safety systems. He has been the key person in developing software qualification procedures for TVO. Before that he was participated in business development work in Russia. During 1994-1995 he was researching influence of contamination and humidity to building stones, concrete and mortar in IBW Germany.

### Risto Nevalainen

Director of FiSMA and STTF Oy

Mr. Risto Nevalainen (Lic. Tech.) has long experience in software measurement and quality topics. His working experience include position as managing director of Finnish Information Technology Development Center during 1989-1995. Before that he had different research and management positions for example in Technical Research Centre (VTT), Technical University of Helsinki (HUT), Finnish Prime Minister's Office and Finnish Economic Planning Centre. Mr. Nevalainen has participated in ISO15504 (SPICE) standard development since beginning. He is Competent SPICE Assessor and ISO9000 Lead Assessor.





# A SPICE-based Software Supplier Qualification Mechanism in Automotive

*Fabrizio Fabbrini, Mario Fusani, Giuseppe Lami, Edoardo Sivera*

## **Abstract**

*Car makers depend more and more on the software they acquire, that forced them to address the issue of assessing the quality of the acquired software. European Car Makers approach this issue by checking the capability of the suppliers' software process. FIAT Auto, since year 2001, has been taking pro-active actions to address the situation. The basic strategy has been to set up criteria for qualifying software suppliers based on measuring the capability of their software process. In this paper, this qualification mechanism is described in detail. The innovative aspect of the approach is to consider the results of a SPICE (ISO/IEC 15504) assessment as one of the contractual requirements to let a software company be eligible for software supplies.*

*As a positive effect, the assessments made on the software process allow FIAT Auto to select suppliers on the basis of some objective and quantitative evidence of their level of quality. In addition, this initiative has been giving FIAT a deeper knowledge of the way its suppliers produce software: this is a fundamental step to building up the capability of monitoring and driving the software projects in a collaborative effort with the suppliers, which can benefit from having possible improvement opportunities pointed out.*

## **Keywords**

Supplier Qualification, Process Assessment, Software Acquisition Process

### 1 Introduction

A few years ago, Information Technology made its first appearance in automobile functions control. Since then, technical solutions for Electronic Control Units have run again through all the stages of the longer evolution of computer systems. Doubtfully accepted at first in a community of system engineers, in the form of scattered, stand-alone program-control based devices, they have boom-evolved into crossed, integrated networks of tens and tens of active nodes, each one performing complex, often time-constrained functions [1, 2, 3]. Continuous market pressure for all-in-a-car integrated services (including engine control, x-by-wire drive, passenger comfort control, web-link and many others) not only claims for deployment of the most advanced system and software engineering practices but is also feeding the most daring technological research challenges. Automotive issues have now become a leading applicative domain for experimenting advances in the broader realm of software and system engineering.

The traditional reliance on Quality Systems Standards such as ISO9001, QS9000 [9] and ISO/TS 16949 [10] has not provided sufficient confidence in the software area. The motor vehicle manufacturers, like others in the defence and aerospace industries, have now turned to international standards for software process assessment, based on ISO 15504 (known also as SPICE) [4] and/or the Capability Maturity Model (CMM) [12], as a mean to identify and control risks and to assess the software capability of suppliers [5], [6], [7]. A common trend in the European automotive industry to face this challenge consists of principally addressing the improvement of the software acquisition process. While different car makers set up their own improvement program, a commonly adopted criterion to assess the capability of the suppliers' software processes is the choice of the SPICE model as the principal mean [8]. In year 2001 an initiative was launched by the Procurement Forum with the principal European Car Makers, their assessors and representative bodies to address the problems related to software assessments in automotive. In the framework of this initiative, a Special Interest Group (SIG) has been founded with the aim to design a special version of the SPICE model (called Automotive-SPICE) tailored on the needs and peculiarities of the automotive business area [13]. FIAT Auto and the System & Software Evaluation Centre at the C.N.R. are part of this Special Interest Group. The first results of the initiative was to create consensus on commonality of approach in order to avoid that suppliers face multiple assessments from multiple manufacturers using different models and criteria and consume resources that put additional pressure on delivery times. The focus on software capability determination by means of software process assessment has already provided significant business benefits, but at the same time has highlighted the scale of the potential problem, particularly with suppliers of safety-critical embedded software system components.

The outcomes of the SIG are the definition of a special Software Process Model and Process Assessment Model for automotive. The expected short term result is a general transition of the European Automotive Industry to use of the Automotive SPICE Process Assessment Model. During this transition phase both process assessment models (SPICE and Automotive SPICE) will be used in parallel within the Automotive Industry depending on the purpose and scope of an assessment. From early 2007 it is expected that assessments based on the ISO-TR 15504 (a former consolidated version of the SPICE process reference and assessment model) process assessment model will no longer be used.

The purpose of this paper is to describe the ESCAPE project, a FIAT Auto initiative for improving its own software acquisition process, as well as to discuss the benefits produced by this project both for FIAT Auto and for the involved software suppliers. This paper is structured as follows: in section 2. the ESCAPE project is described in detail; in section 3. the benefits and the consequences of the project are discussed, in section 4 the next steps of the project are identified and finally in section 5 the conclusions are provided.

## 2 The ESCAPE Project

The growth of software-intensive systems in the vehicles has determined new challenges in the automotive industry, forcing FIAT Auto to change its approach to the relations with suppliers in order to be able to understand, monitor and, if case, drive the development process of the acquired software intensive systems.

Fiat Auto started ESCAPE (Electronics Software CAPability Evaluation) in year 2001 with the aim of improving its own software suppliers selection process and supporting the management of software projects and suppliers. This project is conducted in co-operation with the System and Software Evaluation Centre (SSEC - a group of the Italian National Research Council that performs independent evaluation and certification activity in Information Technology). The ESCAPE project is composed of two phases. In the following, these two phases are described and the related results discussed.

### 2.1 Background: ESCAPE Project Phase A

The phase A was conducted in years 2001-2004 and was based on a pilot set of software process assessments made according to the SPICE methodology. These assessments, having a common scope, were made to a set of 17 Fiat Auto software suppliers. The assessment scope was composed of five processes, selected by Fiat Auto, all of them to be assessed up to capability level 4. The processes included into the initial assessment scope was: Requirements Elicitation, System Requirements Analysis and Design, Software Design, System Integration and Testing, and Project Management processes.

The aim of the phase A was to achieve a complete and detailed picture of the capability of software suppliers for the processes of interest for identifying weaknesses and strengths. The detailed procedure as well as the results of these assessments are reported in previous papers [11, 14]. The most important feed-back obtained by FIAT Auto from the phase A has been the achievement of a deeper knowledge of the software development process of its principal software suppliers.

Although the main interest of FIAT Auto was focused on the engineering processes, we realized that including additional managerial processes into the scope can increase the confidence on the assessment results. In fact, assessing the performance of managerial processes allows to cross-verify some relevant practices necessary for achieving the levels 2 and 3 of the engineering processes. Because these considerations the initial set of processes in the assessment scope has been enlarged at the end of the phase A. The resulting set of processes is described in Table 1.

A specific target capability level was assigned to each process in the new scope. To determine a process capability level the inherent nature of the process itself has been considered as well as the objectives of FIAT Auto. The ultimate FIAT Auto's objective and demand is to monitor the engineering processes of its suppliers in order to get knowledge of the process adopted for developing software; nevertheless directly monitoring these processes may require a very high effort and conflict with the privacy policy of the supplier. From the FIAT Auto point of view, finding the supplier process at SPICE capability level 3 means to get the confidence that a defined process capable of achieving its process outcomes is used to implement, in a managed fashion (planned, monitored and possibly adjusted), the process and appropriately establish, control and maintain its work products. For these reasons, to require level 3 for the suppliers' engineering processes, means to address the initial demand by following a different way than the direct, on site monitoring of the suppliers software development process. The Requirements Elicitation process (ENG.1), having target capability profile 2, is an exception to this scheme. The reason is because such an engineering process depends not only of the supplier process but also of the customer one, then the capability level 2 (i.e. having a managed process) has been considered high enough because the need to have a defined process at the supplier side has not the same relevance than the other engineering processes.

The target capability level of the managerial supporting processes included into the assessment scope (MAN.3 and MAN.5) were set to level 1 (level 2 in the case of the SUP.8 process) because assessing them can corroborate and sometimes simplify the assessment of those process having target capabil-

ity level 3.

acronym	Process name	Process purpose
ENG.1	Requirements Elicitation	to gather, process, and track evolving customer needs and requirements throughout the life of the product and/or service so as to establish a requirements baseline that serves as the basis for defining the needed work products.
ENG.2	System Requirements Analysis	to transform the defined customer requirements into a set of desired system technical requirements that will guide the design of the system.
ENG.5	Software Design	to define a design for the software that implements the requirements and can be tested against them.
ENG.8	Software Testing	to confirm that the integrated software product meets its defined requirements.
ENG.10	System Testing	to ensure that the implementation of each system requirement is tested for compliance and that the system is ready for delivery.
MAN.3	Project Management	to identify, establish, plan, coordinate, and monitor the activities, tasks, and resources necessary for a project to produce a product and/or service, in the context of the project's requirements and constraints.
MAN.5	Risk Management	to identify, manage and mitigate the risks continuously, at both the organizational and project level.
SUP.8	Configuration Management	to establish and maintain the integrity of all the work products of a process or project and make them available to concerned parties.

**Table 1: Processes in the assessment scope**

## 2.2 ESCAPE Project Phase B

The purpose of the ESCAPE Project's phase B is to set up and apply a supplier's qualification mechanism. Such a mechanism is based on the achievement of a defined target capability profile. The success of the ESCAPE project's phase A leads Fiat Auto to integrate the software process assessment based on the SPICE model into the qualification mechanism.

The Fiat Auto suppliers qualification mechanism is based on the expected capability profile produced in the Phase A of the ESCAPE project. In fact, basically, it is required the suppliers to match the capability profile in order to be able to participate in the Fiat Auto's sourcing phase (the sourcing phase is the period in which Fiat Auto select the right suppliers for a specific project). The suppliers that do not match that profile cannot participate at the sourcing phase of a specific project.

Such a mechanism, yet quite rigid, was mitigated by adding the possibility for a company, that is undertaking an improvement program, to participate in the Fiat Auto supplies also if its capability profile is still under to expected one. In particular, for some specific process in the scope, a lower capability level is considered temporary acceptable, under the condition that some process improvements are made in a predefined period of time.

For this purpose, a minimum capability profile was defined. Such a capability profile as well as the expected one are described in Figure 1.

In the case that a company, after the assessment, matches the minimum but not the target capability profile, the qualification mechanism requires that an improvement program is to be set up by the software supplier and a further assessment to verify the improvement is planned and agreed by Fiat Auto. In the mean time between the two assessments, the company is conditionally admitted to the FIAT Auto's sourcing phase. If the second assessment will report the achievement of the target capability profile, the company will be qualified, else it will be not.

The validity of an assessment is three years (36 months).

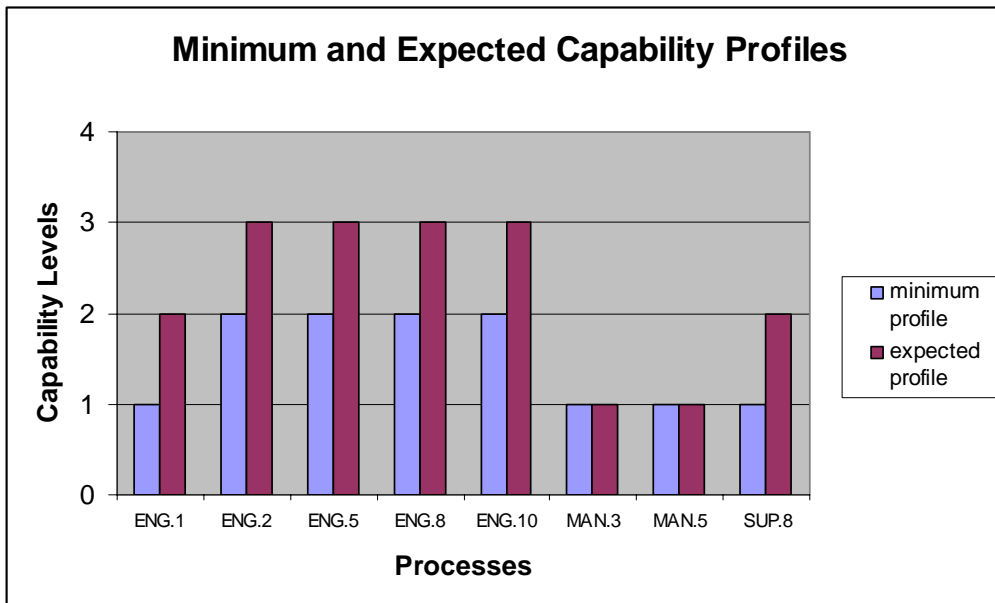


Figure 1. Minimum and expected capability profile.

The current status of phase B is: six supplier qualifications have been made; all of them satisfying the minimum capability profile but at the same time needing to improve some processes to achieve the expected profile. Currently five companies are still going to deploy the improvement plan agreed with FIAT Auto, one company completed the improvement plan and achieved the expected capability profile after an additional assessment.

### 3 Consequences and Benefits

In this section the benefits achieved both by Fiat Auto and its software suppliers with the ESCAPE project are discussed. The principal improvements the ESCAPE project induced in the Fiat Auto process are summarized in the following:

*Sourcing phase:* a relevant benefit for Fiat Auto is the positive impact on the sourcing phase of a specific project. The sourcing phase is important for a project because at that phase the Suppliers of all electronic components of a vehicle are selected. Consequently, the electronic product quality may depend on this phase. The SPICE assessments are effective means to define practical selection criteria because they provide objective, pertinent and unbiased measurements. Fiat Auto is using the SPICE assessment results in order to increase the trust on the software suppliers capability and, then, final product quality.

*Monitoring of the acquisition process:* Another important benefit the ESCAPE project produced is the capability of better monitoring and managing the acquisition process. In fact, Fiat Auto is using the knowledge of the software process of each supplier as well as their capability profile to manage the monitoring of the Supplier process software development. The Design Review Process is a process used in Fiat Auto to monitor its project development. This process is aimed at verify that all the expected advancements (both hardware and software) in the project are in line with the plan. Knowledge of the software suppliers process allows Fiat Auto to synchronise its own Design Review with the development milestones of the software supplies. That improves the capability of Fiat Auto of controlling its suppliers.

*Improvement areas identification:* The SPICE assessment is an effective mean to highlight improvements opportunities in the software process. Often these results are able to highlight weak points in the organizations: generally Fiat Auto starts from these measurements to agree with its own Suppliers some process improvement activities or to define some internal improvement. Moreover, some improvements areas identified in the suppliers process may affect also the Fiat Auto process. For in-

stance, possible weaknesses in the Requirements Elicitation process of the supplier can be due partially to some weaknesses from the Fiat Auto side.

*Communication facilitator.* During a supply, the customer and the supplier shall establish a communication channel to exchange information on the status and the technical aspect of the product to be developed. Often, there are communication barriers due to the different cultural, and environmental conditions of the two partners. A relevant benefit derived from the ESCAPE project concerns the definition of a “common language” between Fiat Auto and its own Suppliers. In other words, using the SPICE definitions of process, work products, process attributes, etc. is a mean to establish a common language in the communication between Fiat Auto and its suppliers.

The benefits obtained by the FIAT Auto’s suppliers can be summarized as follows:

*Improvement accelerator for Suppliers:* The general European trends towards the software process improvement of the automotive industry, witnessed by the Automotive SPICE initiative, have demonstrated that the Fiat Auto strategy, based on the software process assessment using the SPICE model, is shared by many other manufacturers. That gives advantage also for the automotive software suppliers. In particular, the ESCAPE project indirectly impacts also on the Fiat Auto’s Supplier because it has been an important stimulus to accelerate the software process improvements of several software suppliers. Moreover, Suppliers can use the Fiat Auto approach as a sort of benchmark where the required capability profile becomes a target to be aligned with the competitors.

### 4 Next Steps

The next steps of the ESCAPE project are aimed at continuing the path towards the improvement of the Fiat Auto sourcing and acquisition processes that has been triggered in year 2001 with the start of the project. In particular, the current expected capability profile in the FIAT auto supplier qualification mechanism will be the minimum requested profile within year 2008. That could represent a stimulus for a continuous improvement for the suppliers. Moreover, we aim at integrating the specific Fiat Auto supplier qualification mechanism into the Automotive SPICE initiative whether it will evolve as planned. In fact, the objective is to set up a common required capability profile for all the European automotive industries participating in this initiative. That would give relevant advantages for both all the car makers and their software suppliers.

### 5 Conclusions

In this paper we described the ESCAPE project, a Fiat Auto initiative, aimed at improving its own sourcing and acquisition process, by including the achievement of a required capability profile within the requirements a software company shall satisfy to be eligible for a Fiat Auto’s supply. We outlined the advantages that this project gave both to FIAT Auto and its software suppliers. These advantages not only can be quantified as an increased degree of satisfaction for the quality of the acquired software product, but include some positive “side effects” produced by the software process assessments made following the SPICE methodology. These positive effects can be summarized as:

- better supplier selection (only supplier having a high capability profile can be selected);
- better project monitoring (Fiat Auto is now able to identify the principal phases and work products to be controlled during the supplier’s software development process);
- better relationship with own Suppliers (clearer than before, because based on a deeper knowledge of the suppliers organization and processes);
- identification of internal improvement areas addressing specific processes (e.g. Requirement Management, Testing Management, etc.) and work products.

The ESCAPE project, as well as the European initiative called Automotive SPICE, determined practical benefits also for the software suppliers.

## Literature

- [1] Leen G., Hefferman D., Dunne A., "Digital Networks in the Automotive Vehicle", IEE Computer and Control Eng. Journal, Dec. 1999, pp. 257-266.
- [2] Kassakian J. G., "Automotive Electrical Systems: The Power Electronics Market of the Future", Proc. Applied Power Electronics Conference (APEC2000), IEEE Press, 2000, pp. 3-9.
- [3] "Development Guidelines For Vehicle based Software", The Motor Industry Software Reliability Association, 1994. Published by MIRA. ISBN 0952415607.
- [4] ISO/IEC 15504 International Standard "Information Technology – Software Process Assessment: Part 1–Part 5". 2006.
- [5] Paulk M., "Top-Level Standards Map: ISO 12207, ISO 15504 (Jan 1998 TR) Software CMM v1.1 and v2", Draft C (available at <http://www.sei.cmu.edu/pub/cmm/Misc/standards-map.pdf>), Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, 1998.
- [6] Hailey V., "A comparison of ISO 9001 and the SPICE framework". In "SPICE: The Theory and Practice of Software Process Improvement and Capability Determination", El-Emam K, Drouin JN, Melo W (eds). IEEE CS Press, 1998.
- [7] Jung H-W, Hunter R., "The relationship between ISO/IEC 15504 process capability levels, ISO 9001 certification and organization size: an empirical study", Journal of Systems and Software, n. 59, 2001, pp. 43–55.
- [8] El-Emam K., Jung H-W, "An evaluation of the ISO/IEC 15504 assessment model", Journal of Systems and Software, n. 49, 2001, pp. 23-41.
- [9] Quality System Requirements (QS-9000 Third Edition) Version 03.00, DaimlerChrysler, Ford Motor Company and General Motors Quality Publications, March 1998.
- [10] ISO/TS 16949- Quality Management Systems - Automotive Suppliers - Particular Requirements for the Application of ISO 9001:2000 for Automotive Production and Relevant Service Part Organizations, 2002.
- [11] F.Fabbrini, M.Fusani, G.Lami, E.Sivera "Performing Process Assessment to Improve the Supplier Selection Process - An Experience in Automotive" European Software Process Improvement Conference EuroSPI2002, Nuremberg, Germany, 18-20 September 2002. p.267-274.
- [12] Chrissis M.B, Konrad M., Shrum S. „CMMI Guidelines for Process Integration and Product Improvement". Addison-Wesley, 2004.
- [13] <http://www.automotivespice.com>
- [14] Fabbrini F., Fusani M., Lami G., Sivera E. "Software Process Assessment as a Mean to Improve the Acquisition Process of an Automotive Manufacturer". In "Software Process Improvement CMM&SPICE in Practice" Book, Verlag UNI-DRUCK Ed. ,Munchen, Germany, 2002, pp. 142-154.



### 6 Author CVs

#### **Fabrizio Fabbrini**

He obtained his degree in Computer Science from the University of Pisa, Italy, in 1974. Since 1975 he has served as a scientific researcher at the Institute for Information Processing (IEI) of the Italian National Research Council (CNR), where now he is Senior Researcher and coordinates the Software Laboratory of the Centre for Software Certification. Fabrizio Fabbrini's present activity is focused on Software Quality, and more precisely on the development of methodologies and standards for the assessment and the evaluation of software products and processes, with particular attention to Software Engineering Standards and Software Certification. Software Process Assessment & Improvement, Software Verification & Validation, Computer Security & Data Privacy represent the main fields of application of such research activities.

#### **Mario Fusani**

He obtained his degree in Electrical engineering from the University of Pisa, Italy, in 1971. Since 1973 he has served as a scientific researcher at the Institute for Information Processing (IEI) of the National Research Council (CNR), where now he is Senior Research. His present activity is focused on Software Quality, including the development of methodologies and standards for the assessment and the evaluation of software products and processes. Since 1999 he has been the Scientific Coordinator of the Centre for Software Certification of the Italian National Research Council.

#### **Giuseppe Lami**

He graduated in Computer Science in 1994 and get the Ph.d. in Information Engineering in 2003 at the University of Pisa. Since year 2000 he is a researcher at the Istituto di Scienza e Tecnologie dell'Informazione del CNR in Pisa and since year 2003 he is a resident affiliate of the Software Engineering Institute of the Carnegie Mellon University in Pittsburgh, PA (USA). His professional activity is focused on software product quality evaluation and certification and software process assessments (using the ISO 15504 standard model). His research interests mainly address requirements engineering and software product lines. He has experience of teaching university courses on software engineering and software quality.

#### **Edoardo Sivera**

Electronic Engineer (1990 – Politecnico di Torino). He worked (1991-1994) at the Fiat Research Centre analysing and using tools based on formal models for the definition of automotive functional requirements. Since 1994 he has worked in Fiat Auto, collaborating to the development of the architecture of integrated electronic systems based on serial networks (CAN protocol). Since 1998, he has worked on the definition of standards for the communication on serial networks (based on CAN protocol), used in all Fiat recent vehicles. Since 2000, he has been the responsible of the Fiat Auto "Standard and Methodology Software" group. The main objectives are: to manage the development of new equipments containing software; to define a software architecture for these electronic equipments; to develop and use methodologies to analyse and validate the automotive functional requirements.

# Assessment Based Learning Systems – Learning from Best Projects

Richard MESSNARZ<sup>1</sup>, Damjan Ekert<sup>1</sup>

<sup>1</sup>*ISCN GesmbH, Schieszstattgasse 4, A-8010 Graz, Austria*

*Tel: +43 316 811198, Fax: + 43 316 811312, Email: [rmess@iscn.com](mailto:rmess@iscn.com)*

## Abstract

In most cases assessments are purely used to determine a capability level profile of a set of projects and to derive improvement actions for an organisational unit. Since 2003 a group of major firms (sizes from 200 up to 90000 staff) started using a knowledge based assessment portal system in which the assessment results of all projects and divisions are stored in a central multi-user portal system. Here they also store all improvement findings of all assessors in all assessments and it is possible to extract improvement recommendations from a large set of data across the enterprise and to compare projects (which projects carry synergy potentials of a re-usable knowledge).

Since 2005 expert linking and team learning facilities have been added, so that projects which have weak areas can find expert projects (who were strong in this area) inside the corporate firm. Also it is possible to use team learning and training portals so that if a project is weak in a certain area it can enter a virtual training room for upgrading the skills in this area.

This approach is called “Assessment Based Learning Systems”.

In this paper this assessment based learning approach is introduced, where two systems, the Capability Adviser Web Assessment Tool and the Learning Management System Moodle are combined to support the improvement actions in a corporate environment supported by reference projects (best in the area and re-usable know-how) and an online training and tutoring learning cycle.

## Keywords

Process Improvement, Team Learning, Exploitation of Synergy Potentials, Web Based Learning Environment

## 1 Assessment Based Learning Centres Approach

A formal project assessment usually results in capability level profile [1]. It is typical that strengths (high levels) and weaknesses (low levels) are represented in such a profile. Some processes may be already on a higher level, while some processes might show significant gaps.

The assessment based learning centres approach (see Figure 1) [2],[3],[4] is based on the improvement of these gaps by exploiting synergies [5] in the corporate firm through learning from reference projects (expert-links), in-house trainings and tutoring from improvement projects. A reference project is a company project (can be from the same division or another worldwide location) where the processes are on a high/higher level than in the currently assessed project. The collected evidences, assessment reports, assessor comments and ratings from these reference projects (best practice examples) help to exploit corporate synergies for the improvement of the currently assessed project.

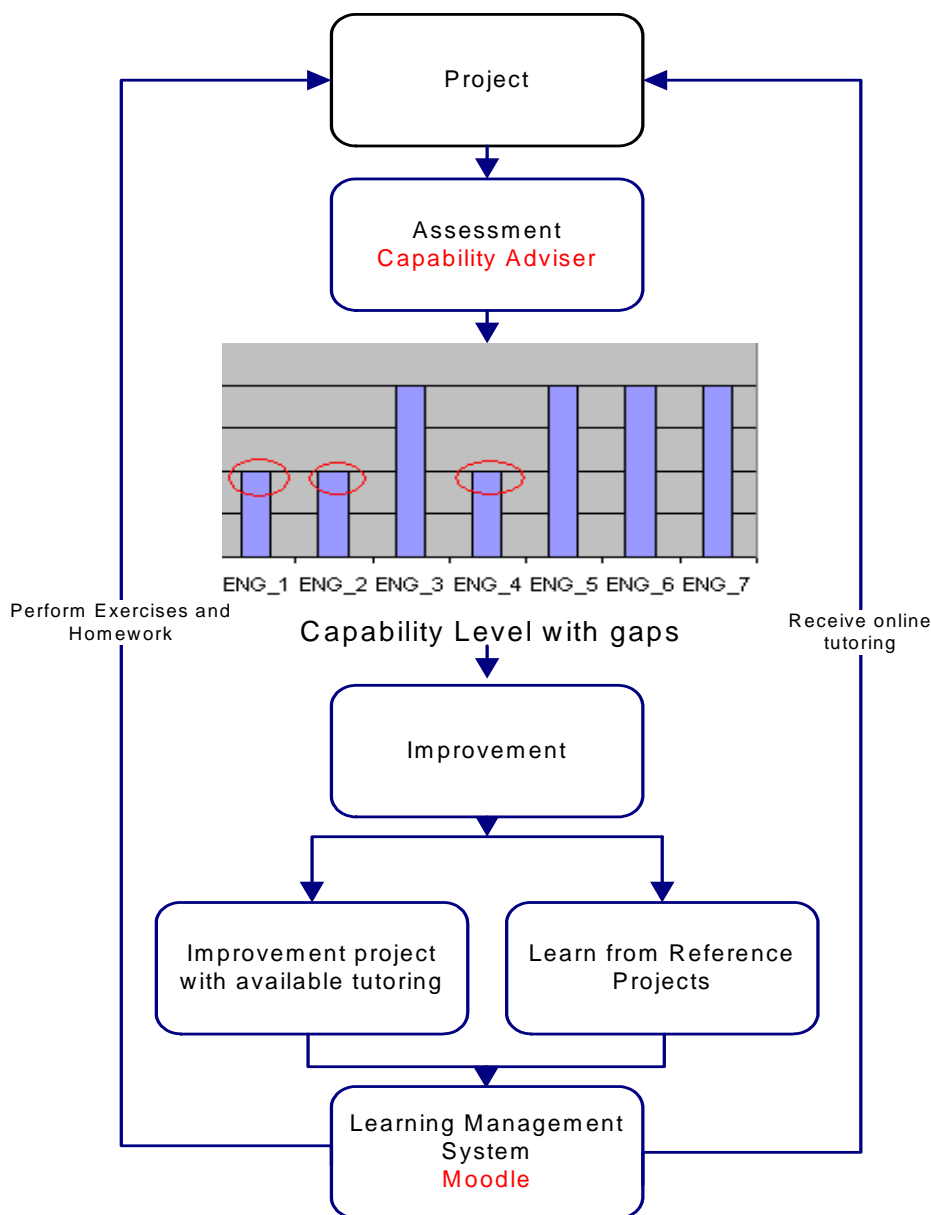


Figure 1: The approach

In addition to that an improvement project (a service project to develop a new best practice to be used in the projects) might provide virtual learning spaces. An improvement project is usually started if all the projects in the company have significant gaps in a certain area or process and new best practices must be developed for the organisation. The results of the improvement projects are then used to improve all the other processes in the company projects. From both, the reference projects and improvement projects, experts are acting as tutors to help to improve the processes by offering online tutoring, trainings, presentations, learning materials, by leading discussion forums, preparing exercises and assignments etc. All this activities are performed online in a Learning Management System.

## 2 The Implementation

### 2.1 The Underlying Systems

For the implementation, the following two systems were used: The process assessments and capability level calculations were done with the Capability Adviser Web Assessment<sup>1</sup> System. For the tutoring, training and online courses the learning management system Moodle<sup>2</sup> was used.

The **Capability Adviser System** is an online web assessment system. It supports different process sets based on different process models (for example ISO 15504, CMMI and Automotive SPICE) and it offers a complete management of assessments, assessors and projects. The system can export the results to various formats such as XML (according to the INTACS Scheme), Microsoft Excel and Word<sup>3</sup>. Some of the features included are evaluation and reporting of results, benchmarking with other projects, online browsing, online self and formal assessment, capability profiles, attribute ratings, ratings on the work product level, and virtual access to assessor comments in a team etc.

Assessors can share their views and comments during an assessment. The assessor results and their valuable comments are kept in the knowledge SQL database so that upcoming assessment can also learn from the mistakes of the previous.

In a previous Minerva project called EPI<sup>4</sup> (Educational Partnerships through ICT) the Capability Adviser was integrated with the free and open source<sup>5</sup> course management system **Moodle** (Modular Object-Oriented Dynamic Learning Environment). A very important reason why Moodle was chosen is it's easy of use and the popularity in university and industry sector. Moodle offers a wide set of Activities (assignments, chat area, discussion forum, presentation, workshops etc.), courses on a daily, weekly or topic basis, user management, translations to over 70 languages and a growing development community..

### 2.2 The Assessment and Knowledge Links

One of the goals of the formal assessment is to determine the capability level of the assessed processes. In the below example (figure 2) the assessment was performed for a group of Engineering processes. The capability level chart for the project shows that some of the Engineering processes achieved a level 2, while others like Requirement Elicitation, System Requirements Analysis and Software Requirement Analysis are still on level 1. With the more detailed Attribute Ratings Profile (figure 3), some minor gaps can be identified on level 1 (*Largely Adequate* instead of *Fully Adequate*), but major gaps can be found on level 2 where the process attributes are rated as *Partly Adequate* and

<sup>1</sup> [www.iscn.com/projects/piconew/](http://www.iscn.com/projects/piconew/)

<sup>2</sup> [www.moodle.org](http://www.moodle.org)

<sup>3</sup> Microsoft, Windows, Excel and Word are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

<sup>4</sup> <http://www-deis.cit.ie/epi/default.html>

<sup>5</sup> <http://www.opensource.org/docs/definition.php>

a lot of improvement is needed. The next example will focus on the improvement of the process Requirements Elicitation.

## Capability Level Charts for ISO15504 Demo

Unit	Assessor	Capability Level					Note
		1	2	3	4	5	
ENG_1 Requirements Elicitation	Demo Assessor	1					
	self assessment						
ENG_2 System Requirements Analysis	Demo Assessor	1					
	self assessment						
ENG_3 System Architectural Design	Demo Assessor	2					
	self assessment						
ENG_4 Software Requirements Analysis	Demo Assessor	1					
	self assessment						
ENG_5 Software Design	Demo Assessor	2					
	self assessment						
ENG_6 Software Construction	Demo Assessor	2					
	self assessment						

Figure 2: Capability Level Profile

## Capability Level Attributes for ISO15504 Demo

Unit	Assessor	Attributes									
		1	2.1	2.2	3.1	3.2	4.1	4.2	5.1	5.2	
ENG_1 Requirements Elicitation	Demo Assessor	LA	LA	PA							
	self assessment										
ENG_2 System Requirements Analysis	Demo Assessor	LA	PA	PA							
	self assessment										
ENG_3 System Architectural Design	Demo Assessor	FA	FA	FA							
	self assessment										
ENG_4 Software Requirements Analysis	Demo Assessor	LA	LA	LA							
	self assessment										
ENG_5 Software Design	Demo Assessor	FA	FA	FA							
	self assessment										
ENG_6 Software Construction	Demo Assessor	FA	FA	FA							
	self assessment										
ENG_7 Software Integration	Demo Assessor	FA	FA	FA							
	self assessment										

Figure 3: Capability Level Attributes

In figure 4 the process *Requirements Elicitation* has been selected. The assessment based learning system offers a list of company references and improvement projects with tutors and in-house trainings and in addition a list of available public courses containing similar topics. The list of projects, trainings and courses depends mainly on the selected target level. In general the lower the level the more reference projects and improvement trainings are available. To provide a wide selection of reference projects with tutoring and in-house training strong management support is needed. It is expected that the tutors use 20-30% of their time for training and tutoring. The selection of the tutors plays also a very important role. Not all experts can pass their knowledge. Training for the learning management system, time to prepare the materials etc. must be also taken in account before selecting a project and their tutor to become a reference project.

The assessment data containing the assessor comments and rated evidences from each assessor from a reference project can be accessed through the *Assessment Log*. The project can this way already improve some of their evidences (in most cases they're documents) by just reviewing the reference materials consisting of the assessment report, assessor comment, evidences, improvement plan etc. In most of the cases (especially with the low level processes) studying the reference project materials is not sufficient. Guidance from experts in form of tutoring and in-house training is needed. In the system the *Register* button automatically registers the user (project member) for the selected tutorial or training in the learning management system Moodle. The same username and password are used by both systems.

Similar to the reference project assessment data the improvement project data can be accessed. An improvement project is started if all the projects from the company have gaps in a particular process or a process area. The improvement project results are afterwards used by the projects. [Tutoring of an improvement projects is very important...](#)

The second list offers additional online Moodle (public) courses connected to the selected topic. The courses are not always public, but rather held in a small community of companies with the same focus (for example as in the Soqrates<sup>6</sup> group). The chosen Customer Relationship Management course from the Innovation Manager training<sup>7</sup> has the same focus as the Requirement Elicitation process but in a more holistic approach. The course deals with topics like learning from customers, how to capture customer innovation plans in advance, establishing knowledge management strategies for collecting and sharing product ideas/requirements etc.

---

<sup>6</sup> [www.sogrates.de](http://www.sogrates.de)

<sup>7</sup> Certified Innovation Manager: [www.innovationmanager.org](http://www.innovationmanager.org)

## Capability Level Charts for ISO15504 Demo

Target Level 2

Unit	Assessor	Capability Level					Note
		1	2	3	4	5	
ENG_1 Requirements Elicitation	Demo Assessor self assessment	1					

**Available tutoring and in-house training**

Project Name	Contact Person	Assessment Details	Course	Sign-in
Demo Reference Project 1	Mr. John Smith	<a href="#">Assessment Log</a>	Tutoring <a href="#">more ...</a>	<a href="#">Register</a>
Improvement Project Req. Ele	Mr. Klaus Jederman	<a href="#">Assessment Log</a>	Requirements Elicitation Training <a href="#">more ...</a>	<a href="#">Register</a>

**Public courses:**

Title	Date	Provider	Sign-in
Customer Relationship Management <a href="#">more ...</a>	13. 6. 2006 - 30. 6. 2006	ISCN <a href="#">about ...</a>	<a href="#">Register</a>

Figure 4: Available Tutoring and Courses

### 2.3 The Learning Cycle and the Tutoring

After the gaps have been identified and the training has been selected the improvement and learning cycle starts. The figure shows the six steps of the learning cycle and tutoring:

1. In addition to the presentation, discussions etc. the tutor sets homework's/exercises for the training
2. The project participants receive their homework, can discuss unclear thing with the tutor and
3. Upload their homework back to Moodle
4. The tutor reviews the homework and provides feedback

The first four steps can be repeated until "enough" homework/exercises are performed. This homework/exercise can be uploaded to the Capability Adviser system as a new evidence (step 5.) which results in a higher capability level (step 6).

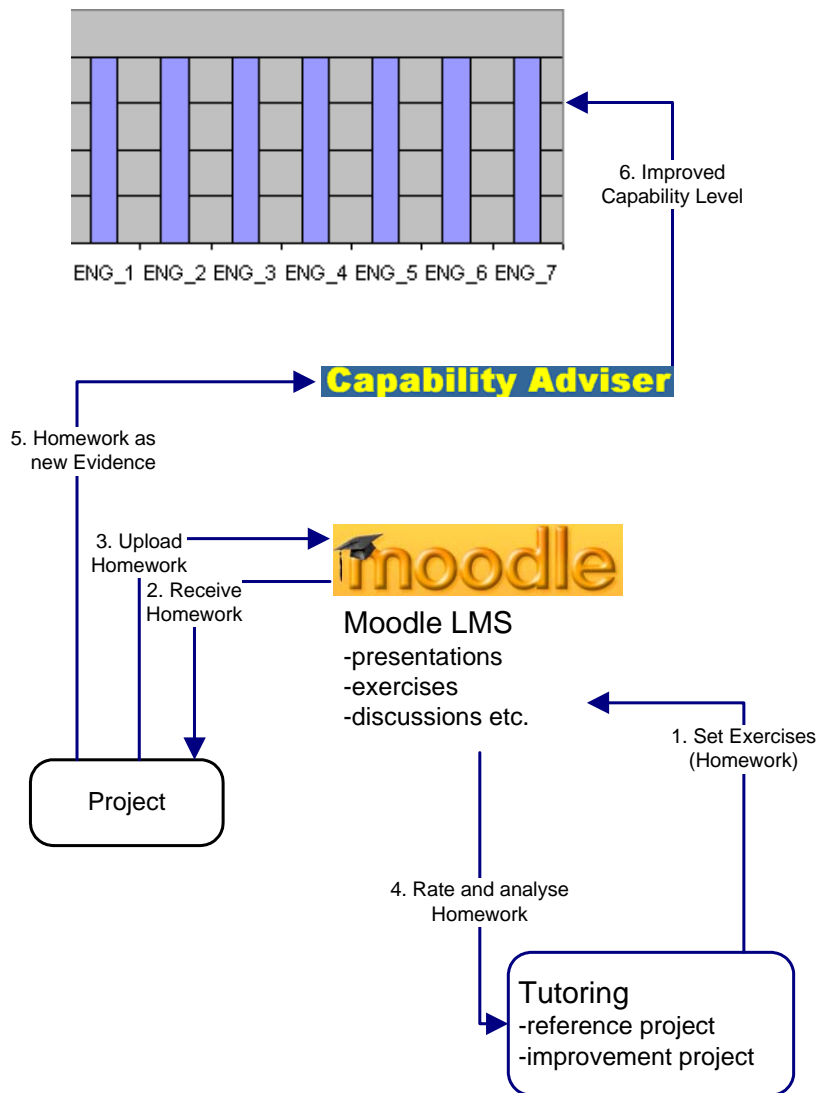


Figure 5: Learning Cycle and Tutoring

The online training in Moodle is divided into two parts. The first part offers some general training information as well as informal means of communication for example a chat room and a discussion room. In the second part the course is planned for a number of weeks. The first week for example offers some training materials in PowerPoint, list of references materials, examples with case studies and already an assignment. The training program can also be divided to topics instead of the weekly structure. Moodle offers here a wide set of activities and possibilities to configure the training individually. The tutoring differs from the training in the way that no additional materials and presentations are provided; the tutor provides and review's homework and participant in discussion forums.



### Requirements Elicitation Tutoring

LMS » RE

Weekly outline

#### Welcome to the Requirements Elicitation Tutoring

For questions i am available in the chat room Mondays and Thursdays from 10 to 11.

-  News forum
-  Chat Room
-  Discussio Forum

#### 1 2 May - 8 May







-  Requirements Elicitation Assignment 1
-  Requirements Elicitation Case Study - division Berlin
-  Requirements Elicitation Tutoring
-  Requirements Elicitation Slides 1
-  References
-  Example 1

Figure 6: Requirements Elicitation Tutoring

### 3 System – Roles and Implementation

The Capability Adviser system supports different roles:

- **Administrator:** The Administrator creates organizational accounts for the different divisions and creates a corporate wide assessor pool.
- **Organization:** Each organization can administer projects, assessments, pool of assessors, tailoring of process selections specific to the organizational unit, and it does the maintenance of project and assessor accounts.
- **Content Provider:** This is a specific user who can enter new assessment models, learning references, and assign courses to specific processes. The system allows entering knowledge links.
- **Assessor:** The assessor has a work bench to assess projects, review evidences (uploaded), rate processes (base practices and generic practices), exchange the data between the whole assessment team online, generate profiles, generate reports, generate assessment logs, etc.
- **Participant/Learner:** The participant is usually a project representative who can log in, perform a self assessment, see the assessor comments and results, generate reports, and upload evidences. They can also request best practice experiences or access online organized courses to learn best practices.

The Moodle system supports different roles.

- **Administrator:** The Administrator creates courses, teaching modules, home work exercises, tutoring facilities in Moodle ([www.moodle.com](http://www.moodle.com)) and configures all courses.

- **Trainer:** The trainer is assigned to specific online courses, is available at certain online times, performs online discussions, provides homework, corrects homework materials online and summarizes the results in the learning team.
- **Student:** The student participates in the course online, accesses the learning material, discusses with the tutor, performs a homework, receives feedback, refines the applied homework (application of concept in the work place), and uses the refined material as an evidence later to prove in assessment that this competence is covered now.

The systems are mostly used for ISO 15504 and combined courses are agreed for a corporate wide improvement in the firm.

Project personnel act as participants in the Capability Adviser System as well as students in the Moodle system.

### 4 Results

The public online courses have a more general approach and a wider target audience than the in-house training courses, which are tailored to the company processes. The Moodle configured Customer Relationship Management course can be seen in figure 5. The course offers PowerPoint Slides extended with English and German voice presentation, as well as exercises, reference materials, discussion area etc. In the year 2005 the Soqrates<sup>8</sup> group (formed as an Bavarian Initiative to share knowledge in important SPI and innovation fields) used the course already to improve their process Requirement Elicitation process.

So far major organisations

- Continental
- ZF Friedrichshafen AG
- Giesecke & Devrient
- T-Systems (German Telecom)
- Etc.

are already using the system. Meanwhile experiences show that using the synergy potentials inside a corporate firm in a tactical way leads to about 50% time reduction in achieving a higher level in a project. (Presuming that expert links can be found and exploited).

---

<sup>8</sup> [www.sogrates.de](http://www.sogrates.de)

**Customer Relationship Management**

ISCN » CRM

Topic outline

[Best Practice Example \(Beispiel\)](#)

1

- [CRM English Slides English Voice](#)
- [CRM English Slides German Voice](#)
- [References](#)
- [Notes](#)
- [Exercise Slides](#)
- [Exercise Upload](#)
- [CRM Discussion Forum](#)
- [CRM - Chat-Room](#)

Figure 7: Public Customer Relationship Management

## 5 Conclusions & Outlook

One of the most pertinent questions a business manager can ask is the following “How can I make my firm succeed where another fails?”

A key concept of the approach is the notion of lever [1]. Levers are means used by a firm to increase its resource generating ability, just as a mechanical lever is used for increasing the force applied to an object. The analogy goes even further. Just as a force can be applied in many different ways to the object resulting in a similar displacement, the use of the different levers can increase the resource generating ability of the firm resulting in similar business benefits. Finally, the resources are used to increase the assets of the firm and to reward employees and stockholders.

In the business manager world the following levers are known (MBA studies) and M. Biro in [1]:

- Operating leverage
- Production leverage
- Human leverage
- Marketing leverage
- Financial leverage

The approach described in this paper directly relates to the human leverage. The exploitation of human leverage is particularly important in software process improvement since software development is a fundamentally human mental process.

An exchange of best practices and a guided (using a learning system underlying the assessment portals) and tutored learning of best practices enables a learning culture which increases the speed of improvement and distribution of knowledge.

SPI is more than a technical issue, and we need more systematic approaches to include the human learning process in such systems in the future.

## 6 Literature

- [1] Messnarz R., Tully C. (eds.), *The PICO - Book: Better Software Practice for Business Benefit - Principles and Experience*, IEEE Computer Society Press, in publication
- [2] 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
- [3] 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
- [4] *A Learning Organisation Approach for Process Improvement in the Service Sector*, R. Messnarz, C. Stöckler, G. Velasco, G. O'Suilleabhain, *A Learning Organisation Approach for Process Improvement in the Service Sector*, in: *Proceedings of the EuroSPI 1999 Conference*, 25-27 October 1999, Pori, Finland
- [5] O'Keeffe, T., & D. Harrington, 2001. *Learning to Learn: An Examination of Organisational Learning in Selected Irish Multinationals*. *Journal of European Industrial Training*, MCB University Press, Vol. 25: Number 2/3/4

### 7 Author CVs

#### 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,
- MediaSF - Eworking 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 author of many publications in e-working and new methods of work in conferences of the European Commission (E-2001 in Venice, E-2002 in Prague), and in the magazine for software quality (Software Quality Professional) of the ASQ (American Society for Quality).

He is a lead ISO 15504 assessor. He has worked as a consultant for many automotive firms, such as BOSCH, ZF TE, ZF N, Continental TEMIC, Audi/VW, etc. He is a founding member of the INTACS (International Assessor Certification Scheme) accreditation board, a founding member of the Austrian Testing Board, a founding member of the Configuration Management Board, and he is the technical moderator of the SOQRATES initiative ([www.socrates.de](http://www.socrates.de)).

#### MSc Damjan Ekert

Damjan Ekert MSc, (dekert@iscn.at) graduated with distinction from the University of Technology in Graz, Austria. He is working with ISCN Austria (<http://www.iscn.at>) since 2000. He has been involved in many EU projects and is currently responsible for the software development of the Capability Adviser System and the NQA e-working platform solution. He is also an ISO15504 provisional assessor.

# Using Groupware Technologies to Facilitate Organisational Learning and Software Process Improvement – A Case Study

*Riikka Ahlgren<sup>1</sup>, Mirja Pulkkinen<sup>1</sup>, Eleni Berki<sup>2</sup> and Marko Forsell<sup>3</sup>*

<sup>1</sup>University of Jyväskylä,  
Information Technology Research Institute,  
P.O. Box35, FIN-40014 University of Jyväskylä, Finland,  
{ahlgren, pulkkinen}@titu.jyu.fi

<sup>2</sup>University of Tampere,  
Department of Computer Sciences,  
Kanslerinrinne 1, Pinni B, FIN-30014, Finland,  
eleni.berki@uta.fi

<sup>3</sup>SESCA Innovations Ltd.,  
iPark, Vaasantie 6, FIN-67100 Kokkola, Finland,  
marko.forsell@sesca.com

## **Abstract**

This paper provides suggestions for software process improvement with respect to communication and the use of groupware for distributed teams. The focus is on facilitating software improvement efforts from a bottom-up perspective, considering individual and team levels. The case study was conducted in a Finnish, rapidly growing software development company with activities at six locations. The utilisation and capabilities of electronic communication tools are analysed, and implications of their use to facilitate process improvement initiatives in a distributed development setting are evaluated. It was found that using a variety of communication tools can provide a learning organisational environment suitable for process improvement activities and facilitate in performing selected improvement practices.

## **Keywords**

Groupware, Communication, Organisational Learning, Software Process Improvement

### **1 Introduction**

Continuous process improvement is essential to the establishment of a quality culture in a software development organisation (Chrissis et al. 2005). Previous empirical research shows that a successful Software Process Improvement (SPI) critically depends on organisational factors (Dybå 2002; Gasston and Halloran 1999; Kock 1999). These studies suggest that software organisations should, in their SPI efforts, focus not only on process definitions and measuring, but also on creating an organisational environment which fosters the organisational success factors.

Furthermore, a number of classical studies in knowledge management and organisational processes coordination have indicated that fostering a learning environment could greatly enhance organisational processes. Total Quality Management (TQM) and Software Quality Management (SQM) approaches in particular emphasise the need for human collaboration and individual and explicit knowledge sharing through successful communication tools.

Process improvement is about changing specific processes to make them more efficient (Adesola and Baines 2005). Organisational learning is another side of the coin, for it provides an alternative metaphor to understand and improve organizations (Robey et al. 1995). Furthermore, it can not be divided into particular business processes; it merely is an integrative process (Robey et al. 1995).

What is captured in process quality models and metrics is a sum of actions by the organization members that contribute to the organizational maturity. Individual learning cumulates to organizational learning, and learning is essential when planning and implementing organisational process improvement efforts (Segal 2001).

Previous research on groupware and process improvement has mostly concentrated on using groupware as a tool for the process improvement group (see e.g. Kock 1999; Kock and McQueen 1995) or on decision support in process re-engineering (Dennis et al. 2003; Hakkarainen 2003). In this paper we focus on facilitating the software process improvement efforts utilising a bottom-up approach. We examine the communication tools that enable the communication on individual and team level, thus scaffolding the collaboration and collective learning. The bottom-up approach is chosen to introduce the process improvement efforts in a practical and immediate way to the software developers and thus, to facilitate the actual improvement when it is taking place (see e.g. Conradi and Fuggetta 2002). This view thus considers the human factors that have empirically been proven to play a critical role in successful software process improvement (see for example Conradi and Fuggetta 2002; Dybå 2002; Hoegland and Gemuenden 2001; Siakas et al. 2003).

This paper is organised as follows: Section 2 outlines the process improvement goals by identifying the communication and learning needs in software development. Section 3 outlines the groupware technologies used in the case. Groupware is presented as the tool to facilitate information exchange, collaborative learning and knowledge sharing within the development processes that a group undertakes. Section 4 describes the research method and the case study organisation, as well as the analysis of the problems encountered. The findings of the case study and conclusions are provided in Section 5.

### **2 Communication as a Process Improvement Goal**

Communication within a team is said to be the most important factor for efficient teamwork (Hoegland and Gemuenden 2001) and thus central for efficient software development. Communication is therefore also our approach to the process improvement.

The organisation, where the case study was conducted, has chosen CMMI as their process improvement reference model. The process improvement goals of our study thus follow this model. In CMMI, the generic practices are performed throughout all processes and in all maturity levels. Hence, their universality makes them a good starting point for improvement actions.

The first process improvement goal of this study was to facilitate the establishment of an

organizational policy (CMMI generic practice 2.1). According to this practice, senior management is responsible for establishing and communicating guiding principles, direction, and expectations for the organization (Chrissis et al. 2005). Our research interest lies in studying how the communication between management and other employees can be facilitated by using the various groupware and particularly the electronic communication tools.

The second improvement goal is related to CMMI general practice 2.5, training the people performing or supporting the process as needed (Chrissis et al. 2005). In the case organisation a particular goal is to support developers' peer-to-peer training in an informal way, and to allow everyone to have mentors whom to ask when problems arise. The aim is also to encourage the developers to cooperate in general, instead of using their time in solving the difficult problems alone.

To motivate the importance of communication in process improvement, the communication quality within a team is described with four attributes, namely communication openness, formalisation, frequency and structure (Hoegland and Gemuenden 2001). Open communication improves the software process by allowing the tasks to be done in a correct and consistent way and by facilitating team collaboration in order to find the defects faster. *Openness* is essential, for holding back important information hinders the integration of the team members' knowledge and experiences on their common task (Hoegland and Gemuenden 2001). This knowledge integration serves as a basis for collective learning.

Changing *communication formality* allows the developers to concentrate on their actual tasks instead of the communication itself. When the formality of communication can be changed regarding the issue and situation, the communication gets smoother which further makes the process go smoother. The variety of tools available (e.g. chat, which does not require attention from the receivers) allow the developers to concentrate in their actual tasks and not on answering the phone in the middle of the busiest time.

*Frequent communication* diminishes the insecurity feeling of the developers. When developers know that they know everything worth knowing, it results in shorter development time by eliminating misunderstandings. Communicating experiences and personal and explicit knowledge enables the organisation to learn, and thus to change particular processes and activities to a more preferred form (Lyytinen and Robey 1999).

*Communication structure* refers to the communication network in use. For teamwork efficiency it is important, that team members can communicate directly with each other, for using mediators is time consuming and can easily cause misunderstandings. (Hoegland and Gemuenden 2001)

Learning as a part of the daily work is an essential factor when it defines high quality ways of developing software (Collin 2005; Holden et al. 2003; Lyytinen and Robey 1999). When individuals become aware of efficient practices and know-how, and consequently apply these to their work, defects can be eliminated and can be fixed faster. Thus, learning enables more effective ways to do the development work, which further enables the process and the product to reach higher levels of quality. To enable individuals' experiences and knowledge to become collective knowledge in the organisation, rich communication is needed. In particular, when employees are not co-located, groupware can greatly facilitate their communication (Ellis et al. 1991).

### **3 Groupware as Support for Communication and Learning**

Process improvement and organisational learning can both be aided by information technology (Robey et al. 1995). In this paper we focus on groupware, particularly on electronic communication tools in facilitating the organisational learning by enabling the developers to communicate efficiently and thus to learn in their work environment. Groupware is defined as a software that assists a group to work together for achieving a shared goal (Hakkarainen 2003). Five different types of groupware are identified: tools for communication, collaboration, coordination and tools for group memory (Hakkarainen 2003). Our focus is on the communication tools, which can further be divided into five



categories: 1) chat, 2) messaging, 3) email, 4) bulletin board and 5) videoconferencing (Hakkarainen 2003).

Chat systems provide synchronous, text-based discussions between multiple users (see Kammonen 2004). Some chat software also allow voice and video discussions and enable sending and receiving of documents. Electronic messaging and email are quite similar; the biggest difference is that email can be sent to anyone who has an email address, but in electronic messaging systems the messages can only be sent inside the environment (Hakkarainen 2003). Another difference is that e-mail fits best to unstructured communication, while messages sent via messaging systems can be supported with semi-structure and metadata (Hakkarainen 2003). SMS is a special form of messaging, for the messages are sent and received by mobile phone. Thus, the message length is restricted, commonly up to 160 characters. A new way of messaging is enabled by Internet telephone (IP telephone). It uses Voice over Internet Protocol (VoIP), which refers to sending telephone calls over data networks, instead of the traditional telephone network, to predefined phone ip-numbers (Talkswitch Visited 29.3.2006.). An electronic bulletin board can be in different formats. For example wiki-webs and blogs can be seen as bulletin boards, since they both are public boards where authoring is either free or limited to a smaller group. A wiki can be defined as a piece of server software that allows users to freely create and edit web page content using a web browser (Leuf and Cunningham 2002). A blog is abridged from "web log". It is a frequently edited on-line journal, where people can post diary entries about their personal experiences (Webster's Online Dictionary). Videoconferencing in turn allows synchronous interaction despite of physical distance. Desktop conferencing systems enable sharing pictures, documents or other relevant material (Ellis et al. 1991), while new cellular phones with integrated camera support mobile videoconferencing.

### **4 Case Study Presentation**

Case study is commonly employed as an empirical research strategy in information systems field, often used for describing relationships within organisational settings (Galliers 1992). In the context of our research, we employ case study as the method to provide an organisational context for the identification of software improvement factors and their introduction and deployment to the particular software development organisation. We embarked to research and study the daily practices that facilitate the creation and support for a total quality organisational environment, favourable for process improvement efforts.

The research was conducted at SESCA Innovations Ltd, which is an international software house whose core business concentrates on IT- and processing industry. The company was founded in 1998, and in the recent years it has heavily been expanded. Currently, SESCA has activities at six different locations in Finland. All together, the number of employees is around 140 people. A current target of the company is to improve their organisational and software processes. For this they have chosen the CMMI continuous model. Albeit young, the company already has tradition in quality work: they obtained the ISO 9001:2000 certificate only a few years after the company started.

During the research process the development manager of SESCA Innovations was interviewed in meetings and in telephone conferences and accompanying e-mail exchange. His co-authorship in this paper ensures the correct interpretation of the obtained information. During the research, electronic communication tools were identified to be the key area to concentrate on, since their presence and use in the organisation are extensive.

Communication and learning at work have been identified to be the key areas for SESCA in their process improvement efforts, although distributed software development sets restrictions for the suitable communication methods. In addition, developers' working hours are flexible, which decreases the time they spend together at work. Due to these factors, electronic communication tools are central for the employees to exchange information. The electronic communication tools at use and the communication features they support are presented in Table 1. A similar classification but within a different organisational scope has been created by and presented in (Kammonen, 2004).

**Table 1: SESCA portfolio of electronic communication tools**

Tools at use	Communication features
Messenger	Chat, document sharing
IP telephony	Presence information, telephone conferences
Talk-mail	Voice messages
SMS	Short personal messages
E-mail	Personal messages and distributing lists
Blog	Authored information bulletin
Wiki-pages	Sharing of common knowledge

### 4.1 Tool-aided Communication at SESCO

In the case organisation electronic communication tools are widely used to facilitate the daily work. Developers use mainly IP telephony, chat and email as communication tools. Chat is used for informal communication, i.e. about how the work is going on and for memory support of small details during the development. A chat conversation could concern, for example, a location of a template or other company specific guidelines. Also, when specific actions, functions or other programming-oriented information are required, they can be requested by using chat systems. Chat itself enables the request and response to be faster and easier than if the advice was asked in person. A chat system, thus, also enables problems to be communicated, shared and solved more quickly.

IP telephony is a new way of communication which has been introduced in the company during past year. One of the key benefits of IP phones has been the presence information, which explicitly points out how to reach a particular person in the organisation. Presence information tells, for example, whether a requested person is currently in the office or not. If the person is not there, another communication tool is chosen, for example e-mail. Also ad hoc conferences are arranged often, since they are very convenient to set up.

Talk mail is less frequently used than IP telephony. Talk-mail enables the caller to move on with her/his tasks, without the need to constantly re-call the requested person. The call receiver then chooses an appropriate time to listen the message left in a talk-mailbox, without needing to answer the phone immediately. SMS messaging is used for similar purposes as talk-mail, though the messages are shorter and thus the subject matter can not be very complicated.

E-mail is widely used by all members of the organisation. Often the contents of email messages are questions and answers about a particular product or project, and at a more abstract level than the issues discussed via chatting. Also, only developers use chat in their mutual communication, and communication between developers and other personnel is either by IP telephone, face-to-face or by e-mail. The management uses e-mail also to inform about upcoming meetings and decisions which have been made (if not informed in monthly personnel meetings).

Web-logs (blogs) are used to give information in a more formal way than other communication tools. Typical contents of a company's internal blog are general information of what is going on in the company. In SESCO the purpose of a blog is to spread knowledge about the goals of the organisation, reasons for needed actions and how well the goals have been or could be met. One of the main reasons for the latter is to share the organisation's vision and mission and make them crystal clear for each and every one in the organization. Except clarity, wiki-pages are planned to facilitate the sharing of company or project wide information, and also to make available software patterns and other reusable assets for the development work; thus to create organisational memory (Ahlgren and Markkula 2005). Currently wiki-pages are in an experimental phase.

However, news concerning a particular individual are never communicated via electronic tools. These changes are always apparent in face-to-face communication. Although the organization is located all around Finland, regular face-to-face meetings within projects are fostered and appreciated.

Taken together, groupware and electronic communication tools are seen to be an efficient way to facilitate daily activities in the case organisation. In particular the communication tools integrated with the IP phone are considered to be an easy and efficient way to improve communication within the

organisation. One step further would be to provide a user-centred development by also integrating the client and end-user (Berki et al. 2003) within this communication loop.

### 5 Case Study Results

Design knowledge is a significant business asset for a software company (Terveen et al. 1993). To create new design knowledge requires learning by individuals and exploitation of the gained experiences. Making process improvement efforts requires proficient communication between the individuals and the importance of communication is highlighted in distributed software development (Haag et al. 1997), as it is in our case organisation. Knowledge sharing and information distribution at team and individual level requires, in a distributed setting, a variety of communication media.

In the case organization, utility tools for quick messages (chat, talk-mail, SMS) are available, known and widely used throughout the stages of the software development. These are perceived as a handy tool in the daily work.

The first process improvement goal in this paper was to facilitate the creation of an organizational policy by communicating the guiding principles, direction, and expectations for the organization. In the case organisation blogs and email were used to facilitate the communication of visions and plans to the personnel, thus to support the creation of an organizational policy. The planned wiki-pages are to further facilitate this communication. This communication supports the software development as such, when developers' working time can be used efficiently in the planned tasks without need for frequent meetings or pondering on the managerial decisions.

The second improvement goal, training the people, was mostly supported by chat, which allows the developers to freely discuss open issues and solve possible problems together, without formal meetings. This is seen to improve the quality of the processes and the products, e.g. software designs. The downside of the chat is, that the discussions or solutions are not generally saved anywhere, thus they are not accessible later on. Also Lyytinen et al. have found peer-to-peer communication to be an effective learning mechanism, particularly in fast-evolving environments (Lyytinen et al. 2002).

As a conclusion to our study, we next summarise the observations and draw comparisons to other similar research findings that define some software and total quality features in an environment suitable to support the process improvement efforts.

- The organisational environment must make information available and provide an opportunity to learn on-the-job, both with information search and by asking colleagues. This maps to Lyytinen and Robey's improvement suggestions for providing incentives to learn (Lyytinen and Robey 1999).
- A challenge for both process and product quality improvement is that teams and members share their knowledge. The case company has chosen this as a principle in their values: they encourage people not to waste any time with problems they are not able to solve alone but ask for help. Company members are obliged to provide the help they possibly can. This means fostering co-operation rather than competing, as it is also indicated in (Kock 1999).
- Be open for new ways of communicating and for experimenting new communication tools and technologies, thus encouraging to lower the educational barriers, as indicated in (Lyytinen and Robey 1999). Software developers can actually be very keen to try and adopt new technological solutions. At the same time, beware of breakdowns in mediated communication. Build trust actively.
- Let people know the benefits of co-operation and communication; this encourages them to share problems, ideas and accept solutions for common development approaches (Jarvenpaa and Leidner 1999; Kock 1999; Manninen and Berki 2004).

Software organizations striving for process quality improvement could revise the use of formal and informal communication tools they have. The defined process might require e.g. paper-based communication in the process, which calls for updating. Our case organization has taken the groupware tools into use with the support of management. Management involvement and action

strengthens the efforts towards quality improvement. To consideration, of course, remains if a tool that retains the communicated content is needed, maybe in a certain form and with certain information security level.

Suggestions for software development organisations to proceed to specific, standard strategies for software process improvement can not be derived from a case study alone. The latter could be dependent on different types of organisations, their aspirations and values and moreover on national culture. The organisational and geo-graphical boundaries (see Siakas et al. 2003) impose different software quality management strategies and different, possibly customised software quality standards and process improvement initiatives. However, the lessons learned and conclusions drawn from this research in a real, software development organisation could be generalised to other, similar to SESCO's organisational culture and values, organisations.

### **6 Acknowledgements**

The research work presented in this paper was done in MODPA research project [<http://www.titu.jyu.fi/modpa>] at the Information Technology Research Institute, University of Jyväskylä. MODPA project was financially supported by the National Technology Agency of Finland (TEKES) and industrial partners Nokia, SysOpen Digia, SESCO Technologies, Tieturi, Metso Paper and Trusteq.

### 7 Literature

- Adesola S., Baines T. 2005. Developing and evaluating a methodology for business process improvement. *Business Process Management Journal* 11(1): 37-46.
- Ahlgren R., Markkula J. 2005. Design Patterns and Organisational Memory in the Mobile Application Development. *Product Focused Software Process Improvement*. Vol. 3547, 143-156.
- Berki E., Isomäki H., Jäkälä M. 2003. Holistic Communication Modelling: Enhancing Human-Centred Design through Empowerment. *5th International Conference on Engineering Psychology and Cognitive Ergonomics*. Vol. 3, 1208-1212.
- Chrissis M.B., Konrad M., Shrum S. 2005. *CMMI: Guidelines for Process Integration and Product Improvement*. Addison-Wesley, Boston.
- Collin K. 2005. *Experience and Shared Practice - Design Engineers' Learning at work*. University of Jyväskylä, Jyväskylä.
- Conradi R., Fuggetta A. 2002. Improving Software Process Improvement. *IEEE Software* July/August: 92-100.
- Dennis A.R., Carte T.A., Kelly G.G. 2003. Breaking the rules: successes and failure in groupware supported business process reengineering. *Decision Support Systems* 36: 31-47.
- Dybå T. 2002. Enabling Software Process Improvement: An Investigation of the Importance of Organizational Issues. *Empirical Software Engineering* 7: 387-390.
- Ellis C.A., Gibbs S.J., Rein G. 1991. Groupware: some issues and experiences. *Communications of the ACM* 34(1): 39-58.
- Galliers R. 1992. *Information Systems Research. Issues, Methods and Practical Guideline*. Alfred Waller Ltd, Chippenham, Wiltshire, England.
- Gasston J., Halloran P. 1999. Continuous Software Process Improvement Requires Organisational Learning: An Australian Case Study. *Software Quality Journal* 8: 37-51.
- Haag Z., Foley R., Newman J. 1997. Software Process Improvement in Geographically Distributed Software Engineering: An Initial Evaluation. *23rd EUROMICRO Conference '97 New Frontiers of Information Technology*, 134-141.
- Hakkarainen P. 2003. Groupware support for operational management. Master's thesis, University of Jyväskylä.
- Hoegland M., Gemuenden G. 2001. Teamwork Quality and the Success of Innovative Projects. *Organisation Science* 12(4): 435-449.
- Holden R., Smith V., Devins D. 2003. Using 'employee-led development' to promote lifelong learning in SMEs: a research note. *Human Resource Development International* 6(1): 125-132.
- Jarvenpaa S.L., Leidner D.E. 1999. Communication and Trust in Global Virtual Teams. *Organisation science* 10(6): 791-815.
- Kammonen S. 2004. Light-weight, text-based communication applications: How can organisations benefit? Master's thesis, University of Jyväskylä.
- Kock N. 1999. *Process Improvement and Organizational Learning: The Role of Collaboration Technology*. Idea group Publishing, London.
- Kock N.F.J., McQueen R.J. 1995. Integrating groupware technology into a business process improvement framework. *Information Technology & People* 8(4): 19-34.
- Leuf B., Cunningham W. 2002. What Is Wiki, <http://wiki.org/wiki.cgi?WhatIsWiki>. Vol. 2006.
- Lyytinen K., Robey D. 1999. Learning failure in information systems development. *Information Systems Journal* 9: 85-101.
- Lyytinen K., Rose G.M., Yoo Y. 2002. Learning in High Gear: Hyper-learning and Dynamic Capability in Seven Software Firms. *Systems and Organizations*. 2(4): 160-194.

- Manninen A., Berki E. 2004. An Evaluation Framework for the Utilisation of Requirements Management Tools - Maximising the Quality of Organisational Communication and Collaboration. BCS Software Quality Management 2004 Conference, 139-160.
- Robey D., Wishart N.A., Rodriguez-Diaz A.G. 1995. Merging the metaphors for organizational improvement: Business process reengineering as a component of organizational learning. Accounting, Management & Information Technology 5(1): 23-39.
- Segal J. 2001. Organisational Learning and Software Process Improvement: A case study. 3rd International Workshop on Learning Software Organizations. Vol. 2176, 68-82.
- Siakas K.V., Berki E., Georgiadou E. 2003. Code for SQM: A Model for Cultural And Organisational Diversity Evaluation. EuroSPI 2003, IX.1 - IX.10.
- Talkswitch. Telephone Calls on the Internet: A VoIP Primer. [http://www.talkswitch.com/gateway/voip\\_primer.html](http://www.talkswitch.com/gateway/voip_primer.html). Visited 29.3.2006.
- Terveen L.G., Selfridge P.G., Long M.D. 1993. From "Folklore" to "Living Design Memory". SIGCHI conference on Human factors in computing systems.
- Webster's Online Dictionary. Webster's Revised Unabridged Dictionary Version 1913. C. & G. Merriam Co, Springfield, Massachusetts, USA. Visited 29.3.2006.

### 8 Author CVs

#### Riikka Ahlgren

Riikka Ahlgren obtained her MSc degree in 2002 majoring information systems and particularly groupware technologies. After graduating she worked a few years in a small software company as a team leader. Currently she is starting her PhD studies in Information Technology Research Institute in the University of Jyväskylä. Her research interests are software patterns and organisational learning in software engineering.

#### Mirja Pulkkinen

Mirja Pulkkinen holds a MA degree and a postgraduate certificate for education. She was working as a teacher of at the University of Jyväskylä, concentrating on the fields of information systems and business communications. Her special interest was learning supported by computer networks and virtual learning environments. She joined 1999 a master's degree programme at the Faculty of Information Technology and obtained a MSc degree majoring in information systems. Since 2001 she is working at the Information Technology Research Institute as a researcher and research project manager. Her research interests cover enterprise architecture as a coordination tool for organizational IS management and planning, knowledge management, and the software development organization. She is a Ph.D. student at the Faculty of IT.

#### Eleni Berki

Eleni Berki obtained her Ph.D. from UK in 2001. She currently is an Assistant Professor of Software Development in the Faculty of Information Sciences at the University of Tampere. She is a member of IEEE and BCS, where she is actively involved in organising and scientific committees on Total and Software Quality Management Standards and Software Process Improvement. Dr Berki has worked as a quality consultant in industry and academy, and as a lecturer and visiting lecturer in European and other Universities. She has more than fifty refereed publications in journals, book chapters and international conferences, and has given a number of talks in international forums and consortia. Her teaching and research interests include requirements engineering, computational models, organisational learning, knowledge creation and management where upon she supervises a number of MSc and Ph.D. students.

#### Marko Forsell

Marko Forsell got his doctoral degree in 2002 from the University of Jyväskylä. After working for 5 years at the University he moved to work for SESCA Technologies in 2003. Since then he has been involved in founding two startup companies SEVECON Group Ltd. and SESCA Innovations Ltd. Currently Mr. Forsell works as the CEO in SESCA Innovations Ltd. and acts as an Innovation Director in SEVECON Group Ltd. Mr. Forsell is also a member of board in ALMA Marketing Ltd. and a deputy board member of regional business development company called KOSEK in Kokkola, Finland.

# Success with improvement

- Requires the right roles to be enacted - in symbiosis

*Jan Pries-Heje IT-University Copenhagen, Denmark  
Jørn Johansen DELTA, Denmark*

## Abstract

This paper focuses on experiences and results from a Danish national research- and collaboration initiative – Talent@IT - on software process improvement (SPI) involving four software organizations and several researchers. The goal for the initiative was to develop a model and a method – *ImprovAbility*<sup>TM</sup> - to improve organizations ability to improve and innovate.

It is a fact that project members, project managers, process owners, project organization, top managers, involved experts and users all have different roles in relation to a main organizational goal: improving and becoming better at improving.

A main part of the research in the Talent@IT project has been to identify the kind of success related to improvement based on analysis on several interviews with the identified involved roles in improvement. This paper presents a role model where the roles are grouped in supporters, suppliers and users and roles are linked to success criteria. Finally it elaborates on how the roles better can be coordinated and enacted by using the *ImprovAbility*<sup>TM</sup> model.

## Keywords

Process Improvement, Organizational Improvement, Roles, Success

## 1 Introduction

Too many software developing organisations fail when they try to improve. They may go through an assessment. They may start one or more improvement initiatives. They may measure what is done. But somehow they never close the “learning cycle” where they continuously measure, improve, evaluate and learn. In fact figures from Denmark shows that more than half of the companies that goes through an assessment never manage to improve successfully.

To cope with this high “mortality rate” a research project was initiated in January 2003. The name of the research project was Talent@IT where the word “talent” in the title emphasised the role of people and individuals and their abilities and capabilities. In the concrete Talent@IT took place from 2003-06 and involved IT-University at Copenhagen, PBS, Danske Bank, ATP, SimCorp, and DELTA – with support from the Ministry of Science, Technology and Innovation.



If we look at a software developing organization then we find that a typical employee will “play a role” in improvement. Most organizations will focus on efficiency and the ability of delivering quality results. Often the main problems in relation to this are late results, missing results or results at inadequate quality due to poor requirements. Improvement is needed at all organizational levels. Everyone (nearly) has an interest in taking part in improving the organization – and have their “fingerprint” on the improvement result. In the end improvements are meant to help the employees to do a better job.

Prior research in process improvement (PI) has revealed that it is very difficult to reach the goals of efficiency and quality for several reasons. Lack of management commitment, submerged in day-to day work (part time is no time), PI workers living in an ivory tower, missing PI strategy, lack of plan and control, PI undermined by bureaucracy, firefighting and resistance against change. These arguments can be used for product improvement just as well.

However, some of the reasons can be seen from more than one side. Take for example “management commitment”. Employees may claim that they miss management commitment or they may say that managers are not involved. But if you on the other hand ask the managers, then they may tell you that they are never informed or they may tell you that the employees prefer not to involve management. Thus the same thing can be seen very differently depending on the role you have.

This paper therefore looks at process improvement and different roles. The empirical background for doing this is research from the Talent@IT project, especially Pries-Heje et al. (2003), Eichen & Dreyer (2005) and Elisberg et al. (2006). Furthermore this paper builds on personal experience from working with process improvement in several Danish companies as researchers and consultants.

## **2 Research Method**

We selected successful and failed projects as an arena of particular interest from the viewpoint of improving the ability to improve. We can highlight two key reasons for this interest. First, we appreciate the learning that can be harvested by looking at projects in retrospective. Second, in opposition to many other studies we decided to look at both SPI projects where other software developers are the users and at traditional IT projects in IT organizations.

We used an existing research collaboration called Talent@IT to select companies. As mentioned there are four companies that participated in this research collaboration. Each of the companies was asked to appoint two successful and two failed projects. We asked that the companies appointed two SPI projects and two normal innovation projects, preferably a successful and a failed one of each type. Furthermore we asked to have SPI projects that had delivered results that were used in the innovation projects.

We then conducted interviews in the projects. We interviewed the project manager and 1-2 project members. We interviewed the sponsor or owner of the project, typically a manager in the organization. We interviewed the users; for an SPI-project that meant other developers, and for innovation projects that typically meant end users. In 16 projects we conducted more than 50 interviews in the period from summer 2003 to summer 2004.

Typically every interview was conducted by two people. One interviewing and one taking notes. Subsequently all interviews was transcribed and analyzed using Grounded Theory techniques.

## **3 Definition of roles**

The first question that comes to mind when one starts to look at process improvement and roles is of course: What roles are there? One answer to this question is that every activity has someone doing it – the “Performer”, responsible for developing and implementing the improvement activities in the organization. Sometimes the result of doing something is a result that the performer will use but more often there is another person using the results of the activity – the “User”. Furthermore the performers

very often do not act alone. There will be a sponsor supplying money, a champion helping with technology transfer, an owner that owns the benefit coming out of using the result, or a manager making sure that a the performer is allocated to this specific activity and not three others. In general we can say that we have a “Supplier” role, characterized by distributing resources, knowledge and power.

If we now look at these three roles we can do that at different levels of the organization. If we look at the organization as a whole we will typically find something like a process improvement department or an SEPG (Software Engineering Process Group) that is responsible for performing the process improvement. The user of process improvement will typically be a customer or the like, and the supplier will be someone from top management enacting the role as sponsor.

In the individual project you can also find the same three roles but enacted differently as well as at the individual level. In Table 1 (inspired by Elisberg et al. 2006) we have given an overview of the roles at the three different levels mentioned. And after the table we have given an account of how we see each role to be enacted in the process improvement arena.

	<b>Organization</b>	<b>Project</b>	<b>Individual</b>
<b>Supplier</b>	Management / Sponsor	Project Organization	Expert
<b>Performer</b>	PI Unit	PI Project Manager	PI Developer
<b>User</b>	Customer	Project Manager	Product developer

Table 1. Process Improvement Role Matrix

### ***Management / Sponsor***

The Sponsor has the overall responsibility for aligning the improvement program according to the actual business needs. The responsibility for initiating and supporting the improvement activities in the organization is located here. The sponsor – typically a person from top management - is the person (or group) that endorses the improvement programs or projects and demands the results. This type of role is found among top managers with responsibility for business, product and process development. Only at that level of the organization there are enough power and influence.

### ***Project Organization***

The Project Organization (or Steering Committee) is the official body responsible for defining, scoping and controlling projects in the organization. They have to follow the project closely, and if unexpected problems or necessary changes occur they have to make decisions on changes of the basis or direction of the project. An important task is to ensure results – which often require continued contact and ongoing involvement of different groups of stakeholders. Normally this group is a group selected from management. Depending on the situation this group of people could be supported by external experts.

### ***Experts***

This can be internal or external experts or consultants, with the necessary competences to support typically Management or the Project Manager.

### ***Process Improvement Unit (Process owners)***

This is normally a visual part of an organizational diagram. It could be the department for Quality Assurance, a department for Methods or a less substantial part of the organization such as a Software Engineering Process Group (SEPG). This Unit is the owner of the processes in the organization. They have the responsibility – often not clearly defined – to maintain the formal set up of processes in the organization, to develop new processes, and to diffuse and ensure adoption of processes. This role is often staffed by employees with strong interests in quality, efficiency and ongoing process improvement.

### ***PI Project Manager***

The project manager is the person (or persons) that in practice prepares and carries out the project, and often also including the diffusion and adoption of the change in the organization. He or she must perform within the chartered course set by the PI top manager and is the facilitator for the PI effort. An important task is to run the project, including all normal project activities and disciplines such as plan-

## Session 2: SPI and Communication Management

ning, teaming, managing, monitoring and controlling the project and its risks. This role is often manned with an influential, high status project manager with a solid knowledge from within the organization.

### **PI Developer**

This person or normally group (or groups) of persons forms together with the project manager the project team. Depending on the project goals the project is manned with persons holding the necessary competences. This role is staffed from every necessary part of the organization – if just the persons have the right qualifications.

### **User**

When it comes to change in an organization, the users (of the change) can be everyone in the organization (customer, project manager, product developers, suppliers and performers). Only the experts are not taken as users. So in this context the users include those who perform the improvement work.

In an organization all these roles are necessary to be successful. They have to work closely together to “push” the change through, which is illustrated in figure 1 below.

The seven different roles: Management, Project Organization, Project Manager, Developers, Process Owners, Users and Experts; that we have defined are in figure 1 shown as part of an improvement project, as part of the organization, or as external to the organization. Suppliers are located in the upper part of the figure, performers in the lower part to the left, and users in the lower part to the right.

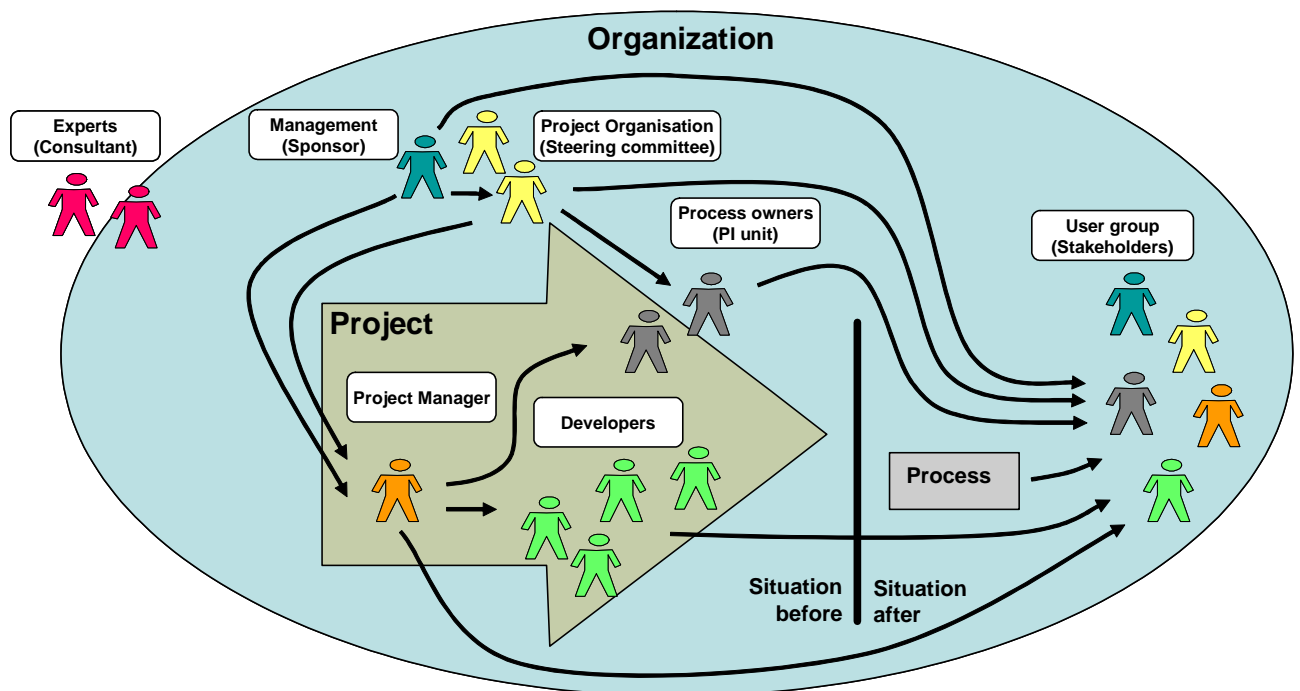


Fig 1. Role model for change – push of a change. Black arrows indicates communicated requirements.

The process improvement process (sic!) can develop in the following way. (1) Management initiates a project and (2) sets up a project team together with Project Organization (could include the development, method department and human resource management), which (3) in collaboration with the Project Manager and the Process owners (could be process improvement unit) forms a project. (4) The project is managed by a Project Manager and the work is performed by Developers with help from the Process owners. (5) Experts can be necessary to support Management or Project Managers with external knowledge or to avoid roles to be accused to ride their own hobbyhorses. The change – e.g. a new process ends at the users.

From looking at figure1 it becomes clear that Management, Project Organization and the Project Manager (and of course the project as a whole) are key roles to ensure successful change throughout the organization – ending with a diffused and adopted change. Each of these roles has three or more starting arrows indicating that they have the primary keys to the process. All the requirements (the push for change) end at the users “doorstep”.

A success demands activities addressing diffusion and adoption – an often forgotten or neglected “part of the play”. It also requires a recognized need from the user – which is much easier if the users have been involved as stakeholders before and during the project.

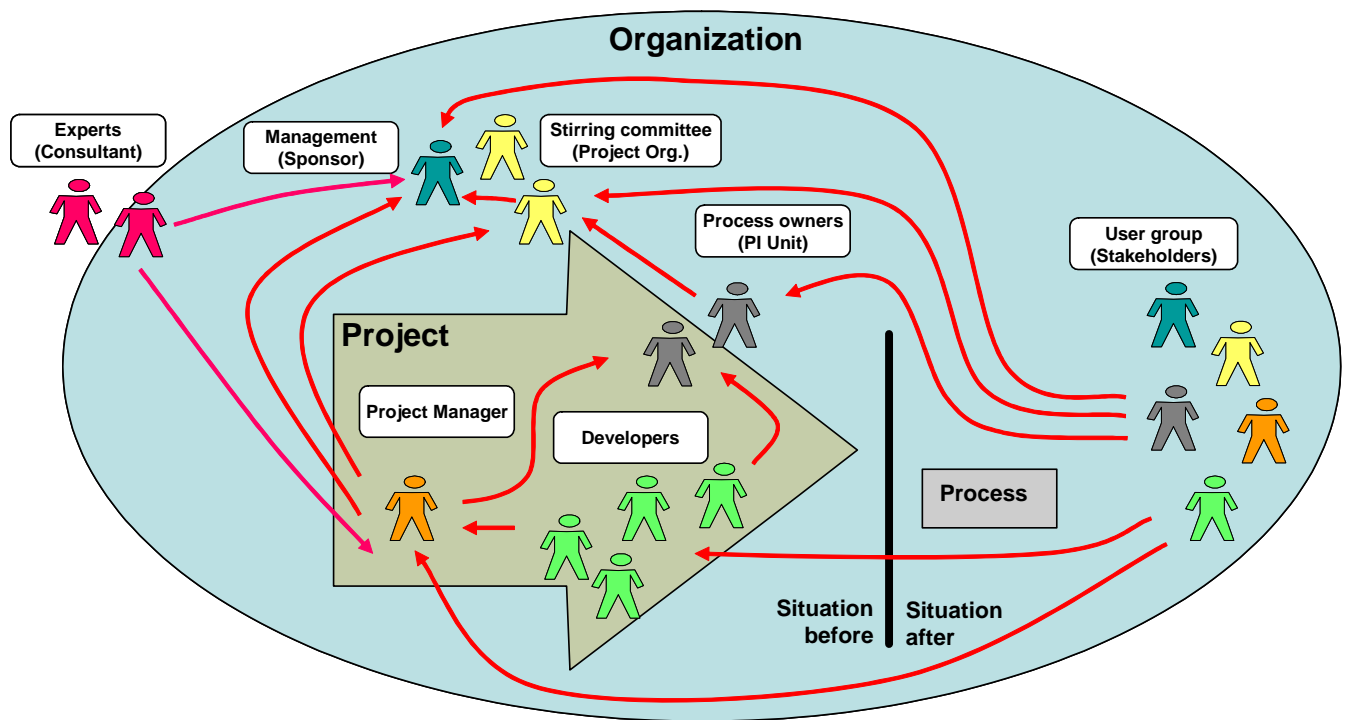


Figure 2. Role model for change – pull of knowledge

To ensure the best possible setup of a process improvement project all stakeholders have to be activated – eliciting information from different sources in the organization to be able to define the requirements, scope and benefits for the project. This is illustrated in figure 2.

From the illustration in figure 2 it is clear, that Users, Process Owners and Management are key roles. It is also clear that the Project Manager is the most important person in that he or she controls the communication path – giving this role has the broadest organizational contact.

### 4 Success

The quest for defining success in IS has been on the agenda in the IS community over two decades, since Peter Keen in 1980 at the first meeting of the International Conference on Information Systems (ICIS) identified it as one of five issues in need of being resolved (DeLone and McLean, 1992).

A lot of effort has been used since then to discuss what constitutes a success? In this research we find different and sometimes opposing viewpoints in the same projects and about the same processes. Thus what is considered a success from one point of view is not a success from another point of view.. In our research we interviewed stakeholders in the same projects but having different viewpoints. The result of the analysis gave the following results.

### **Management / Sponsor**

Management is mainly concerned with the successful implementation of PI in the sense that it proves useful to the organization. They are also interested in whether the new processes are in fact used homogeneously across the business unit. Management focus on the strategic benefits gained from PI. Success is equal to the achievement of the formal objectives – including a focus on cost.

### **Project Organization**

The Project Organization role focuses on quality and productivity as a success criterion. The quality is linked to the final product in terms of classic project management goals: time, quality, costs and functionality. The project Organization also point to success criterion that is linked to the dissemination of knowledge across projects. By supporting PI the Project Organization expects to be less dependent on the individual, achieving a more flexible organization.

### **Experts**

By getting personally involved in the design process the Experts acquaint themselves with the consequences of PI in the organization. Success for this role is constituted by the influence on the design of processes through active involvement.

### **PI Unit (Process owners)**

The PI Unit role is not concerned with the added value of PI as opposed to Management. The PI Unit is concerned with the organizational capability to deliver stable processes based on best practice that increase the productivity of employees in the organization. The PI Unit takes the responsibility for seriously demonstrating the value of improvement initiatives both to Management and employees in general.

### **PI Project Manager**

The PI Project Manager emphasize that it is important that the organization accepts the new processes. It is evident that the PI Project Manager must take into consideration many stakeholder expectations that sometimes are contradicting each other. They must perform within the chartered course set by Management and the Project Organization and at the same time deal with the acceptance at the individual level. On basis of the analysis PI Managers can be characterized as a facilitator for PI effort.

### **PI Developer**

Besides from being in control of the PI work the data reveals that PI developers regard their effort as a success if it enhances the organizational effectiveness. But they are also driven by the anticipation of the entire organization benefiting from their contribution and continuous effort to consolidate processes. PI Developers are therefore concerned with designing processes that is practicable for users. People who enact the role as process developers find it satisfying to contribute to the organization and have influence on the structural design of the organization.

### **User**

Users in general demand that the result of PI must back up daily work activities, if they are going to regard it as a successful improvement. If this requirement is not met chances are that they will find alternative ways of bypassing the defined processes. Users also require organizational wide support to consider PI as successful.

In figure 3 this “landscape” of roles and success criteria is unfolded. The success criteria in relation to the improvement project are listed for each role.

## Session 2: SPI and Communication Management

Role Viewpoint	Supplier			Performer			User
	Management Organization	Project Organization Project	Expert Individual	PI Unit Organization	Project Manager Project	Developer Individual	Customer Project Mngr. Product Devl. All types
Success in project	Strategic alignment Cost Customer and employee satisfaction	Increased Quality of final product Dissemination of knowledge Obtain the mission	Influence Engagement The results are in demand	Inc. productivity Dissemination of best practice Justify PI Improved quality	Satisfying stakeholders expectations Achievement of project objectives	Process design (operationally) Provide value for users Quality	Enhanced support Simple and easy to use processes Help in daily work
Success in Use	More insight Better basis for decisions Better performance	Better control More visibility Better predictability	More involvement Participation in more tasks – have the solution in demand	Gain more appreciation for the mission, accept and continuous PI	Agreement on project objectives Better process basis	Increased quality: less defects Simple & effective processes Best Practice	Demonstrate visible results at high quality

Figure 3. Roles with different success criteria's. Red text can be directly traced back to research interviews. Black text is based on knowledge and experience of the authors.

The rational goals for a project are normally agreed as success factors by all roles in the organization – such as the requirements for the project. The expected goals in relation to the stakeholders are normally related to specific roles.

From figure 3 it can be seen, that the view on success is different from role to role, and different seen from a project perspective and from a user perspective. From the project point of view, the suppliers relate success more in direction of organizational benefits (rational goals). Whereas the performers relate success more in the direction of the end users (expected goals). Seen from a users point of view, success is more role specific (rational goals).

To obtain a more detailed view, we will analyze which needs are in focus for the different roles.

### 5 Different needs and perspectives in relation to roles

Analysis of the project interview data in Talent@IT as well as several years (more then 25) as employee, researcher and consultancy in many primarily Danish companies working with process improvement can be expressed in the picture given in figure 4 of the different roles in a “play of change” game.

Role Viewpoint	Supplier			Performer			User
	Management Organization	Project Organization Project	Expert Individual	PI Unit Organization	Project Manager Project	Developer Individual	Customer Project Mngr. Product Devl.
Horizon	Vision and strategy	Mission & objectives	Tasks	Mission	Objectives	Tasks	Tasks
Focus	Improvement Economy and business development	Overview and follow-up - scope and management	Deliver value -The knowledge the organization is missing	Consistency - structure and quality	Plan, budget and results - control	Solutions, methods, techniques and career	Solutions, efficiency, quality and security
Problem	Missing basis for decision making	Missing control and security	Missing continuously involvement	Missing trenchancy and accept	Missing stability - moving target - poor capacity	Missing time and quality - rework and overtime	Missing or too poor results on a scanty basis
Solution	More insight: The right Information, data and dialogue	Better predictability: Higher maturity	More usable services and in demand: More intervention	Process improvement: Organizational qualify	Operational processes: Methods and techniques	Qualify: Competence rewarding education	Product improvement: Efficient solutions

Figure 4. Roles with different perspectives - focus, problems and solutions

This picture of the main roles view on horizon, foci, problems and solutions in figure 4 illustrates a clear difference between the roles.

The relation between problems, solutions and success criteria is obvious – and also natural. The performers are more focused on running the project effectively and maturely, minimizing related problems, increasing quality and improving qualifications. The suppliers on the other hand are more focused on structure and control, overall productivity and quality – a broader organizational view.

The characteristics of the perceived problems are very different. One of the keys to a more successful improvement initiative in an organization is to focus on this difference. If you have one role for example as performer then the less you focus on your “own” problems the more typical conflicts will be. It is clear, that solved problems for one role can help to solve problems for other roles.

Let us take an example. The Management problem “Missing basis for decision making” can be decreased by a solution more insight, better predictability (solution for Project Organization) and process improvement (solution for Process Owners).

A main conclusion is that insight in other roles problems and potential solutions are very important for success. It is also clear, that the solutions in fig 4 directly are linked to the corresponding role successes.

### **6 Using the *ImprovAbility™* model to give the necessary insight**

We used the interview data as a whole to build an *ImprovAbility™* model, which is a model that can be used to measure an organizations or a projects ability to succeed with improvement. After having build the *ImprovAbility™* model we tested it in real life in the organizations participating in the Talent@IT project

The core assumption behind the *ImprovAbility™* model is that the parameters identified in the interviews on success and failure projects can be used to identify an organizations ability to improve by encouraging activity that has shown to be related to success and avoiding activities that has shown to lead to failure.

The ability of an organization to produce success and avoid failure – the ability to improve - depends on the organizations ability in coping with four groups of parameters:

- Parameters related to *initiation* of projects typically ideas for new SPI or Innovation projects
- Parameters related to *projects*, from the very first hour and until a result is taken into use
- Parameters related to results *in use*, from the first user uses the new process or product for the first time and until full deployment
- Parameters related to the enterprise *foundation*

If we identify where the different roles have their main importance or are most powerful we will get the picture below in figure 5.

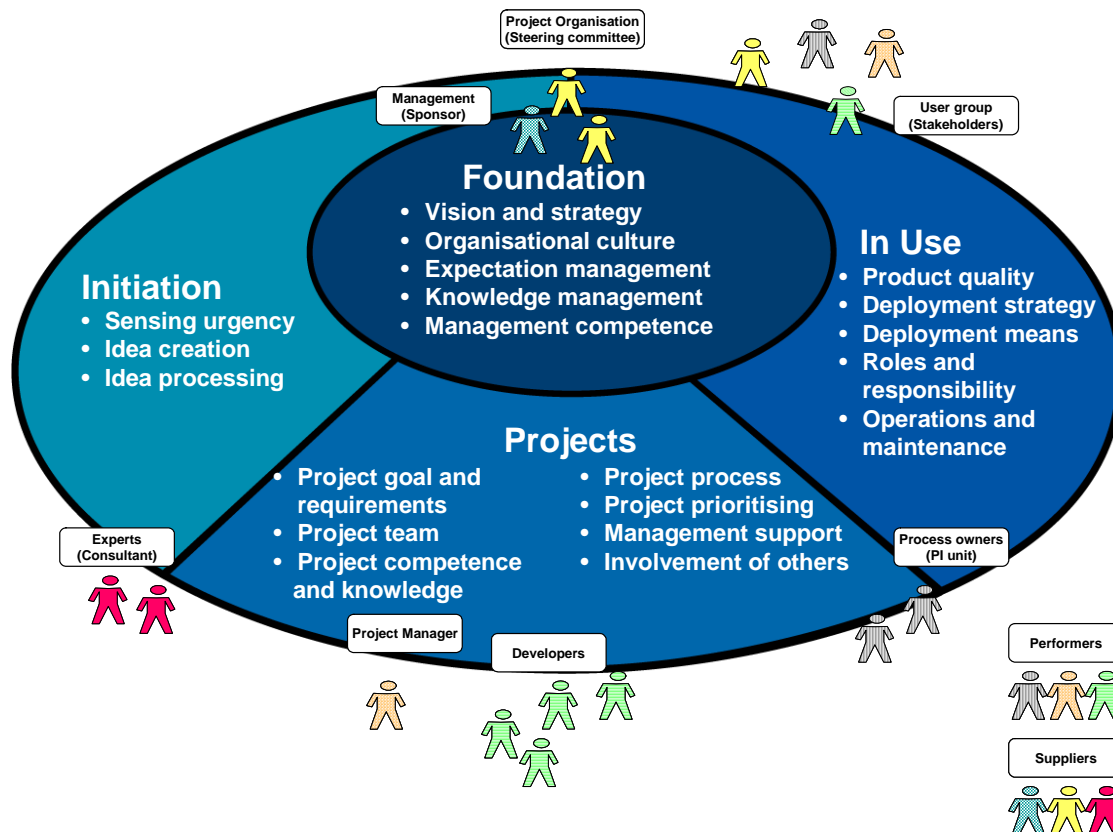


Figure 5. The *ImprovAbility™* model, including main roles

From the picture in figure 5 we can see that the suppliers are located primarily in the initiation and near the In use categories. The performers are tightly coupled to the project. The explanation of the identified roles is easy to explain in relation to the model categories, which also gives a link to the parameters in the model – which parameters are of most interest for the specific roles.

## 7 Conclusion

This paper has identified three major roles in three different organizational settings. Analysis revealed that the view on success, even in the same projects and about the same things, were quite different.

For each role we identified key foci, problems and solutions and we discussed how they were related. We also hypothesized that if you enact one role but show empathy for the problems related to other roles you can increase the change of success.

Finally we showed how the *ImprovAbility™* y model may be a concrete way to identify the roles, problems and solutions leading to success in a concrete organizational setting.

For more information on practical use of the *ImprovAbility™* model and method the reference Høeberg et al. (2006) is the one.

## 8 Literature

1. Pries-Heje, Jan (2003). Role Model for the Organisational IT Diffusion Process. Proceedings of the Fifth IFIP 8.6 Working Conference, on The Diffusion and Adoption of Information Technology, 6<sup>th</sup>-8<sup>th</sup> October 2003, Elsinore, Denmark.
2. Eichen, Kim & Dreyer, Tina R. Succes med SPI – Starter hos projektlederen. Talent@IT midtvejsrapport. ISBN ...



3. Elisberg, Thomas; Hansen, Galina & Hansen, Nikolaj Grønning. A Key Success in SPI – Mapping Stakeholders' Success Criteria.
4. Pries-Heje, Jan; Johansen, Jørn (2005). AIM – Ability Improvement Model. Proceedings of the 12<sup>th</sup> European Conference, EuroSPI 2005, Budapest, Hungary, November 2005. Springer. ISBN 3-540-30286-7.
5. Goldenson, Dennis R. & Hersleb, James D. (1995). After the Appraisal: A systematic Survey of Process Improvement, its Benefits, and Factors that Influence Success. Technical Report CMU/SEI-95-TR-009. Software Engineering Institute, Carnegie Mellon University, Pittsburgh
6. Mathiassen, Lars; Pries-Heje, Jan & Ngwenyama, Ojelanki (2002). Improving Software Organizations – From Principles to Practice. Addison-Wesley, ISBN 0-201-75820-2.
7. Dybå, Tore. (2000). An Instrument for Measuring the Key Factors of Success in Software Process Improvement. Empirical Software Engineering. 5. P. 357-390. Kluwer Academic Publishers. The Netherlands.
8. Høeberg, Anna; Olsen, Klaus (2006). *ImprovAbility*<sup>TM</sup> at the Project Level. EuroSPI 2006, Joensuu, Finland, Oktober 2006. Springer.

### 9 Internet references

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

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

IT-University of Copenhagen: <http://www.itu.dk>

### 10 Author CVs

#### Jørn Johansen, DELTA

Jørn Johansen is the manager of the Department IT-Processes at DELTA. He has an M.Sc.E.E. from Ålborg University and has more than 25 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 11 years he has worked at DELTA as a consultant and registered BOOTSTRAP, ISO 15504 lead assessor, and also CMMI assessor. He has participated in more than 40 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 than 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 Hørsholm; Denmark; Phone +45 72 19 40 00

Fax +45 72 19 40 01 E-mail : [joj@delta.dk](mailto:joj@delta.dk)

### Jan Pries-Heje, IT-University Copenhagen

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.

Jan Pries-Heje, Can be contacted at IT-University of Copenhagen, Rued Langgaards Vej 7; DK 2300 København S; Denmark Phone +45 72 18 50 00 Fax +45 72 18 50 01 E-mail: [jph@itu.dk](mailto:jph@itu.dk)



# Measurement practices in Fidenta

*Erkki Savioja & Markku Tukiainen*

## **Abstract**

Measurement is practiced in large number of software organizations. Measurement may have many purposes, e.g. it can have a business goal orientation and software improvement orientation. It is an essential part of the processes of a software company, one of its' benefits is that it can be used for customer satisfaction and trust building to show the value creation that the quality work has for the customer. Measurement alone will not improve the software process, but it is a necessary step to collect information of the state of affairs and to evaluate the effort in software improvement activities. In this paper, we will describe a measurement program in effective use in a medium size Finnish software company, which combines in its measurement procedures business goal orientation and a dimension of software process improvement from perspective of engineering processes.

## **Keywords**

Measurement, software process improvement, management

### 1 Introduction

Measurement has a definite role in a software company's quality improvement work [2]. Software Process Improvement (SPI) has to be continuous and it cannot be successful without systematic monitoring and measurement [3]. Measurement is practiced in large number of software organizations, but at the same time many organizations are having difficulties in establishing and maintaining their measurement programs [1]. This article describes a measurement program run in practise, and evaluates the benefits and downsides of the measurement.

The objectives for the measurement programs are wide and highly dependable of the state and the capability of the company's software processes [8]. Measurement activities are considered successful if they help project stakeholders first to understand what is happening during their processes and second, to control what is happening on their projects. Beside these organizational and internal values, there is a benefit to show the competence of the organization externally. This increases the organization's competitiveness at the software market.

Measurement is an essential part of the processes of a software company, one of the benefits is that it can be used for customer satisfaction and trust building to show the value creation the quality work has for the customer. Measurement alone will not improve the software process, but it is a necessary step to collect information of the state of affairs and to evaluate the effects of software improvement efforts.

This paper presents the measurement program run in a middle size Finnish software company in the following way:

Section 1 describes the company and the background for the measurement program.

Section 2 defines the measurement from the management perspective.

Section 3 describes the measurement program in practice.

Section 4 describes the utilisation of the measures in the company.

Section 5 draws some conclusions of the measurement work.

### 1.1 Company facts

#### 1.1.1 TietoEnator Group

Fidenta is a part of TietoEnator Group. With over 15 000 experts, TietoEnator (TE) is one of the largest IT services providers in Europe.

The Group has chosen to focus on areas where it has the deepest industry expertise. The principal ones globally are banking, telecom, healthcare and forest. In these areas, TietoEnator works hand in hand with many of the world's leading companies and organisations and is growing with them and is now active in more than 25 countries.

TietoEnator is strengthening its international competitiveness by harmonizing its operating model, processes and services globally.

There are common, group level measures concerning harmonised processes and own, business area related measures for sub-business units.

Continuous improvement is an essential part of TietoEnator's process culture. Common measures are based on experience gained in various countries and customer environments.

### 1.1.2 Fidenta Oy

Fidenta is the joint venture owned by TietoEnator and Nordea Bank. The structure of company's partnership model is described in figure 1. Fidenta belongs to TietoEnator Group and Nordea Bank is Fidenta's only customer. Fidenta operates on TietoEnator's Banking and Insurance area, which is one of the five vertical business areas of the Group.

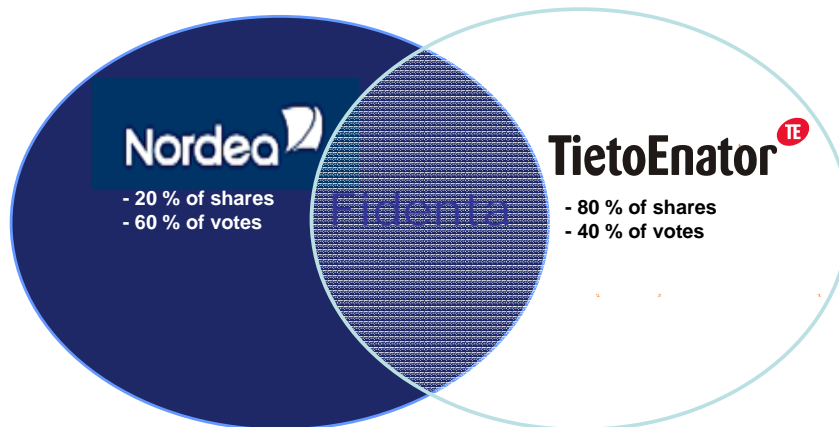


Figure 1. Joint venture structure of Fidenta

Value adding partnership has been targeted in Fidentas by the arrangement, where customer has a majority of votes and TietoEnator has a majority of shares. The basic idea is that customer could concentrate on what they want IT to do, while Fidenta makes it possible by taking care of developing innovative IT solutions that realize the visions of the customer.

Effectiveness is produced through the win-win situation, where cornerstones are close cooperation, trust to each other and high transparency between the customer and Fidenta aiming to a seamless end-to-end service chain helping the customer to manage and run its business better.

Fidenta provides systems development, integration, consultancy and application service management services as a package tailored to customer's needs.

The internationalisation processes of both owner companies have initiated big changes also to Fidenta's operational environment and Nordic level working procedures have become a part of Fidenta's everyday life.

Fidenta has done systematic software process improvement (SPI) more than ten years and deployed solutions are based on best practices and utilisation of experiences. This is in line with measurement requirements, because collected data enables an evaluation of current situation and aligning of development efforts to preferred direction.

Utilisation of best practices is also targeted by the use of process maturity models (CMMI, SPICE) and other standards (ISO9001:2000). Project manager certificates of IPMA (International Project Management Association) are a part of Fidenta's training program for project managers.

### 2 Measurement from management perspective

Collecting improvement data and performance measures was planned as a part of the development of company's business system and process descriptions. Measurement results are linked to the annual action planning and usefulness of measures and how good they fit to management needs are evaluated regularly e.g. in EFQM self-assessments.

The basic target is that decisions should be based on facts. To fulfil this target, information is produced through monitoring and measuring current performance and it is utilised in planning and aligning future actions aiming to get preferred changes to happen.

Measurement is a part of the steering and control system of the company. Target levels for measures are set up in the annual action planning process by Strategic Management Team (SMT), which consists of the managing director, four business unit managers and the manager of Consulting Services unit. Action plan is published in the intranet. The organisation of Fidenta is presented in the figure 2.

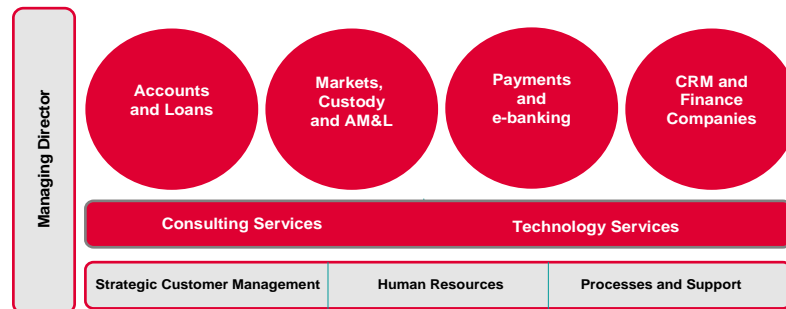


Figure 2. Fidenta's organisation

Most of the measures are created on monthly bases; some are gathered quarterly, some bi-annually and annually. Core measures are monitored in bi-annual management quality reviews, which are carried out by Quality Manager in the Development Management Team (DMT), which consists of managing director, four business unit managers, managers of Consulting Services, Technology Services, Strategic Customer Management, Human Resources, Processes and Support units, Quality Manager and a representative of the employees. Measures are accumulated to history data annually and new trend charts are created.

A part of measures are used as an alarm system. Actions are taken only when results deviate too much from expected ones. Measurement results out of expected limits set up analysis phase in action aiming to find actions to get performance back to the acceptable level.

Measures are used also for follow up and controlling that expected changes in behaviour have really happened. For example some new procedure has been launched and management want to get evidence that it really has been implemented as a part of every day life.

Decisions concerning measurement and quality assurance in general are taken by Fidenta's DMT.

Within the launch of harmonised processes, TietoEnator has launched also some common measures linked to each process.

### 3 Running of the measurement (in practice)

#### 3.1 General

Measurement results are utilised in software process improvement in Fidenta i.e. different performance dimensions are monitored through measures in order to make the required enhancements accordingly. BSC is used to outline measurement areas as described in a figure 3.

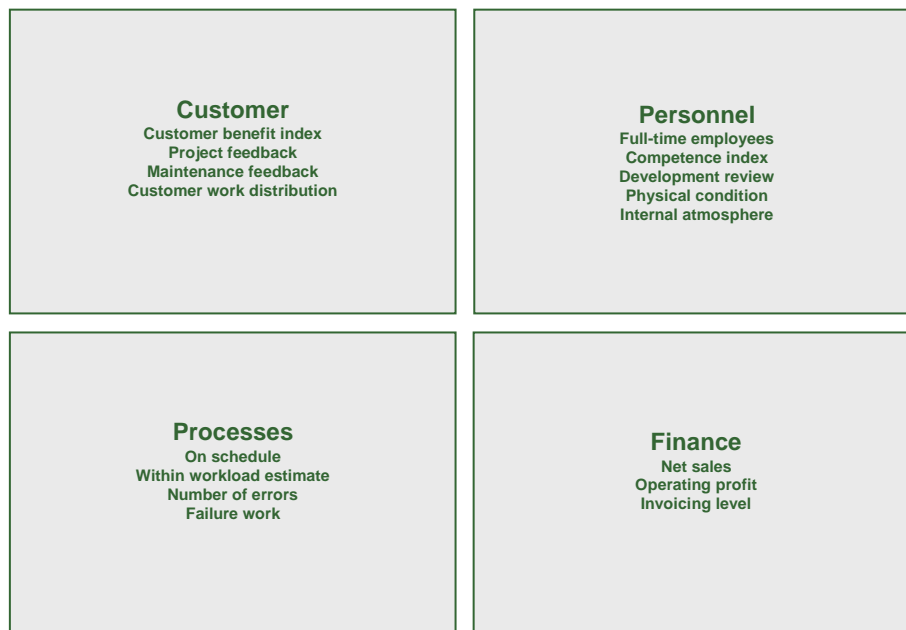


Figure 3. Fidenta's BSC measures

Besides here described core measures Fidenta runs also e.g. java code HTML reviews, which are also kind of measures giving direct impulses for developers to line their behaviour according to agreed standards. But we don't regard these procedures as measures, but they are considered more like quality reviews.

Measurement results are available in the intranet as well as descriptions and annual trend charts.

#### 3.2 Examples of core measures

##### 3.2.1 Customer area

###### Customer benefit index

Fidenta's Managing Director collect data for this measure by interviewing customer's top management according to the twelve question survey form. Average of each question is a score, which values are monitored over the time

###### Project, maintenance and consultant feedback



Collection of data for these measures is done by specific enquiry forms. For part of the questions is expected written answers, parts are a tick to a box type. Responsible person for delivery in Fidenta sends feedback form to the stakeholders on customer side. All scores are analysed on sub-unit and company level. Quality Manager takes care of company level summary quarterly and results are also part of bi-annual Quality Review for management.

### 3.2.2 Processes area

#### **Keeping the timetable, keeping the work load estimation in project deliveries**

These measures indicate how accurate estimations have been given. They are calculated from planned and actual figures. Results are reported as percentage figures. Connected to these measures number of project deliveries, percentage level in keeping timetable and work load and their standard deviations are monitored.

#### **Number of errors**

This measure monitors the correctness of deliverables. All the failures in production, which are classified to be due to programming errors, are counted monthly. Connected to this measure is calculated also a percentage of work hours used to corrections of failures out of total customer work hours.

#### **Correctness of deliverables**

This measure is quite new. Original data is in the work hour reporting system. Collection is automated. This measure monitors the relational proportion of total project hours, which is used during acceptance testing for fixing the defects. The figure is weighted to the scope of the project.

### 3.2.3 Personnel and finance areas

Value Creation Capital from Employee satisfaction survey is one measure on personnel area. On finance area measures are same as commonly used in all companies.

## 3.3 Other measures

Besides core measures there are 15 other regularly and systematically counted and monitored measures related e.g. to competence profiles, training hours/ euros and language skills.

## 3.4 External assessments

#### **ISO Quality certificate follow-up audits**

Fidenta got the ISO 9000:2000 quality certificate in 1997. It was upgraded to ISO9001:2000 in 2003. The quality certificate requirements are monitored in annual follow-up audits, which have been run as combined CMMI assessment during last two years.

#### **CMMI assessment**

Fidenta has carried out CMMI assessments since 2004. The assessments have been CMMI B-type assessment with the continuous set of the model. In assessor team have been three external competent CMMI assessors plus two or three internal TE-CMMI Lead Assessors. In 2005 Fidenta reached maturity level 2, five processes were on capability level 3.

Results of CMMI assessments have been cornerstones of planning and implementing Fidenta's development efforts in 2005 and 2006. The assessment report has been analysed thoroughly in the Quality Team. The proposal for internal development projects and improvement efforts has been created. The proposal has been presented to the DMT for approval and decision of actual efforts taken.

The focus in choosing development efforts has been on how good they enhance an achievement of company's business goals. Model requirements have always been only a secondary cause.

According to the improvement roadmap Fidenta is aiming to get its project and application service management processes on CMMI capability level three. The target is based on the mapping between these core processes of Fidenta and CMMI processes. The roadmap is based on Fidenta's business goals and the analysis of assessment 2005 results.

### 3.5 Self assessments

Fidenta has carried out EFQM self-assessment five times since 1997 and the intention is to run it every other year. Fidenta took part to Finnish quality award competition in 2001, but did not win the service provider series. Scores has improved from about 350 up to 511 points.

Assessments have always been conducted by an external consultant specialised to EFQM model to ensure the correct evaluating level. Assessments have been carried out mainly by a group of all management group members added by few representatives of different work roles in Fidenta.

EFQM assessments have produced a lot of improvement initiatives, which has been very much in line with other improvement proposals (e.g. CMMI assessments, ISO 9001:2000 follow up audits). They have been analysed together with improvement proposals from other sources and utilised in the annual action planning process.

## 4 Utilisation process of measures

### 4.1 Utilisation elements and continuous improvement

Measures are used for monitoring purposes, to ensure that performance is in line with stated targets. Our approach has been a goal-driven measurement where business goals are interpreted into measurement goals [5]. In organization wide development, the purpose has been in Balanced Score Card measures, which could be applied through different types of businesses and sub-organizations. Often this has resulted in emphasising the financial and management control-type measurements.

But running measurement is seen more profitable for the organisation while it is used in directing the development efforts. It produces data to action planning and creates input to continuous improvement process. In many cases there is a disagreement between the developers and managers of how beneficial the measurement activities are for the organization. In the cases where the developers perceive the measurement practice will have positive impacts on their productivity and product quality, the improvement actions in measurement are well adopted [7].

In Fidenta the measurement program has been well taken by the whole personnel. The measurement results are used in many types of situations. For example the measures for customer satisfaction index and project feedback index are evaluated regularly. Core measures are handled in the board meeting quarterly. In the board there are representatives of both owner companies and the customer side has the majority of votes.

Fidenta's Quality Team is a virtual team; the structure of QA parties is described in the figure 4. The basic role of QA Team is to be a two directional communication channel between management and business unit personnel. There are no full time members in the team; business unit representatives are recycled every now and then and the QA Manager is a part time project manager too.

All employees are targeted to get familiar with a collection of improvement ideas and encouraged to give their impact.

Experience utilization and learning loop have been identified as important success factors of the com-

pany. The Quality Team analyses improvement data collected from different sources: measurement results, initiatives from personnel through QA Team members, internal and external audits, testing, risk analysis and self assessments. The process is transparent for employees through the intranet, where proposals and decisions are available. After implementation effects are evaluated according to continuous improvement concept.

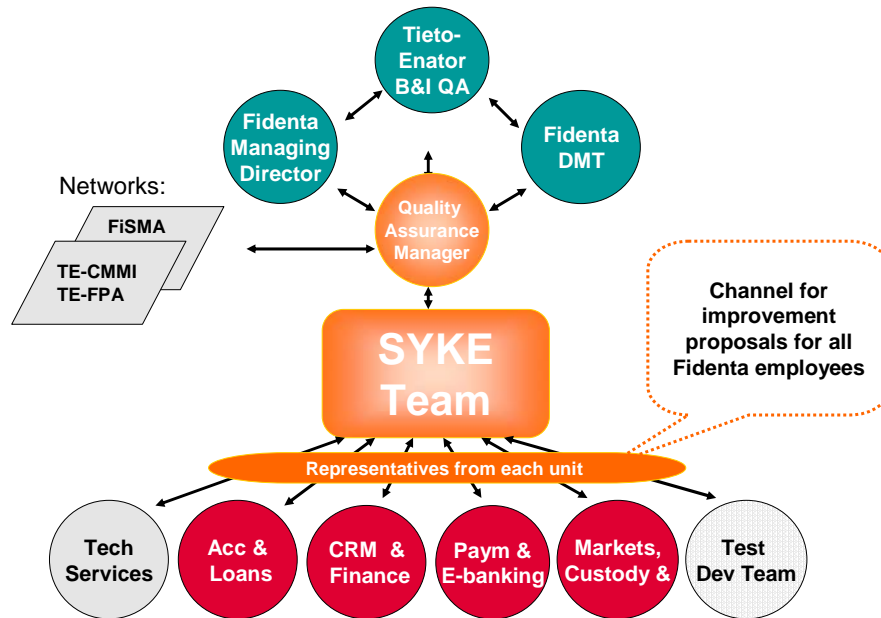


Figure 4. Fidenta's QA organisation

## 4.2 Lessons learned

According to our experiences running the measures have had a positive influence to the development of the company. Concerning the usage of measures here are listed some areas where quality practices and measurement results have been used.

### Communication

- Open communication, transparent operations of improvement proposals and analysis of assessment results activate employees to take the initiative and give their impact to development efforts
- Employees' attitude towards measurement is more positive, when they are aware of the link between improvements in everyday life and measures
- Management commitment is a must

### Benchmarking

- Collection of measurement data enable benchmarking
- Utilisation in customer relationship management
- Motivation effect on internal improvement efforts
- Some measures are common for all units on TE's Banking and Insurance business area, so performance could be quarterly compared to the benchmark data of other units

- Effect on internal company image

### Maturity models

- Help to outline development activities; to take more conscious steps towards more capable processes
- Interesting point in 2005 was that same needs for improvement efforts came up in both CMMI and EFQM assessments; in this case models from two different viewpoints seem to speak same language (people concerned was in self-assessment mainly managers and in maturity assessment developers, project managers and testers)
- Our experience is that two dimensions of development sources: business goals and model requirements are not necessarily so far from each other

### Convincing customers/ employees

- Company gets competitive benefits by applying systematic processes; long trend values of different measures ensures for customers a picture of reliable partner with good quality functions compared to companies, which could not point out any measurement data of their performance
- Logically run measurement improves also employee satisfaction and have positive impact on the internal company image

### Bonuses

- Some measures e.g. operating profit and a level of customer feedback has been used as criteria on employee bonuses

### System improvement/ maintenance criteria

- System responsible people use the measure "Percentage of work hours for fixing production errors" while negotiating with the customer about the development needs of a system/ application

### Virtual quality team

- One experience to share is that credibility is higher, when quality team members themselves do also project work; this make it easier to get personnel's acceptance for proposed changes

### Analysis on sub-unit level too

- A company level analysis of measures is not enough, because a lot of details are lost. That's why we have spread the analysis on a sub-unit level. Quality Team members collect bi-annual quality reviews of their sub-unit and present them in department meetings. For instance evaluations for a certain question on project feedback enquiry might be ok on company level, but some sub-units are not performing well enough, so it is possible to direct improvement efforts correctly

### Worst performing systems

- Top-twenty list of worst performing applications was proven to be an effective way to reduce production failures. An unacceptable high number of working hours used to be spent to correcting production errors instead of project or development work. The action was a weekly meeting with management and application responsible people to go through the top twenty worst performers, experiences were shared and needed actions were discussed and decided to get performance on a right track. Wanted effects was reached and this measure has been constantly on acceptable level afterwards

### Needed efforts for measuring

- A measure could be effort consuming to implement, but it must be easy to run

### Recruitment

- Measurement practices has effect on preferred employer opinions; many people interested to work in Fidenta do appreciate systematic working procedures, which are indicated by long time series of measures

### **5 Conclusions**

Measurement is practiced in large number of software organizations. It is considered as essential part of Software Process Improvement (SPI) work. Despite this many measurement programs tend to fail at some point. It has been argued that one of the reasons for this could be the differing opinions about the focus of measurement goals, e.g. towards business goal orientation or software process improvement. In this paper, we have described a measurement program in effective use in a medium size Finnish software company. The connection of business goal orientation and software process improvement from engineering processes' view point can be successfully combined [8]. This is what has been happening at Fidenta.

One of the interesting future research questions would be how to maintain the improving trend that the measurement program has created. There are the motivational questions of how not to merely maintain the status-quo but to innovatively improve using the measurement data and processes. There is also a question of how to change the measurement practises and methods when the software processes change. There has been some positive evidence of using process modelling with automated tools to integrate measurements into the developing software processes, see e.g. application of the Framework for the Modelling and Measurement for Software Processes (FMESP) [6].

Another important question for future is, from financial point of view, how to get competition benefits out of measurement process in most effective way.

### 6 Literature

- [1] Goethert, W., Fisher, M. (2003). Deriving Enterprise-Based Measures Using the Balanced Scorecard and Goal-Driven Measurement Techniques, October 2003 , Technical Note, CMU/SEI-2003-TN-024.
- [2] The SPIRE Handbook (1988). Centre for Software Engineering, 1998.
- [3] Fenton, N.E., Pfleeger, S.H. (1997). Software Metrics: A Rigorous & Practical Approach. 2.Edition, International Thompson Computer Press, 1997.
- [4] Wiegers, K. E. (1999) A Software Metrics Primer, *Software Development*, July 1999.
- [5] Goethert, W., Hayes, W. (2001). Experiences in Implementing Measurement Programs, November 2001, Technical Note, CMU/SEI-2001-TN-026.
- [6] Canfora, G., Garcia, F., Piattini, M., Ruiz, F., Visaggio, C. (2005). Applying a framework for the improvement of software process maturity. *Software—Practice and Experience* 36(3), 283-304, 2005.
- [7] Green, G., Hevner, A., Webb Collins, R. (2005). The impacts of quality and productivite perceptions on the use of software process improvement innovations, *Information and Software Technology* 47, 543-553, 2005.
- [8] Trienekens, J., Kusters, R., Rendering, B., Stokla, K. (2005). Business-oriented process improvement: practices and experiences at Thales Naval The Netherlands (TNNL), *Information and Software Technology* 47, 67-79, 2005.

### 7 Author CVs

#### **Erkki Savioja**

Fidenta Oy  
Nihtisillantie 3  
P.O. Box 24  
FI-02631 ESPOO  
[Erkki.Savioja@tietoenator.com](mailto:Erkki.Savioja@tietoenator.com)

Erkki Savioja has over 20 years of experience of working with banking software in different roles. Currently he is Fidenta's Quality Manager, the driver of Fidenta's Quality Team and a member of Fidenta's Development Management Team. Measurement, project management and software process improvement are among his interest areas and he has been Fidenta's representative from 1998 in the network of Finnish Software Measurement Association (FiSMA) He has graduated from Turku University computer science as a main subject.

#### **Markku Tukiainen, Ph.D.**

Department of Computer Science  
University of Joensuu, P.O. Box, FIN-80101 Joensuu, Finland  
<http://www.cs.joensuu.fi/~mtuki>  
[Markku.Tukiainen@cs.joensuu.fi](mailto:Markku.Tukiainen@cs.joensuu.fi)

Markku Tukiainen is a professor of Software Engineering in the Department of Computer Science at the University of Joensuu, Finland. His main research interests in software engineering are in Software Process Improvement in the small and medium size enterprises and the development of Certification of Software Engineering Professionals. Research of SPI in SME's involves a very close connection with software companies and this has been established through a SPIN network called tSoft in the city of Joensuu area ([www.cs.joensuu.fi/tSoft](http://www.cs.joensuu.fi/tSoft)). In Certification of Software Engineering Professionals the objective is achieving and promoting a globally accepted benchmark for the certification of software engineering professionals through an ISO standard development (ISO/IEC JTC1/SC7/WG20).

# SPI of the Requirements-Engineering- Process for Embedded Systems Using SPICE

*Alexander Poth*

## **Abstract**

Complex embedded systems need consequent and complete requirements engineering. The embedded system must specify and separate into modules which are developed by the suppliers. For effective specification an adequate process with methods and tools must be set up. This paper describes the introduction of this process for an automotive OEM which was designed by MB-technology.

## **Keywords**

SPI (Software Process Improvement), SPICE (ISO 15504), Requirements Engineering, Product Lines, Development Model, Supply Chain, Embedded Systems, Automotive



## 1 Introduction/Context

This paper focuses on two aspects. One is the method for the requirements engineering, the other is the process which will be used to handle the requirements engineering method. The context of the project in which we introduced the methods and process is an automotive OEM. With typical challenges: select an individual vertical integration level for each of the different sub-systems or elements of the system, separate the system in modules and integrate them (define the interfaces), manage the product variability by (software) product lines, a development process conforming to SPICE.

## 2 Requirements-Engineering

On a high level view the requirements-engineering is an iterative, incremental and recursive thing. Figure 1 describes this in a graphical form. Iterative means that a function/feature which is implemented in the first iteration can be added or fine-tuned in the next iterations. The incremental part is that not all functions/features are implemented in the first iteration. By prioritization and release planning the functions/features are realized. The recursive part is that the OEM and the suppliers are making their own requirements-engineering phase. This requirements-engineering is focused on different abstraction levels and different outputs but is done by all parties. The described model is compatible to the V-Model and the [W-Model] and includes aspects of the [RUP].

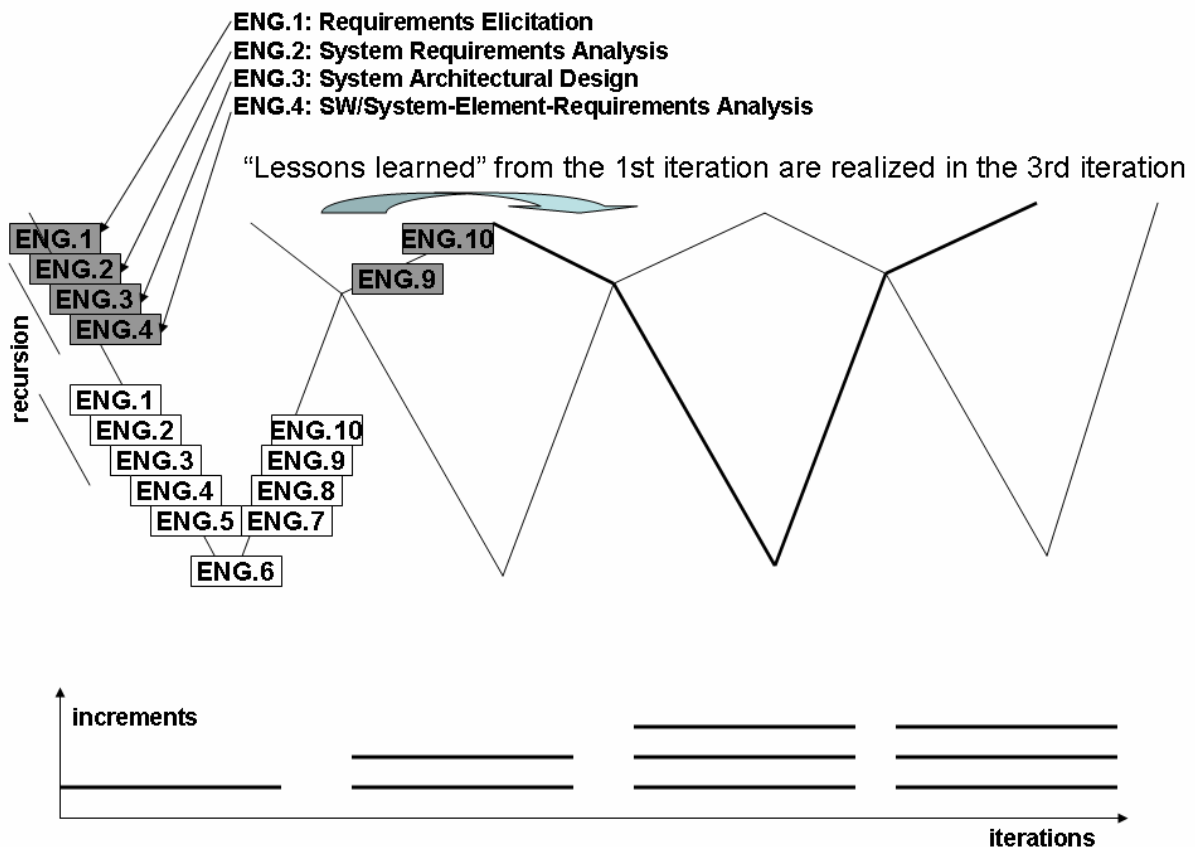


Figure 1: Description of the iterative, incremental and recursive part of the requirements-engineering in the “Diamond-Model”

The grey boxes relate to the OEM and the white ones are done by the suppliers. There is no limitation for iterations or recursion but they are effectively time-limited because for each iteration or recursion time is needed and for every iteration or recursion an overhead is generated. This paper is only focused on the ENG.1 to ENG.4 processes which are embedded in the “diamond development model” (diamond is motivated by the V and the “roof”).

In Figure 1 the first iteration of the OEM ends after the second iteration starts. This is motivated by the supply chain because the supplier starts directly after finishing his work of the present iteration with the next iteration. On the OEM side the present iteration is finished with ENG.10 – so an iteration of a supplier is shorter. The start of the next iteration can be defined earlier as described in Figure 1 if the allocation of resources make it possible

In Figure 1 the “lessons learned” of the OEM are based on the results of the ENG.9 and ENG.10 which provokes some changes or optimizations to the initial specification. These findings are detected after the second iteration has started and so they are input for the third iteration.

An important thing is that the OEM starts parallel to the software requirements specification (ENG.4) with the hardware and mechatronic specification because a typically embedded system consists of these 3 engineering domains. Software requirements are often called function requirements by the OEM but in real life the functions are made in software by the supplier – this motivates to apply the ENG.4 to the OEM too.

## 2.1 Requirements Documentation Classification

In this section the requirements documentation which is the process outcome are described and classified by their types and their scopes. The early “beginning points” of information documentation in the documents are evidence for the incremental method. The documents are useful for all embedded requirements which are software, hardware and mechatronic requirements.

### Framework-Specification:

- Defines the project scope and environment by non-functional requirements (NFR) and functional requirements (FR). Results are all constraints and relevant system-requirements
- Limitation of the technical project goals for example innovations which are described by NFR and FR. Depending on experience from other projects the “old aspects” are not of interest. Here the focus is on all critical system elements. The confinement level of the specification is that the project manager has a “good feeling” to set up a project with a fixed budget and deadline (start of production)
- Standard [variation points] examples are vehicle-series, body types or country aspects. Further special variation points like standard or luxury. The result are the main variants
- Quality goals (oriented on ISO 9126) of the product, which influence the technology, costs and milestones

### System-Specification:

- Identification of all system constraints in detail
- Identification of all system-FRs
- Identification of all system features
- Identification of all system-NFRs, focused on impacts for the system-design
- After the design phase: integration of design-specific requirements for example: integration of an old sub-system (without changes to it) is only possible with a wrapper and moving a feature to an other system element

### System-Design:

- All system-requirements are processed in design-views (oriented on IEEE 1471). Outcomes are:

- Functional view: describes the functions and their relations
- Structural view: describes the modules and their relations
- Distribution view: describes the relation between functions and modules
- Other optional views: for example the safety view for the ISO 61508
- Also relevant in the context of view are:
  - Interfaces
  - Communication matrix
- Consistency between design-views and global optimizations are made
- The abstraction level is chosen for separate the parts of the system elements to the suppliers (here the recursive element of the 1<sup>st</sup> level is defined. OEM-System= sum of all sub-systems = sum of all supplier-systems = sum of all sub-sub-systems = sum of all sub-supplier-systems = ...)
- Integration- and test-aspects are considered
- Components/sub-systems are identified as elements of the system architecture
- Functions are identified
- Functions are mapped to physical system elements and focused on quality goals like ISO 61508
- Interfaces between functions and components for the realization of the system are defined

### Element-Specification:

- Function: the specification of a logical function under technical system context. Signals are described HW-independent, the logical behavior of the function (example: algorithm) is specified and the interfaces are instantiated
  - This is a good start point for product lines because the same function often is used in different contexts. Example: an engine controller is used in all vehicles and is adapted to the “physical” context: 4 or more cylinder, diesel or gasoline, ...
- Component: the “container”-specification of functions, which specifies the physical HW and context for the functions. These contexts are the sensors and actuators and HW-abstraction wrappers
- Sub-system: specification of the interface and the “high-level” functionality of the sub-system. The abstraction level is chosen in order to give the suppliers enough degree of freedom for their own realization concepts
- Interface: mapping of the logical function to physical signals. Depends on the abstraction level the signals are defined down to the “Bit-Level” typically the OEM specified Message-IDs and data-values

## 2.2 Used Requirement-Engineering Methods

In this section the methods are described which are used to set up the requirements-documents. All methods are implemented in the requirements-management-tool [DOORS]. DOORS is for the process context the configuration management and the traceability environment for the requirements.

### 2.2.1 Modularization

On a schematic view the relations between the element-specification-modules are defined in Figure 2. The framework specification, system requirements and system design are unique modules. There are many element modules for the specification of an entire system.

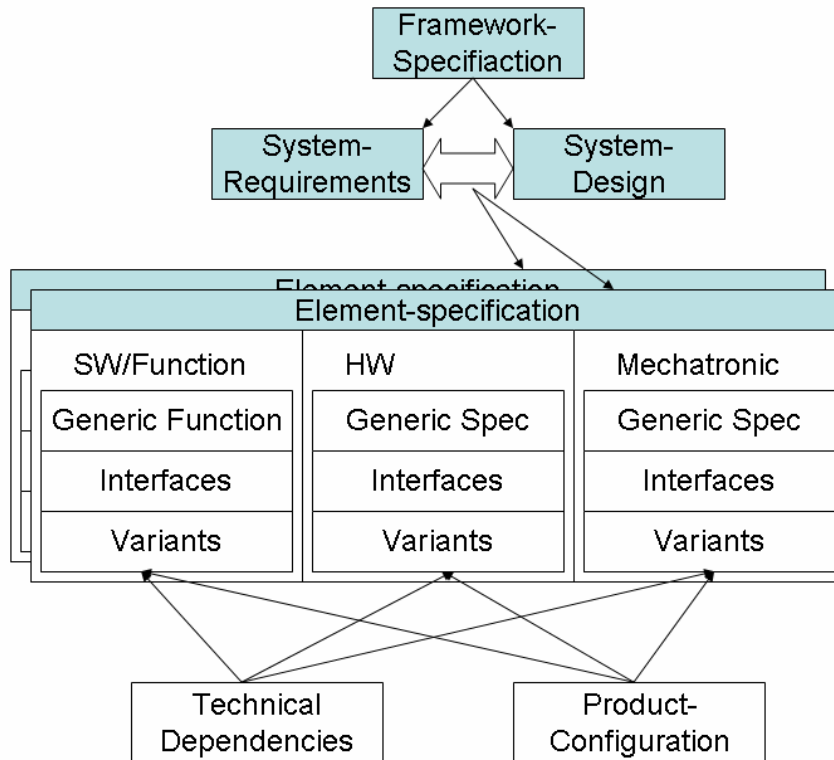


Figure 2: Schematic requirement-modules and their relations.

The configuration handles the product-specific instantiations of interfaces, functions and HW-elements. The dependency-List is an optional element and not described here.

### 2.2.2 Variability Handling

The basic idea of the variant handling in this context is that it is not possible to plan the [product line] at the beginning in detail. To handle this real life observation we can only support a method which is able to react to the new variants. This reaction provokes a refactoring of the present specification. The present specification will be separated in to the [common part], the interface which describes the variant point context and at least 2 variants. One of them is the “old” specific specification part and the other specifies the new variant.

The modularization into requirements modules which handle logical and physical (implementation) aspects make it easy to make this separation for the variant points. Typically the logical part (function) is constant only the realization varieties between the products

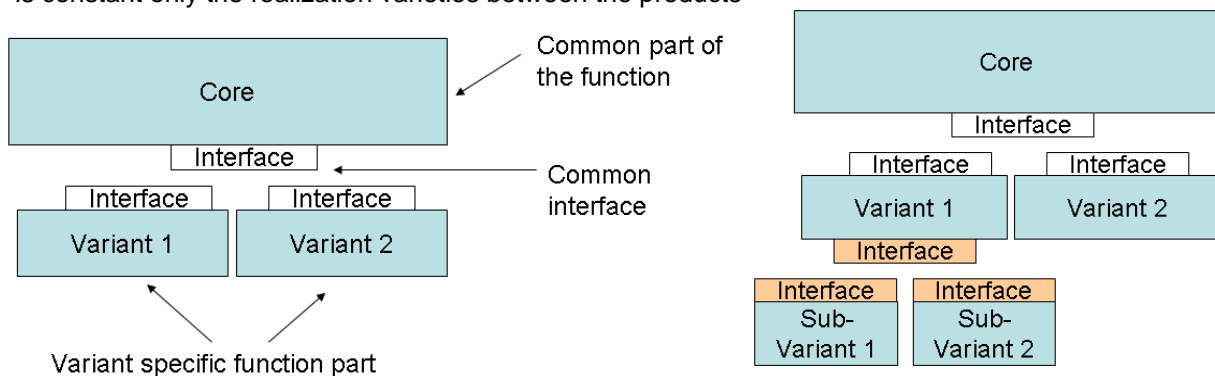
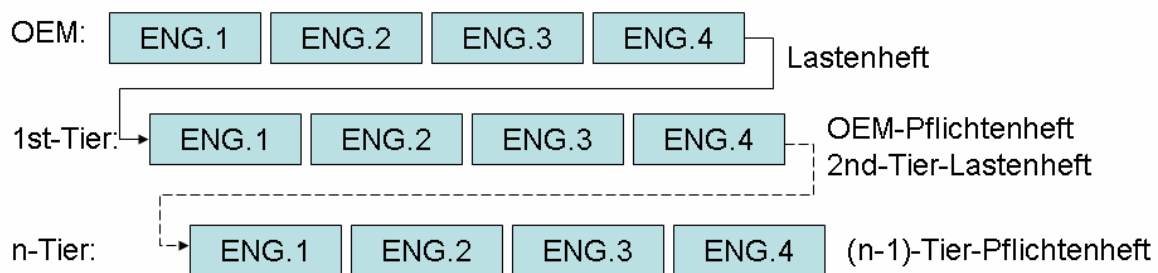


Figure 3: Variants on an interface (interface = context specification of variant point).

Important for this kind of documentation is that the variants are clustered because the presented variant handling method scales with the complexity and do not hide the complexity by e.g. abstraction. An advantage of this method is that the developer documents all variants in detail and if he is not able to manage his variants it is time to “simplify” the products because all variants produce costs in the product life cycle. Observations in real life projects have shown that the products normally are built by a set of standard variation points. If a product sale variant is depending on different modules it is possible to build these too (shown in left part of Figure 3).

### 3 The Requirements-Engineering-Process conforming to SPICE

In the context of automotive manufacturing SPICE is defined in the automotive SPICE which is made by the [SIG]. In this assessment model for the referenced project was chosen the [HIS]-Scope and some other processes like the reuse processes (REU). In the requirements engineering context the ENG.1 up to ENG.4 are relevant.



**Figure 4: Typical recursive specification of a system on different abstraction levels.**

Figure 4 describes the different specifications in the supply chain. In German is distinguished between purchaser specifications ([Lastenheft]) which is focused to define the interfaces and the “what is needed” and contractor answer (Pflichtenheft) which is a specification focused on “how can this realized”. All involved parties have to handle all requirements engineering processes but for example the OEM does not specify the entire software. He specifies the function on a logical level. But this logical level could be an engine operation map which will be set up direct in an array. On the other side a supplier is focused in the ENG.2 not on the entire vehicle he is only focused on a part (sub-system) of the vehicle or and element of the vehicle. But all parties can only set up the Lastenheft at the end of ENG.4. Until all the requirements are specified which are needed to complete the ENG.4 it is possible to add at least one Tier more. The Pflichtenheft could be finished between the end of the ENG.1 to ENG.4 depends on the project focus. At least at end of the ENG.4 the process outcome is the Lastenheft for the next supplier in the chain and the Pflichtenheft for the purchaser. Normally other internal specification documents are built for the part of work which is not outsourced and not relevant for purchaser – this part is the know-how and the business of the supplier.

Our proceeding for the process improvement was:

- Analyze present situation, for reuse of good workflows or process elements
- Define final situation, this could be partially parallel because not all depends on the present situation
- Make a gap analysis to set up process elements, methods and tools to fill the gap

Often you will detect a lot of constraints by analyzing the present situation, but in this context it is masked because it is very specific and is not useful in the context of this paper.

Up to here the development model is defined, the requirements documents are classified, the document contend is defined, the requirements reuse strategy is defined and the tool environment as requirements engineering platform is defined. Now a small overview is given of aspects which are required by automotive SPICE and the realization in the project. The ENG.1 mentioned always “customer” which is do define into the project context. In the organization of the example project is set up customer is defined as person or group (control board) who pay for the development. To obtain the

requirements in the process are defined cyclic status meetings in which the customer can articulate change requests and give some advices for the next steps – a product definition is an iterative thing.

For ENG.1 to ENG.4 the process used a defined data structure in DOORS as communication platform. The traceability is given by the data structure, the prefixes of the requirement-IDs and if necessary by DOORS-links – but here is a rule: less is more – because a jungle of links is not useful. The data/module structure is predefined and by the system design the modules in which the requirements will be hold are defined – by this way the traceability from ENG.1 up to ENG.4 is defined.

For the prioritization which is demanded in ENG.2 and ENG.4 is set up a column in the DOORS data-model to prioritize or plan the releases for functions – a lower level is not useful because a function is the smallest thing that could work. Further, there is a column for change-request-IDs so it is possible to trace every change down to each requirement which was impacted. Also this column is very useful for measurements which impacts have change requests for old and new requirements.

The fulfilment of ENG.2.BP4 is in the process an activity for the design review defined. No design verification method for embedded systems out of the box for this organizational context is available. So an inspiration on [ATAM] was useful – the output is a lightweight checklist and set of questions for the design verification – this is sufficient to satisfy the requirements of SPICE.

For the fulfilment of REU.2.BP2, BP7 and BP8 it is useful to compare the common part and the variants of requirements modules. This is a good indicator for the present reuse and identifies potentials for more reuse.

After set up of the process environment by the process documentation, workflows and tools it was set up a pilot project with “friendly users“. These users are good for lessons learned and if they “provoke” some improvements they are better “multipliers” because now they feel a little like the process designer. After the process adaptation by the pilot project all new projects started with the new requirements-engineering process in the customer company.

### **4 Conclusion/Lessons Learned**

In complex systems the process, methods and tools must be harmonised and aligned to the process standards which was in this example SPICE. An observation is that it is complicate to plan product lines in detail and so it is skilfully to develop a method which supports the refactoring of the requirements like the presented approach which is based on the modularisation and classification of the requirements.

The question of the customer at the start of the improvement project was: is it better to align the processes on SPICE or CMMI? – The sponsor for the improvement project is the project manager of a big cross-function project. This motivates SPICE because it is basically focused on projects and not on organisations and one further benefit on SPICE is that it is an assessment model so it supports better assessments of sub-projects in the line.

The orientation on SPICE was very useful but the orientation of SPICE on software always has extended to the hardware and mechatronic. This embedded system engineering is better focused in CMMI and the next step is to extend the processes to this model.

The process improvement was a good point to introduce the new methods and tools which were demanded by the developers. This coupling by the introduction makes it easier to set up processes because nobody has something to change on the tool environment if the tool environment is new.

For the project the processes are useful to separate the work into different task and activities based on the process outcomes which are required by SPICE – before the project manager had to define the activities and the outcomes – now this is often directly made by the process.

### **Literature**

[ATAM] [www.sei.cmu.edu/publications/documents/00.reports/00tr004.html](http://www.sei.cmu.edu/publications/documents/00.reports/00tr004.html)

[common part] N. Heumesser und F. Houdek. Towards Systematic Recycling of System Requirements. International Conference on Software Engineering 2003 (ICSE'03), 512-519.

[DOORS] <http://www.telelogic.com/corp/products/doors/index.cfm>

[HIS] <http://www.automotive-his.de/processae.htm>

[Lastenheft] M. Weber, J. Weisbrod: Requirements Engineering in Automotive Development: Experiences and Challenges. At International Requirements Engineering Conference 2002 (RE'02)

[product line] R. Kolb, D. Muthig, T. Patzke, K. Yamauchi: A Case Study in Refactoring a Legacy Component for Reuse in a Product Line. International Conference on Software Maintenance 2005 (ICSM'05), 369-378

[RUP] <http://www-128.ibm.com/developerworks/rational/products/rup/>

[SIG] <http://automotivespice.com>

[Variation Point] S. Bühne, K. Lauenroth, K. Pohl, M. Weber: Modeling Features for Multi-Criteria Product-Lines in Automotive Industry. Workshop on Software Engineering for Automotive Systems (SEAS), at ICSE 2004

[W-Modell] A. Spillner: The W Model: Strengthening the Bond Between Development and Test. SQE STAR EAST 2002

### **5 Author CV**

#### **Alexander Poth**

He studied computer engineering at the Technical University of Berlin and the Universidad Politecnica of Madrid.

Since 2003 he has been working on Requirements-Engineering methods and concepts. In 2004 he qualified as an iNTACS SPICE assessor and in 2006 as an iSQI Professional for Project-Management.

Since 2004 he has been working for MB-technology as a Quality-Management-Consult and Engineer with focus on automotive Electric/Electronic-Systems and automotive Software. Previously he worked as a software developer from 2000 for a telecommunications OEM and an ERP-Systems OEM.





# Extending the Rational Unified Process with a User Experience Discipline: a Case Study

Hans Westerheim, Geir Kjetil Hanssen

SINTEF ICT, Trondheim

[hans.westerheim@sintef.no](mailto:hans.westerheim@sintef.no), [geir.k.hanssen@sintef.no](mailto:geir.k.hanssen@sintef.no)

**Abstract.** The Rational Unified process is widely used as a process framework for software development. The introduction and use of the RUP is not straight forward. Experience and research have shown that some sort of tailoring of RUP to the software development organization and the software development projects is necessary to be able to use the framework in a productive way in the projects. In this paper we describe software development in an organization with high focus on the user experience aspects and user interface design in their projects. This organization did tailor RUP to its types of projects by adding a user experience discipline. An evaluation of two development projects using this tailored RUP was done by the means of Post Mortem analyses conducted by external researchers. The main conclusion is that RUP needs to be tailored to be able to support the design of user experience oriented issues, and that the RUP structure is suited for doing so.

## 1. Introduction

The Rational Unified Process, RUP, is a comprehensive process framework for software development projects. RUP defines a software development project as a set of disciplines, e.g. requirements handling, implementation etc., running from start to end through a set of project phases. A project is performed by a group of actors, each having one or more well defined roles. Each role participates in one or more activities producing one or more artefacts. A discipline can run in iterations, that is, repetitions within a phase. Activities, roles and artefacts are the basic process elements of RUP.

Given the complexity and richness of RUP it needs to be tailored to the context of use as stated by Jacobson, Booch and Rumbaugh [1] p. 416: *It has to be tailored to a number of variables: the size of the system in work, the domain in which that system is to function, the complexity of the system and the experience, skill or process level of the project organization and its people.*" In this paper we report some experiences from a software company that has adopted a tailored version of RUP that particularly support development of graphically rich and user friendly solutions. RUP – out of the box – is by the most designed to support development of solutions with rich functionality. However, developing appealing solutions with an advanced look and feel cover additional challenges than purely technical solutions. How should requirements be captured and documented? How should the design be expressed and communicated? How to test? The software company being studied in this case did specialize on this type of solutions and our study covers two projects taking the new discipline named User Experience Discipline into use for the first time. The study is based on PostMortem analyses [2] of the projects that covers both positive and negative experiences. Our findings indicate that the new specialized discipline did help the company to produce good solutions but that it is a challenge to coordinate and balance the technical and design-oriented parts of a development project.

## 2. The tailored RUP – tRUP

The Rational Unified Process is founded on three core principles:

1. Use-case driven; requirements are documented in the form of Use-cases (as defined by UML [3]). A Use-case diagram is a graphical representation explaining users performing actions. A user may also be an integrated system. Use-cases are suited for documenting functional requirements.

## Session 3: SPI and Requirements Management

2. It is architecture-centric. As Use-cases define the function of the system, the architecture defines the form. Functionality and form (the system architecture) must be balanced and mutual supporting.
3. It is iterative and incremental. Being iterative means that the disciplines (design, implementation etc.) are performed as a series of iterations, each involving all or most disciplines. Incremental means that the solution is built in piece-by-piece where each piece adds to the previous.

Tailoring RUP may involve the inclusion and exclusion of low-level process elements such as roles, activities and artefacts but will always preserve the three core principles. In our case the tailored RUP (named *t*RUP) had replaced the entire Requirements-discipline with the User Experience Discipline which includes a new set of roles, activities and artefacts designated for design and development of user experience features of a solution. The new discipline is shown in Figure 1, and summarized in Table 1.

As RUP is tightly connected to the Unified Modelling Language, UML one implication is that RUP is weak at handling the non-functional aspects of the information system to be developed. The new discipline addresses this weakness by offering described artefacts suited for describing user experience issues.

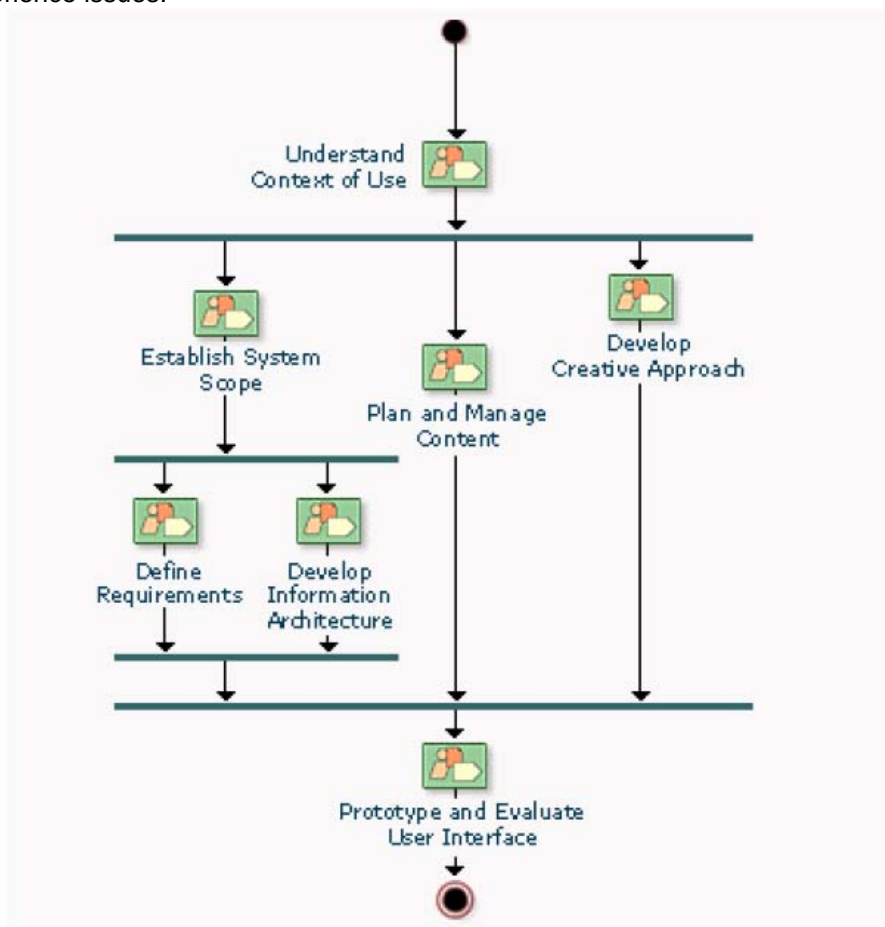


Fig. 1. The User Experience Discipline

## Content of the new discipline

The table shows the main content of the new user experience discipline:

Activities	Artefacts	Roles
<ul style="list-style-type: none"> <li>- Establish System Scope</li> <li>- Specify System Interactions &amp; Presentations</li> <li>- Plan &amp; Manage Content</li> <li>- Develop Creative Approach</li> <li>- Prototype and Evaluate User Interface</li> </ul>	<ul style="list-style-type: none"> <li>- User Research Report</li> <li>- Actor Catalogue</li> <li>- Use Case Model</li> <li>- Software Architecture Document</li> <li>- Creative Concept</li> <li>- Content Inventory</li> <li>- Content Development Guide</li> <li>- Visual Development Guide</li> <li>- Site Map</li> <li>- Wire frames</li> <li>- Usability Test Report</li> </ul>	<ul style="list-style-type: none"> <li>- Requirements Analysts</li> <li>- Usability Evaluator</li> <li>- Subject Matter Expert</li> <li>- Business Strategist</li> <li>- Creative Concept Director</li> <li>- Content Analyst</li> <li>- Brand Strategist</li> </ul>

**Table 1.** The content of User Experience Discipline in *tRUP*

## The activities explained

**Understand context of use** focuses on the characteristics of the users, the tasks to be performed by the users and the organizational and physical environment of the system. The context is formed by these elements.

**Establish System Scope** uses the Use Case model and refines it. The activity is to be done in close cooperation with the users and other stakeholders of the system. Interaction and integration with other systems are also documented. The baseline for the system scope is usually created in this activity.

**Specify System Interactions & Presentations** focuses on the interactions which the system will have with the users as well with other systems. The activity will also deal with the presentation aspects of the system to be developed.

A main task in this activity is to develop the information architecture. Both the groupings of content, data and functionality will be described in this architecture.

**Plan and manage content** focuses on the content of the web site. This includes the evaluation of strategies for the business and the brand. The visual aspects of the content are closely related to the organization of the content, and are an issue in this activity.

**Develop Creative Approach** serves as the kick-off when it comes to work with the creative development of the system. The conceptual and aesthetic aspects of the system are created during this activity. The development of the creative hypotheses is an important task in this activity.

**Prototype and Evaluate User Interface** supports development of the user interface prototype, based on the requirements, and evaluates this prototype. This is a very iterative process, and must be accomplished in close relationship with the end users of the system.

## *tRUP* in this company

At the time of this research *tRUP* was documented by the means of electronic documents. The documents were available through a rather simple web interface. There was no electronic process guide in place.

The introduction of tRUP in this company was mainly motivated by this new discipline since the company had its main focus on web development, with the emphasis on the user and the user's experience of the web sites. The original disciplines in RUP were offered to the development projects, mainly by the project manager, but the use of these disciplines was not mandatory. The project managers and the software developers were not trained in tRUP.

### 3. Method

During the research on the tailoring and use of RUP in organizations, the authors have been identifying and evaluated approximately 100 research papers presented either in journals or in conference proceedings. The papers evaluated were either indexed by Compendex<sup>1</sup>, ISI Web of Knowledge<sup>2</sup> or ACM<sup>3</sup>. Many of the papers are proposing a tailoring of RUP and how to implement and use the tailored RUP. However, most of them lack studies based on empirical data from the actual tailoring and use. The Norwegian tradition in software process improvement research is to collect and analyse empirical data taken from the companies developing software. This is also the case in this study. The researchers have performed the research in conformity with given guidelines for empirical research [4, 5].

This research presented in this paper is planned and accomplished as a case study. Case study research is described by Yin [6].

#### Study context

The company in this case study was a Norwegian branch of an international company developing web based information systems with a special emphasis on interactive and user-friendly solutions. The projects were organized as typical software development projects managed by a project manager.

By the time of this study, the Norwegian branch had approximately 70 employees. Most of the projects were small, both in calendar time and in man hours, lasting for some weeks. Projects with duration over 4-6 months were considered as "large projects". The customers were both commercial companies and public organizations. The company did develop both systems mainly serving information, as well as integrated systems with ecommerce functionality built in using integrated third-party solutions.

Traditionally in the company, the software developers and the user interface and user experience developers were working in parallel, not really cooperating during the work. The project managers were mainly recruited from the software development part of the company. In some cases this resulted in a low mutual understanding between the two groups; the technical developers and the user experience developers. To improve this situation and establish a unified process tRUP was adopted.

This study draws upon observations and analysis of two projects following the new RUP-based process; Project F developed a White/Yellow-pages solution for a national phone-indexing service. The delivered solution was based on a third-party solution for indexing and searching and a rich and user-friendly front-end. Project J did a redesign of an existing web-site keeping most of the technical functionality.

#### Data collection

Post Mortem analysis has been used as the main tool for collecting data in this case study [7, 8]. The main use of PostMortem in the two case-projects was to document the positive and negative experiences with the newly introduced tRUP-process in general and with the added User Experience Discipline particularly. A PostMortem analysis is a group oriented technique, where brainstorming and the use of sticky notes are important. Each participant is asked to write their positive and negative experiences. The notes are then grouped and summarized [9] as a collective activity resulting in a set of experience-groups with defined names. The results are documented in the form of a PMA-report that shows experiences at a group level. During the PMA-sessions, the researchers took personal

---

<sup>1</sup> [www.engineeringvillage2.org](http://www.engineeringvillage2.org)

<sup>2</sup> [www.isiknowledge.com](http://www.isiknowledge.com)

<sup>3</sup> [www.acm.org](http://www.acm.org)

notes during the work sessions. These notes are also treated as data since they give valuable information about the Post Mortem analyses.

### Data analysis

The data from the Post Mortem analyses has been structured and compared in a process similar to the principles of Constant Comparison [5]. The analysis is applied both at the written PMA reports as well as the notes taken by the researchers. The PMA reports in this case study follow a template which ensure that both positive and negative experiences from the project are documented [7]. The researchers used mind maps to take notes during the PMA work shops<sup>4</sup>. Findings from each case are compared and statements and experiences that resemble are grouped.

## 4. Results

When analyzing the data from the two Post Mortem analyses we find three main results regarding the design-team internal coordination of work, coordination with the technical side of the project and documentation of design-oriented requirements.

### Design-team internal coordination

Project members report that the use of the new User Experience Discipline improves the coordination of design-related roles such as the usability specialist, the usability evaluator and the producer which act as a mentor across several projects. The User Experience discipline defines these, and other roles, and their designated activities and results.

As this type of work had been a part of the business for a long time, there was established a set of best practices, including roles. Through the use of the User Experience discipline the project members saw that it resembled with these established best practices, however they became more formalized and not at least visible to other parts of the development organization. Those working with user-experience type of tasks appreciated this visibility as it made their work more recognized amongst the others that focused more on the technical sides of the projects.

### Cross team organization

When it comes to the actual outcome of this new discipline the results from the Post Mortem analyses clearly indicate that the discipline worked as intended; the defined artefacts was produced and contributed to the progress in the project. However, as it seems to have worked as a process guide for the user-experience oriented project members it seems quite clear that the design work was separated from the rest of the project, including the project manager. This division was intentional; however the drawbacks became pretty obvious when this was put into practice. There was a low level of coordination of the technical (traditionally functional) requirements and the user experience requirements. Accordingly this also affected the testing negatively. This lack of coordination is the most prominent learning from the two projects.

### Expressing design-oriented requirements

One of the core features of RUP is the strong focus on use cases as a mean of documenting requirements. However, in this case the feedback from the two projects clearly tells that use cases are not suitable for documenting graphical- and design –related issues. Therefore the projects used forms such as mock-ups, pure textual descriptions and design sketches. This is an important contribution from the new User Experience discipline which defines this type of artefacts. All these three artefacts

---

<sup>4</sup> [www.mindjet.com](http://www.mindjet.com)

are used as input to the activity Prepare User Interface Design. The participants at the Post Mortem analyses reported that this type of artefacts worked as an important medium for communicating with the customer. Further on they also emphasized the importance of involving the customers during the course of the project – as the solution grew. Besides this, such artefacts were also reported to positively affect the communication with the technical side of the project.

### 5. Discussion

Good communication is a basic prerequisite for succeeding in all types of software engineering processes, regardless of process flavour. In agile processes good communication is ensured by informal, tight, direct and frequent exchange of experience and opinions. In RUP the communication is more formally based on defined roles developing defined artefacts which are used as input by other roles. This means that the defined artefacts must be suitable for bringing information from one role to the next clearly and consistently. As RUP tends to focus on the development on technical functionality rich solutions most artefacts reflect this view. In this case we see that the new User Experience discipline which includes new types of (communication) artefacts such as User Research Report and Creative Concept clearly improved the communication at two levels. Firstly the internal communication within the design-team was reported to be improved, secondly these new design-oriented artefacts turned out to be working efficiently as a communication aid towards the customers and especially end-users.

While a standard Use Case diagram explains the functionality and order of actions in a system the new artefacts explained the look and feel of the system. This was not new to the design oriented developers; however the new process represented a more formalized view on their roles and function in the total project. As the PostMortem analyses showed us, this new discipline complied with existing best practice. In a comparison of five technology acceptance models, Riemenschenider *et al* [10] explains Compatibility as one factor for technology (process) uptake and success. They define this factor as *“the degree to which an innovation is perceived as being consistent with the existing values, needs, and past experiences of potential adopters”*. It is likely to believe that the positive view and experiences by using the new discipline can be explained by this effect. The design-oriented developers expressed satisfaction as their function now became more visual to the rest of the projects, especially to the technical-oriented developers. Here we also find a explanatory technology acceptance factor; Subjective Norm [10] which is defined as *“the degree to which people think that others who are important to them think they should perform the behaviour”*. In our case the formalization of roles, responsibility and contribution to the total project most probably have contributed to an increased subjective norm.

As the new User Experience discipline are reported to have improved internal design-team and customer/end-user communication it did not affect the communication with the technical side of the project to that extent that it could have done. The two parts of the projects traditionally worked pretty much separated, a practice which was continued with the new discipline. This was the most prominent negative experience and correspondingly improvement issue from the PostMortem analyses. The developers clearly wanted the two teams to cooperate more in practice. Our two case-projects can be seen as pilots for the new experience, thus we believe that this is only a result of an easy first try and that new projects should integrate the technical and design-oriented sides to a much larger extent. As the new design artefacts was suitable for documenting user experience requirements and design the new discipline did not contain any guidelines for testing. As standard by RUP, testing is separated into the Test-discipline which did not encompass the new User Experience Discipline. We believe that there is a great potential for process improvement by extending the Test-discipline to involve end-users frequently to verify the design aspects of the solution. Agile processes [11] holds many practical guidelines for this that easily could be incorporated.

### 6. Conclusion

In total we see that for a development organization that emphasize the user experience, RUP should be extended to support new roles, artefacts and activities. However it is important to strongly integrate both the technical/functional parts of the work with the design-oriented parts. Given the situation, the

development of a great look and feel may be at least as important as the technical and functional sides of the end product. The process should clearly reflect this.

This case study clearly shows that it is possible to create a discipline for user experience development using a traditional RUP structure.

### 7. Further research

The authors have been studying the tailoring and implementation of RUP in several case studies [12-15]. During the work it has become clear that there are few empirical studies covering such tailoring and implementation reported [16, 17]. Most of the papers dealing with RUP are proposing a tailored RUP, but with no empirical data to confirm the usefulness of the tailoring. There are also a considerable numbers of "white papers" covering RUP, but the scientific quality of these is impossible to measure.

IBM Rational provides a comprehensive set of tools supporting RUP<sup>5</sup>. The new version of the tools provides functionality for selecting roles, artefacts and activities. The selection may be connected to types of projects.

One interesting issue would be to conduct empirical studies on companies implementing RUP using the new set of tools.

### 8. Acknowledgement

The researchers want to acknowledge the Norwegian Research Council for its support to the software process improvement projects PROFIT and SPIKE under which the collection and analysis of data in this case study have taken place. The researchers will also thank the company in this case study for giving us access to their projects and employees.

### 9. References

1. Jacobson, I., G. Booch, and J. Rumbaugh, *The Unified Software Development Process*. Object Technology Series, ed. G. Booch, I. Jacobson, and J. Rumbaugh. 1999, Reading, Massachusetts: Addison Wesley Longman Inc. 463.
2. Birk, A., T. Dingsøy, and T. Stålhane, *Postmortem: Never Leave a Project without It*. IEEE Software, 2002. **19**(3): p. 43 - 45.
3. Krutchen, P., *The Rational Unified Process: An Introduction*. 2nd ed. 2000: Addison-Wesley. 298.
4. Kithenham, B., et al., *Preliminary Guidelines for Empirical Research in Software Engineering*. IEEE Transactions on Software Engineering, 2002. **28**(8): p. 721-734.
5. Seaman, C., *Qualitative methods in empirical studies in software engineering*. IEEE Transactions on Software Engineering, 1999. **25**(4): p. 557-572.
6. Yin, R. and D.T. Campbell, *Case Study Research*. 2002: Sage Publications Inc.
7. Birk, A., T. Dingsøy, and T. Stålhane, *Postmortem: Never leave a project without it*. IEEE Software, special issue on knowledge management in software engineering, 2002. **19**(3): p. 43 - 45.
8. Collier, B., T. DeMarco, and P. Fearey, *A Defined Process For Project Postmortem Review*. IEEE Software, 1996. **13**(4): p. 65-72.
9. Stålhane, T., et al., *Post Mortem - An Assessment of Two Approaches*, in *Empirical Methods and Studies in Software Engineering: Experiences from ESERNET*, R. Conradi and A.I. Wang, Editors. 2003, Springer Verlag: Heidelberg. p. 129 - 141.
10. Riemenschneider, C.K., B.C. Hardgrave, and F.D. Davis, *Explaining Software Developer Acceptance of methodologies: a Comparison of Five Theoretical Models*. IEEE Transactions on Software Engineering, 2002. **28**(12): p. 1135 (10).
11. Cockburn, A., *Agile Software Development*. The Agile Software Development Series, ed. H.J. Cockburn A. 2002: Addison-Wesley.

---

<sup>5</sup> www.rational.com



## Session 3: SPI and Requirements Management

12. Westerheim, H., T. Dingsøy, and G.K. Hanssen. *Studying the User Experience Discipline extension of the Rational Unified Process and its effect on Usability. The design of a case study.* in *WSESE 2002*. 2002. Rovanjemi, Finland: Fraunhofer.
13. Hanssen, G.K., H. Westerheim, and F.O. Bjørnson. *Tailoring RUP to a defined project type: A case study.* in *6th International Conference on Product Focused Software Process Improvement, PROFES*. 2005. Oulo, Finland: Springer.
14. Hanssen, G.K., H. Westerheim, and F.O. Bjørnson. *Using Rational Unified Process in an SME - A Case Study.* in *12th European Conference, EuroSPI 2005*. 2005. Budapest, Hungary: Springer.
15. Westerheim, H. and G.K. Hanssen. *The Introduction and Use of a Tailored Unified Process A Case Study.* in *Software Engineering and Advanced Applications, 2005. 31st EUROMICRO Conference on*. 2005: IEEE.
16. Bygstad, B. *Controlling iterative software development projects: The challenges of stakeholder and technical integration.* in *The Hawaii International Conference on System Science*. 2004. Big Island, HI, United States: Institute of Electrical and Electronics Engineers Computer Society.
17. Folkestad, H., E. Pilskog, and B. Tessem. *Effects of Software Process in Organization Development - A Case Study.* in *Advanced in Learning Software Organizations: 6th International Workshop, LSO 2004*. 2004. Banff, Canada: Springer.

# Improving Requirement Reuse: Case Abloy

*Jukka Heinänen & Markku Tukiainen*

## **Abstract**

Managing the concurrent development of several integrated systems and software products is a difficult task. At the same time there is a need to specify and maintain all relevant documents, both by the sales and marketing persons and developers in order to produce marketing, technical and training material. There is also continuous need to make changes and develop new product versions for the customers. In this paper we discuss how to improve reuse of existing documentation and what needs to be done to requirement management process and documentation practices in the industry settings to enhance all uses of product documentation.

## **Keywords**

Reuse, SPI, Requirement Management

### 1 Introduction

In the industrial systems software development there is still a strong call for developing practical tools to manage requirements during the lifecycle of the product. From the reuse benefit point of view it is important to use the existing requirements documents as a starting point for further development. We have realized in the practical industrial settings that quite often circumstances are not so promising to reuse existing documents. In almost every case there is a need to first update the documents and then carry on to the next phase of development of the system. This causes the developers to waste time and money. While developing new systems, it is evaluated that 50 to 80% of the total costs will be spent during the maintenance phase after the first version development [1]. Reuse of existing artefacts could reduce the needed investments and could speed up the development process.

Abloy Ltd is one of the leading manufacturers of locks, locking systems, architectural hardware, and is the world's leading developer of products in the field of electromechanical locking technology. Abloy is part of the ASSA ABLOY Group, which is the world's leading manufacturer and supplier of locking solutions, dedicated to satisfying end-user needs for security, safety and convenience.

Abloy is developing order handling solutions, locking systems specification solution for locksmiths and importers and locking systems management systems for the customers. Software product development is based on product family principles. We have realized a need to improve the development process and increase reuse of existing information and documents.

This paper presents the improvement actions and results in the following way:

Section 2 defines the aspects of reuse and reuse approaches

Section 3 describes the improvement actions in the example company

Section 4 shows the possible ways to improve the software process

Section 5 makes a summary of the current findings.

### 2 Requirements reuse

Our work has focused on the first phases of requirement engineering, mainly to the requirement definition and system design. We have been interested in evaluating all the product documentation in order to find out how easily these documents could be reused.

A term reuse means that something is possible to use again in another time to produce something. A reuse approach is an approach in software engineering to promote a reuse as a major factor. Ezran [2] describe software reuse as "Is the systematic practice of developing software from a stock of building blocks, so that similarities in requirements and/or architecture between applications can be exploited to achieve substantial benefits in productivity, quality and business performance [2].

Somerville and Sawyer [3] separate requirement reuse in two concepts; *Direct requirement reuse* – "Means that a requirement from one system is taken as a requirement for some other system with minimal change", and *Indirect requirement reuse* – "means using existing requirements in the elicitation process to prompt users for their specific requirements"[3].

So the main requirement for reusing existing requirements are that the process for reuse is explicit and all elements are systematically prepared and designed to be reused.

### 2.1 Improving requirements reuse

In the industry setting the reuse of existing requirements could mean, that the company has documented all processes and all product and software development requirements and related documents are up to date and managed in a repository using a requirement management tool. Support from this kind of tool is crucial, because it forms a reuse library. There are approaches, which have concentrated on requirement reuse or approaches, where requirement reuse is a part of the whole approach. These include the following:

- Software product families
- Domain and feature modelling
- Patterns
- Specification reuse
- Use cases and scenarios
- Viewpoints
- Component based software engineering
- Component off the shelf
- Aspect oriented development and
- Software product lines.

In the recent study of the software engineer work, it is argued, that engineers update documentation quite seldom, so those documents are useful for the developer only in the limited scale for developing new features. These findings are interesting, because the role of the documentation plays a very important role on communicating the requirements from the users up to the developers of the project [9]. Software reuse should be integrated into the usual development process, then it becomes a lens through which engineers view the problem [10].

At Abloy the main focus is how to reuse the existing software requirement specifications (SRS) and other related documents when developing systems. Looking at the collection of different approaches, it is not easy to evaluate the best possible approach for the company and even more difficult to change the current working methods to support a reuse based processes and view the problem as a reuse problem. The next example shows what Abloy has done when improving its requirement management process and evaluate the possibility to reuse existing documents and artefacts.

## 3 Case Abloy

Abloy has acted as a test company for adopting BaRe, a basic systematic requirement engineering practices with an easy to adopt method for small and middle size companies [6]. This participation and improvement work produced a basic documentation structure and the documentation templates. After this development phase, new projects have used the templates. During these projects, there arised several new needs for reusing these documents. The development teams calculated that during a software product development project there was several new documents, which needed the same information that was already available in those central requirement management documents. These new documents are mainly for launching the product to the market, training customers using the product and informing customers for new features and functions. All these documents form a document matrix where we have also identified the basic views for these documents[11].

### 3.1 Requirements documentation

Therefore, there was a need to specify the overall information structure for the software product. In the

figure 1 there is an example of the information structure. This figure describes, that there are several documents needed, before the whole product is ready to launch to the market.

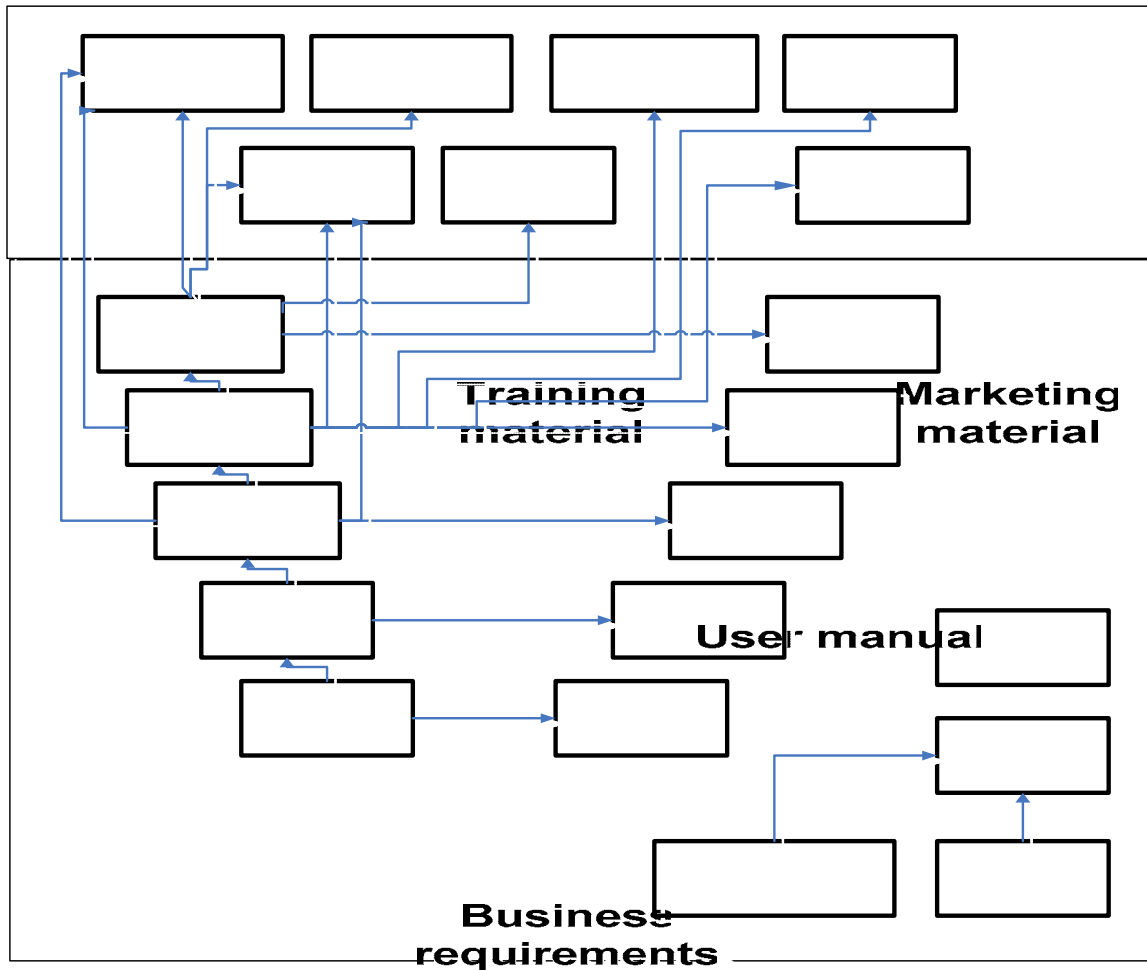


Figure1: Information structure of the product.

The Software requirement specification (SRS) describes the system to be developed. At Abloy this includes Business requirements, Software requirements, User requirements, Use cases and Technical specification documents. Other documents use the same information from the SRS documents. Most of these documents are stored in a requirement management tool. This tool can manage software requirement specifications and it forms a reuse library [4]. Without this kind of requirement management tool, it is quite difficult to improve the capability of requirement management and maintain the documentation.

It's relevant to notice that other documentation of the company have relation to software products, for example process documentations, user manuals, training material, marketing material, business plans, customer needs and business opportunities. Developers need these documents to analyse how they influence other stakeholders and customers. If this information is not available, these influences can be wrong, without a proper evaluation or missing.

Other area with a need to document requirements is in the operating of the system. There is a need to describe and document installation, maintenance, operational guide, environment document and help desk guide. All these documents have strong links to actual software requirement specification, and it is not possible to launch a new product without these operating documents.

Our experience shows, that the software requirement specification in Abloy's case is not enough to specify all needed requirements and to document all aspects of the product. Main source for other

documents is SRS, but these are documents, that are only loosely connected to requirements found in SRS.

When analysing these documents for reuse purposes, these loosely coupled documents are easy to reuse, because their structure and content can be reused both vertically (for the next version of the product) and horizontally (for the new product in the new application area).

### 3.2 Improved documentation practices

At Abloy the development focus has been how to reuse the existing software requirement specifications (SRS) and other documentation. Table 1 shows a collection of improved approaches and situations for requirements reuse in practice. The terminology used in the table is:

- Vertical reuse - Reuse which exploits functional similarities in a single application domain, [2]
- Horizontal reuse - which exploits similarities across two or more application domains. [2].
- Direct requirement reuse - Means that a requirement from one system is taken as a requirement for some other system with minimal change, [3]
- Indirect requirement reuse - means using existing requirements in the elicitation process to prompt users for their specific requirements, [3].

Current system requirements	Development step	Application domain	Reused elements	Reuse type
Current SRS and system	Next version of the system, Updated SRS	Single application domain	Requirements	Indirect requirement reuse
Current SRS	Same system, but to other organization, updated SRS	Single application domain	Requirements and/or architecture	Vertical reuse
Current SRS	Replace existing system with a COTS-package, SRS are requirements for the new system and acts as a check list for the project.	Single application domain	Requirements	Indirect requirement reuse, It is not necessary that all functions are the same in the new system.
Current SRS	Develop same system but with a new technology. Requirements are first updated and then validated and accepted. Need to validate the architecture also.	Single application domain	Requirements and partly architecture, eg. interfaces	Indirect requirement reuse, vertical reuse
Current SRS	When developing same kind of systems for several application domains	Several application domains	Requirements and architecture	Horizontal reuse

Table 1: Improved types of requirements reuse.

The reuse could be divided into following five types:

1. Current system requirements and developing the next version of the system

This case is very common at Abloy and is relevant in every case when developing a new version of the system. The most critical issue is that all documentation should be up to date in order to minimize the efforts evaluating existing requirements and to develop new features for next release. Development group uses existing requirements as a starting point to describe needed modifications.

### 2. Current system requirements and delivering the system to other organization

This case is relevant when a system is delivered into an other organization ( a division or group), which have similar type of business structure and customers, but different products or marketing structures. A typical case is first sell software product to domestic markets and second to sell the same product to export markets. Products are the same but many other things differ (selling process, order handling process, training, marketing structure, customer supporting requirements, customer requirements). Development group uses existing requirements to specify needed modifications and tries to find out as many functional similarities as possible.

### 3. Current system requirements, replacing system with COTS – package

This case is applicable when a tailored system is replaced with a new package software ( COTS-package). If a company have a current documentation of the system, it is possible to compare functionality between the old and the new system.

### 4. Current system requirements and using new technology in the development

This case is relevant when developing a new system, which replaces an existing one and there are only technological reasons to develop new system. The most important issue is that all requirements are up to date and developers could easily find out needed functions and other requirements

### 5. Current system requirements and developing same kind of system for other application domains.

This case is applicable when development team develops core technology platform and other development teams are using this technology for developing different products (partly same functionality, different customer needs and to different markets). This means that the requirements for the core technology need to be updated and available for all other development teams.

## 3.3 Requirements as corporate asset

After the first version of the software product has been developed, the documentation of the system remains steady and is not updated, although there have been changes to the system or some parts of the system. Many of those documents like business rules, marketing material, user manual, customer training material, product specifications for the customers and customer needs, will be relevant for the company after the development project. However, organizations do not always realise, that these requirements have more value for the company than the exact documentation of the development project [8].

If a company gather functional requirement, which defines enterprise and business units processes, they provide an advantages compared to their competitors. Gathering these requirements from all major systems, they could have a thorough knowledge and documentation of their systems, and that will be a very important information base for the future, that is an investment in the intellectual capital as Salit defines it [8].

At Abloy we are heading to maintain documentation updated, even though it is a tough task for change management. In all cases, we have not succeeded to do as well as we have planned.

If the system documentation is not updated, there will be obstacles for maintaining the system, for example, there are no proper instructions for computer failure, network problems or backup recovery. So it is vital that documentation is kept “alive” and up-to-date during the development phases and during the whole product life cycle and that the documentation is consistent with the product itself ([7], [8]). This is the major requirement for reuse and it is an obstacle for reuse if the documentation is deficient or inconsistent.

#### 4 Improving requirements management and reuse

Reuse has a strong technology dimension, in the sense that it needs major changes in the business area, changes in the development and maintenance methods and in the technology, which supports all these drivers. Our experience shows, that continuous improvement through the capability levels gives structure for the reuse improvement work. We have followed the guidelines depicted in table 2.

Guideline	Level	Description
Define a standard document structure	Initial	Define a standard for requirements documents which includes all relevant system information.
Uniquely identify each requirement	Initial	Make sure that each requirement has an unique identifier and that this identifier is always used to refer to that requirement.
Define policies for requirement management	Initial	Have an explicitly defined change management processes and put procedures in place to assure that this process is followed.
Use checklists for requirement management	Initial	Develop checklists of potential problems and explicitly check off these problems during the requirements validation process.
Use scenarios to elicit requirements	Repeatable	Develop a set of usage scenarios and use these with stakeholders to elicit their specific requirements.
Specify requirements quantitatively	Repeatable	Where it is appropriate, specify requirements such as reliability requirements in terms of system attributes which can be objectively measured.
Use prototyping to animate requirements	Repeatable	Develop an executable prototype of a system based on outline requirements and get stakeholders opinions of the prototype facilities.
Reuse requirements	Defined	Whenever possible, reuse requirements from previous system as these will ( to some extend) have already been validated
Specify system using formal requirements	Defined	If you are involved in the development of a critical systems, use a mathematical specification language to specify the functionality of the system.

**Table 2: Reuse in maturity levels [3]**

In every maturity levels, there are guidelines to define and utilize tasks and practices to reuse requirements. When a company's process is in the level 3, defined process, then there is clear description for reuse requirements. Therefore, a company must incrementally improve their process when targeting to fully utilize requirements reuse. But still it is possible to reuse documentation at the lower levels of the maturity.

For requirement management we have used these guidelines in initial and repeatable levels and now we are developing our process for stabilizing reuse practices and reuse requirements effectively.

We do not have specified and not collected any measurement data for quantifying documentation reuse level or data for analysing how accurate are our documents in each products documentation structure.

One simple way to find it out is that in each case when someone use this documents, does he/she need to update this document before he could specify those new required changes. If he need to update it, then the previous task have been deficient.

#### 5 Conclusions



In this work we have described how company Abloy reuses existing software product documentation and what improvements have been done in the current processes and documentation practices. The main findings could be summarised as follows:

- There is a possibility for business benefits.
- It will reduce re-work when documentation is up to date.
- There is a need to specify an information structure for the whole documentation
- Even in the lower levels of maturity there are possibilities to reuse artefacts.
- Documentation is a corporate asset and is very valuable for the company.
- Organization should use a common requirement management tool for managing requirements and to create a reuse repository.
- The accuracy of all documents are very important and it should be a key task for developers to keep them up to date.

Our main conclusion is that it is a challenge to make a change for reusing the existing documentation, but there are benefits even in the short time span. The reuse should be planned carefully and in the early stages of development when implementing new processes and working methods in the company.

### 6 Literature

- [1] Hall, T. (1992), *Software Reuse and Reverse Engineering in Practice*, edited by P.A.V. Hall, UNICOM, Chapman & Hall, London, 1992.
- [2] Ezran, M., Morisio, M., Tully, C., (2002), *Practical Software Reuse*, Springer-Verlag London.
- [3] Sommerville, I. and P. Sawyer (1997). *Requirements Engineering: A Good Practice Guide*, John Wiley & Sons.
- [4] Volere, 2005, Volere document templates ( [www.volere.co.uk](http://www.volere.co.uk), reviewed 24.3.2006)
- [5] Cybulski, J.L., K. Reed: (2000), *Requirements classification and reuse: Crossing Domain Boundaries*, Sixth International Conference on Software Reuse, Vienna, Austria, 2000.
- [6] Nikula, U, (2004), *Introducing basic systematic requirements engineering practices in small organizations with an easy to adopt method*, 2004, Lappeenranta University of Technology, Finland.
- [7] Parnas, D. L, Madey J, (1995), *Functional documents for computer systems*, *Science of Computer Programming*, 25, 1995, 41-61.
- [8] Salit, R., (2003), *Requirements Are Corporate Assets*, *IEEE Software*, May/June, 2003
- [9] Lethbridge, T., Singer, J., Forward, A., (2003), *How Software Engineers Use Documentation: The State of the Practice*, *IEEE Software*, November/December, 2003, pages 35-39.
- [10] Predonzani, P, Succi, G, Vernazza, T, (2000), *Strategic Software production with domain-oriented reuse*, Arctect House, inc.
- [11] Paech, B, Dörr, J, Koehler, M, 2005, *Improving Requirements Engineering Communication in Multiproject Environment*, *IEEE Software*, January/February, 2005, pages 40-47.

### 7 Author CVs

#### **Jukka Heinänen, M.Sc.**

Abloy Oy  
PL 108, FIN-80101 Joensuu, Finland  
<http://www.abloy.fi>  
[jukka.heinanen@abloy.com](mailto:jukka.heinanen@abloy.com)

Jukka Heinänen is an IT Manager at Abloy Oy, Joensuu, where he leads the IT department and is responsible for IT systems and software product development. He has over 20 years of experience of working with the IT systems and software development. His research interest include requirement reuse, requirement management and software process improvement. He received his MSc in computer science from the Oulu University.

#### **Markku Tukiainen, Ph.D.**

Department of Computer Science  
University of Joensuu, P.O. Box, FIN-80101 Joensuu, Finland  
<http://www.cs.joensuu.fi/~mtuki>  
[Markku.Tukiainen@cs.joensuu.fi](mailto:Markku.Tukiainen@cs.joensuu.fi)

Markku Tukiainen is a professor of Software Engineering in the Department of Computer Science at the University of Joensuu, Finland. His main research interests in software engineering are in Software Process Improvement in the small and medium size enterprises and the development of Certification of Software Engineering Professionals. Research of SPI in SME's involves a very close connection with software companies and this has been established through a SPIN network called tSoft in the city of Joensuu area ([www.cs.joensuu.fi/tSoft](http://www.cs.joensuu.fi/tSoft)). In Certification of Software Engineering Professionals the objective is achieving and promoting a globally accepted benchmark for the certification of software engineering professionals through an ISO standard development (ISO/IEC JTC1/SC7/WG20).

# Exploring National Cultural in Software Development Practices

*Aileen Cater-Steel and Mark Toleman*

## **Abstract**

The software development industry has become globalised. Trends which have contributed to globalisation include the maturation of software industries in developing countries, collaborative teams covering extended geographic areas, and migration of computing professionals. This paper analyses the Australian computer professional workforce and determines that 40 percent of computing professionals were not born in Australia. Results from surveys about adoption of software development best practice conducted in 16 countries are then summarised and analysed using Hofstede's cultural dimensions. The discussion considers the efficacy of the concept 'national culture' in light of the analysis and concludes that information systems researchers need to reconsider what national culture is, and how it can best be measured.

## **Keywords**

National Culture, Globalisation, software development best practice

### 1 Introduction

Software development has become a global activity and it is recognised that the business environment and culture varies from one location to another (Shore and Venkatachalam 1995). Dramatic improvements in software development tools and methods have also allowed geographically and culturally diverse developers to collaborate in global software development teams (Karolak 1998). Added to this, recent migration trends have resulted in a multicultural information communication and technology (ICT) workforce in Australia, and in other countries such as the USA, Great Britain and Ireland. The concept of *national culture* as defined by Hofstede refers to the collective mental programming shared by people which distinguishes the members of one nation from that of other nations (1980). To date, there has been limited research into the role of national culture in software development, and doubts raised about whether national culture, as defined by Hofstede, ever actually existed (Myers and Tan 2002). For example, cultural factors were identified by Paulish and Carleton (1994) as evidenced by differences in adoption of software process methods in Siemens sites in Germany and USA. In commenting on the fact that the capability maturity model integration (CMMI) and ISO 15504 have two dimensions, the process dimension and capability dimension, Biro, Messnarz and Davison (2002) call for a third dimension to CMMI and ISO 15504—the cultural dimension—because ‘the national cultural position of the company may determine a different meaning and suitable improvement actions’ (p. 36).

This paper explores the relevance of national culture solely in relation to the software development team. Other SE research has considered the role of national culture in relation to developing systems for a culturally diverse range of users, and the deployment, use and management of international information systems. Essentially, the study attempts to validate Hofstede’s national culture dimensions for the case of software developers.

In the next section (§2), the emergence of a multicultural software development industry is discussed and the Australian computing professional workforce is analysed to determine the extent to which recent immigration trends have impacted. In §3, the results from a software development best practice survey carried out by the European Software Institute (ESI), and replicated in Queensland are used to highlight variations in the adoption of software practices across 16 countries. Following the approach of Biro, Messnarz and Davison (2002), Hofstede’s (1980) five generic factors which characterise value systems in different national culture dimensions are described in §4 and used to explore the relationship between the ESI survey results and the cultural dimensions. The discussion (in §5) focuses on the outcomes of the immigration analysis and survey analysis, in particular highlighting limitations in the concept of national culture and its shortcomings in explaining issues in software development. The conclusion (§6) suggests directions for future research.

### 2 Multinational Software Development

Three recent trends have necessitated the consideration of the effect of national culture: the globalisation of the software industry; geographically dispersed collaborative software development teams; and migration of software developers.

#### 2.1 Globalisation of Software Industry

Over the last decade, global development efforts have become the industry norm rather than the exception (MacGregor et al. 2005). Previously, systems were either developed locally, or software development was carried out in countries with relatively mature software industries. With the recent liberalisation of markets and economic progress in many developing nations, emerging countries such as India are increasing in software development capability, and gaining a greater share of the international market (Costlow 2003).

In order to maintain a role in the domestic and international market, software firms are under pressure to comply with recognised software process improvement programs such as the CMMI, which was developed by the Software Engineering Institute (SEI) at Carnegie Mellon University in Pittsburgh. CMMI has gained international acceptance in the software development community: of the 87 CMMI appraisals performed up to mid 2003, only 39 were carried out in the USA (Phillips 2003). Third party assessments of software development processes provide evidence to investors and customers of the firm's commitment to software quality (Saran 2001). Increasingly, large Australian software purchasers, such as the Defence Materiel Organisation (DMO), Telstra and ANZ Bank, are recognising CMMI benchmark results when selecting their suppliers (Howarth 2004).

Countries such as Australia and India are adopting standards such as the US-based CMMI in order to be competitive. Consequently, local development firms 'must struggle with systems development methods created in other countries' (Shore and Venkatachalam 1995, p.5). As well as taking into account the variation in national cultures, Krutchen (2004) believes 'everyone knows the difficulty of adapting technology and methods to other cultures' so the variation in adoption of specific software development practices across different countries should come as no surprise. The cost and complexity of the CMMI sets a high hurdle for domestic firms to overcome: 'the Standard CMMI Appraisal Method for Process Improvement (SCAMPI) audit process is rigorous and intense. Accenture, for one, reported spending 8045 hours—six months' solid work for a team of eight—implementing the CMMI Product Suite ... IBM's CMMI Level 5 certification required a review of 1100 developers in four cities, testing compliance with 425 distinct CMMI practices; its SCAMPI audit required five weeks' work from a team of nine auditors' (Braue 2004).

Large multinational corporations need to balance their systems to accommodate the local needs of host organisations, as well as the centralised needs of headquarters (Cheung and Burn 1994). It has been determined that critical success factors for IT vary, depending on geographic region, therefore, it is important for multinational corporations to understand the different sets of issues so that management can take appropriate actions to achieve success (Khandelwal and Ferguson 1999).

## 2.2 Collaborative Teams

It has been claimed that the use of widely dispersed collaborative teams is prompting a radical change in the way software is developed (Karolak 1998). Promoted as a just-in-time approach, organisations such as Bangalore-based Infosys provide low-cost, world-wide application developers and use Internet-based, open-source tools to create application development teams in collaboration with their clients' IT staff (McCarthy 2003). Such teams are able to respond quickly when new applications are required urgently, in contrast to in-house development projects which are often delayed due to the organisation's systems development backlog.

## 2.3 Australian Multicultural IT Industry

In response to a skills shortage, Australia's immigration policies were changed in 2001 to encourage overseas ICT professionals. This policy change was reversed in 2003 as it became apparent that the skills shortfall had been grossly overestimated and migrants were competing with unemployed locals for a shrinking number of IT positions (AAP 2003).

As a result of the approval of permanent and temporary visas, the Australian IT industry now has many computer professionals from a wide range of birthplaces. An analysis of the 2001 ABS Census revealed that 40 percent of computing professionals, 35 percent of IT Managers and 32 percent of Computing Support Technicians were not born in Australia. (ABS 2005)

The majority of permanent and temporary migrant computing professionals recorded their birthplace as Asia (41%) or Europe (35%). Considering the relative populations of areas, the proportion of migrants from New Zealand (6%) is very high, probably due to its close proximity, common language and similar culture to Australia. Of the remainder, the breakdown was Africa (6%), America (5%) and other (7%).

The proportion of migrant IT workers has increased since the 2001 census. In the two years from July 2003 to June 2005, 18,000 Computing Professionals were granted skill stream visas (Vanstone 2005), and in the last financial year, a further 3,379 temporary skilled visas were awarded to IT workers (Bajkowski 2005). Consequently, the multicultural diversity of software development teams in Australia has increased substantially.

Although organisations have been undertaking software development activities for many years, the lack of implementation of local or international standards and curriculum has resulted in a wide variation of practices in use. This variety in software development practice has implications for organisations purchasing software, off-shore outsourcing, and collaborative teams with diversity in terms of geographic location or multicultural members. The next section reports on the results of surveys used to determine the variation in adoption of best practice in software development.

### **3 ESI Best Practice Survey**

Although many authors refer to software developers using dominant, prevalent, or common practices, there has been little research to date to document actual current use. The most widely reported survey of best practice in Europe was that conducted by the ESI (Dutta et al. 1998b; ESI 1996). In 1995, the European Commission launched the European Systems and Software Initiative (ESSI) program with the aim of motivating organisations to test and deploy software best practices. The ESSI program was administered by the ESI as part of the European Commission's Information Technologies program (Dutta et al. 1999). Organisations were encouraged to apply for funding to enable them to adopt a specific software process improvement (SPI) project in a real-life commercial environment. The ESSI program included a longitudinal study of European software practices to assess and monitor the level to which European software developers were adopting best practices.

The ESI developed the Software Best Practices Questionnaire (SBPQ) to collect data for the ESSI program. Previous research in software process improvement and popular models such as the CMM, Bootstrap and ISO 15504 influenced the development of the questionnaire. On three occasions (between 1995 and 1997), the questionnaire was distributed by the ESI as part of the call for proposals for ESSI funding. Respondents were explicitly informed that the questionnaire was independent of the funding proposal review process (Dutta et al. 1999).

A best practice is defined as 'a management practice that is widely recognised as excellent and is recommended by most practitioners and experts in the field' (ESI 1997). The SBPQ represents the 'subjective consensual views of multiple experts' (Dutta and Van Wassenhove 1997), and comprises a subset of core software development practices including organisational issues, standards and processes, metrics, control of the development process, and tools and technology.

The content of the questionnaire has been criticised on two counts by Dutta and Van Wassenhove (1997): firstly, it overlooks important issues related to organisational and acquisition management; and, secondly, it does not include practices associated with high maturity organisations (for example, CMM level four and five practices).

However, despite its shortcomings, the ESI study yielded valuable findings from the analysis of the 1,279 responses received over three years. There were 463 responses to the first survey in March 1995 from 17 countries (ESI 1996). The second survey was conducted in mid 1996 and received 488 responses from 17 countries (ESI 1996). The third and final survey in 1997 generated 397 responses (ESI 1997) and showed 'wide variation in both awareness and application of process improvement techniques' (Dutta et al. 1999). On average, the respondents had adopted 51 percent of all practices. The difference in the average adoption varied markedly, as detailed in table 1: firms from the United Kingdom and France showed the highest overall adoption rates, while Sweden and Spain had adopted the lowest proportion of practices.

For specific practices, adoption varied greatly from one country to another. For example, while 83 percent of Norwegian firms establish a change control function for each project, only 20 percent of Belgian organisations have adopted this practice. Both Belgium and Denmark scored an average of 53 percent for maintaining awareness of CASE or other new software development technologies compared to only 8 percent of Irish firms. While Spain scored poorly on most practices, it was the

leader for controlling estimates, schedules and changes, and also obtaining signoff from all parties before changing plans (Dutta et al. 1999).

**Table 1. Best Practice average adoption for each country– overall and for each group of processes. Source: (ESI 1997).**

Country (N)	Org. Issues	Standards & Processes	Metrics	Control	Tools	Overall Average Adoption
Queensland	48	54	35	49	47	48%
Australia (205)						
Austria (16)	66	50	42	60	46	53%
Belgium (15)	52	41	40	46	40	43%
Denmark (17)	64	53	46	63	53	55%
Finland (4)	63	56	50	54	50	55%
France (18)	72	62	61	76	58	65%
Germany (62)	55	48	43	52	47	49%
Greece (18)	63	57	49	65	50	57%
Ireland (12)	51	43	36	51	45	45%
Israel (11)	57	47	38	55	34	46%
Italy (77)	57	52	50	61	40	52%
Netherlands (30)	57	49	41	51	48	49%
Norway (6)	60	53	44	61	48	53%
Spain (34)	53	44	36	57	35	44%
Sweden (13)	38	36	25	33	26	32%
U.K. (52)	66	63	52	67	50	60%

(Note ESI results are from 1997 survey. Countries with less than 4 responses were omitted.)

The ESI survey was replicated as a mail survey in Queensland in 1998. The intention was to conduct the survey initially in Queensland, then later in other Australian states. The survey returned 205 responses from organisations which develop software for sale or for internal use. As shown in table 1, the average adoption of 48 percent by Queensland firms is lower than that reported by 10 of the listed countries in the final ESI survey, but higher than Belgium, Ireland, Israel, Spain and Sweden. Queensland organisations showed strength in the use of standards and processes (54% adoption), but were very weak in the application of metrics (35% adoption).

As world class standards are dynamic, the set of practices considered to be the *best* changes over time, especially in software development which has frequently adapted to changes brought about by evolution of technology (Finkelstein and Kramer 2000). With the passing of time, best practice becomes standard practice as other superior practices emerge (Cragg 2002). Recently, agile software development methods have been promoted as best practice. Proponents of agile methods would compile a different set of best practice techniques, focussing on customer satisfaction and early incremental delivery of software; small highly motivated project teams; informal methods; minimal software development work products; overall development simplicity stressing delivery over analysis and design; and active and continuous communication between developers and customers (Pressman 2003).

The best practice questionnaire heavily emphasises project management, but has no practices relating to risk management, measurement, validation, joint review or audit. The questionnaire does not include Beta test management, an important process for firms developing packaged software. Jones (2003) notes that Beta testing has been used since the 1960s, and Cusumano et al. (2003) reported its widespread use at 73 percent. Therefore, it is recognised that the items from the ESI questionnaire may not provide an entirely valid measurement of best practice across the industry. This point is acknowledged by the ESI: 'progress in software engineering may not be visible along dimensions measured in the survey' (ESI 1998, p. 29).

Another issue to consider is that software practices may have changed significantly in the six years between the design of the ESI questionnaire and its use in the Queensland survey. For example,



reuse is now recognised as one of the most valuable software development practices (Mili et al. 1995) but is not included in the ESI questionnaire. So while longitudinal studies such as that undertaken by the ESI are valuable in mapping the take-up rate of recommended techniques and practices, the data collection tools need to be kept up-to-date while still providing comparative data.

The results of the ESI survey highlight disparities across a range of 16 countries in terms of their adoption rates of software development best practice. The next section explores the potential role of national culture in explaining such differences.

### 4 Analysis of Adoption by Country

The difference in the ESI best practice adoption levels across Europe raises the question of national cultural issues, which has been briefly explored by Dutta, Lee and Van Wassenhove (1998a) who used Ronen and Shenkar's (1985) national culture clusters to compare adoption of clustered countries. Dutta, Lee and Van Wassenhove (1998a) observed that Germany and Austria behaved similarly; however, with respect to Scandinavian countries, they found considerable variance warranting further research. The clusters derived by Ronen and Shenkar measure work goals, values, needs, and job attitudes and are named Anglo, Germanic, Nordic, Latin European, Latin American, with Australia classed in the Anglo cluster along with United Kingdom, Ireland, USA, Canada, New Zealand and South Africa (Mahoney et al. 2001).

Hofstede's initial research was based on the analysis of 116,000 IBM employees from more than 50 countries surveyed over 6 years from 1967. His results have been applied by researchers and verified by many replications (Hofstede and Hofstede 2005). In defending the strength of the underlying theory of static national culture in the face of global use of email and other technology, Hofstede and Hofstede believe 'the software of the machines may be globalised, but the software of the minds that use them is not' (2005, p.330). A dimension is defined as 'an aspect of culture which can be measured relative to other cultures' (Hofstede and Hofstede 2005, p.23) and the five dimensions are explained in table 2.

**Table 2. Hofstede's dimensions (Mahoney et al. 2001)**

<i>Dimension</i>	<i>Low Score Value</i>	<i>High Score Value</i>
Power distance	society de-emphasizes the differences between citizen's power and wealth	inequalities of power and wealth within society
Individualism vs collectivism	collectivist nature with close ties between individuals	individualism and individual rights are paramount
Uncertainty avoidance	tolerance for variety of opinions, less concern about ambiguity, uncertainty	low tolerance for uncertainty and ambiguity
Masculinity vs femininity	value social relevance, quality of life, welfare of others	aggressive goal behaviour, high gender differentiation, males dominate
Long term vs short-term orientation	place less emphasis on hard work, perseverance	embraces long-term devotion to traditional, forward thinking values

The scores for each of Hofstede's five dimensions for the countries where the ESI best practice survey was conducted are listed in Table 3.

In relating Hofstede's dimensions to the adoption of best practice techniques, it could be expected that higher adoption may be associated with low uncertainty avoidance (willingness to adopt new techniques), and low individualism (conformance to group working practices). Hofstede's scores indicate that Australians, as shown in Table 3, compared to others, have low uncertainty avoidance (would be quick to adopt innovations) but high individualism (resistant to standard work practices) (Mahoney et al. 2001). To investigate if an association exists between the dimensions and the adoption of best practice, Pearson's correlation tests were performed, based on the data in tables 1 and 3.

**Table 3. List of relevant Hofstede Dimension Scores for each dimension by country (Hofstede and Hofstede 2005)**

Country	Power Distance	Individualism	Uncertainty Avoidance	Masculinity	Long term orientation
Australia	36	90	51	61	31
Austria	11	55	70	79	31
Belgium	65	75	94	54	38
Denmark	18	74	23	16	46
Finland	33	63	59	26	41
France	68	71	86	43	39
Germany FR	35	67	65	66	31
Great Britain	35	89	35	66	25
Greece	60	35	112	57	Not available
Ireland	28	70	35	68	43
Israel	13	54	81	47	Not available
Italy	50	76	75	70	34
Netherlands	38	80	53	14	44
Norway	31	69	50	8	44
Spain	57	51	86	42	19
Sweden	31	71	29	5	33

Before discussing the analysis, it is important to recognise some differences which exist between Hofstede's populations and those involved in the ESI surveys. Firstly, Hofstede's score for Great Britain was used in the analysis with the United Kingdom ESI survey data, although it was not reported if any Northern Ireland firms responded to the ESI survey. Secondly, Hofstede's score for Germany relates to the time period when Germany was divided and represents the west area (FR) whereas the ESI data was collected from a united Germany. Finally, Hofstede's score for Australia was used in the analysis with the survey data collected from Queensland as it was considered that the Queensland responses were representative of the Australian software development population.

As shown in table 4, no significant correlations were identified of best practice adoption with any of Hofstede's dimensions. Therefore, Hofstede's theory is not validated for the case of software development organisations across 16 countries. The next section discusses reasons for the lack of correlations, the sample upon which Hofstede based his theory, and also the notion of national culture.

**Table 4. Pearson Correlation test results Hofstede's dimensions and Best Practice adoption.**

Hofstede's Dimensions		Org. Issues	Standards & Processes	Metrics	Control	Tools	Overall
Power Distance N=16	Pearson Correlation	.058	.141	.349	.216	.126	.172
	<i>p</i> (2-tailed)	.831	.604	.185	.421	.643	.524
Individualism N=16	Pearson Correlation	-.170	.120	-.001	-.154	.154	-.001
	<i>p</i> (2-tailed)	.530	.658	.997	.569	.569	.997
Uncertainty Avoidance N=16	Pearson Correlation	.247	.092	.295	.289	-.005	.187
	<i>p</i> (2-tailed)	.357	.735	.267	.278	.985	.488
Masculinity N=16	Pearson Correlation	.169	.170	.190	.244	.108	.211
	<i>p</i> (2-tailed)	.531	.530	.480	.362	.691	.433
Long Term Orientation N=14	Pearson Correlation	.171	.043	.177	.019	.417	.158
	<i>p</i> (2-tailed)	.558	.885	.544	.949	.138	.590

(Note: Long term orientation dimension scores were not available for Greece and Israel)

### 5 Discussion

This study raises doubts about whether it is reasonable to expect that software development best practice adoption would be related to Hofstede's national culture dimensions. Other researchers have successfully applied Hofstede's scores in IT related research, for example, Frank et al. (2001) found evidence that innovativeness correlates with low uncertainty avoidance in a study of the adoption of mobile technology across Finland, Germany and Greece; and more recently, Borchers (2003) applied Hofstede's theory to understand project problems experienced by project teams of Indian, Japanese and American software developers.

In recent years, Hofstede's analysis and model have drawn criticism (McSweeney 2002). One of the issues raised by Myers and Tan (2002) and other researchers concerns the ability to generalise Hofstede's scores, considering the limited demographic variation in the population surveyed: the survey data was mainly from male employees of one multinational organisation (IBM) and severely limited in terms of the range of ages of respondents. Although the proportion of female computing professionals in Australia is low at 22 percent (ABS 2005), Hofstede's sample did not accurately represent female workers. Another related issue, also explored by Myers and Tan, is whether national culture remains static—as claimed by Hofstede—or contested, temporal and emergent as claimed by Kahn (1989)

The diversity of birthplaces in the Australian workforce of computing professionals (presented in §2) provides support for the temporal and emerging nature of national culture, influenced by changes in the ethnic and racial mix of the population. It is suggested, therefore, that Hofstede's sample of Australian IBM male employees from 1967 to 1973 may not represent the diverse workforce which exists in Australia today, and which is gradually changing as more migrants are employed in ICT positions.

Hofstede maintains that although people may have similar occupational and organisational culture, as evidenced through similar practices, national culture is about deeply held values and is part of the mental software acquired from family and school during the first ten years of life (Hofstede and Hofstede 2005). This issue has been explored by Shore and Venkatachalam (1995) who recognise that organisational culture may play an intervening role in the influence of national culture on organisational behaviour. Therefore, the disparate rates of adoption by country found in analysing the ESI survey results (in §3) may be caused by factors other than the deeply held cultural values. For example, the practices used by firms may originate from the methods and techniques taught in the curriculum of local colleges and universities, or individual government purchasing policies promoting various methodologies such as CMM. Factors such as these may foster standardisation within the local industry, but may be the source of variations when comparing diverse geographical groups of software development firms. The ESI best practice survey, although providing a valuable snapshot of the state of practice in many countries, was designed to measure behaviour, not culture. Culture is very difficult to measure and it has been recommended that in-depth case studies, discourse analysis and ethnographies are required rather than surveys (Myers and Tan 2002; Sharp et al. 2000).

### 6 Conclusion

The software development industry has become globalised due to trends such as the maturation of software industries in developing countries, collaborative teams covering extended geographic areas, and migration of computing professionals. Multinational corporations, software purchasers, firms undertaking off-shore outsourcing, and firms with teams of local or distributed developers all need to be aware that practices used by software developers vary according to geographic location, in the same way senior managers need to be aware of local business practices in negotiating contracts with international business partners.

The migration analysis provided a profile of the multinational nature of the Australian computing workforce. This workforce analysis provided evidence for the argument to view national culture as temporal and emergent, rather than the notion that national culture is static. Furthermore, the correlation analysis failed to prove a link between any of Hofstede's cultural dimensions and practices

surveyed in the ESI best practice survey representing 16 countries. Consequently, researchers are advised against the use of simplistic frameworks such as that espoused by Hofstede, and are encouraged to explore the concept of national culture with appropriate research methodologies.

#### Acknowledgement

The authors thank the ESI for granting permission to use the Best Practice Questionnaire, and the Software Quality Institute of Griffith University for funding the survey. Associate Professor Terry Rout of Griffith University provided valuable assistance during the design and execution of the study.

#### References

- AAP. 2003. ICT jobs cut from govt's migrant skills list. in *Sydney Morning Herald*. Sydney.
- ABS. 2005. 2001 Census of Population and Housing: Number of Employed Persons by State of Usual Residence by Birthplace by Occupation. in *Unpublished*. Brisbane.
- Bajkowski J. 2005. India dominates IT visa numbers. *Computerworld* **28**(6): 1,30.
- Biro M, Messnarz R, Davison A. 2002. The impact of national cultural factors on the effectiveness of process improvement methods: the third dimension. *Software Quality Professional* **4**(4): 34-41.
- Borchers G. 2003. The software engineering impacts of cultural factors on multi-cultural software development teams. Pp. 540-545 in *Proceedings of the 25th International Conference on Software Engineering*. Portland, Oregon: IEEE Computer Society Washington, DC, USA.
- Braue D. 2004. Certification: What's in a name? in *Technology and Business Magazine Australia*.
- Cheung HK, Burn JM. 1994. Distributing global information systems resources in multinational companies: a contingency model. *Journal of Global Information Management* **2**(3): 14-27.
- Costlow T. 2003. Globalization drives changes in software careers. *IEEE Software* **20**(6): 14-16.
- Cragg PB. 2002. Benchmarking information technology practices in small firms. *European Journal of Information Systems* **11**(4): 267-282.
- Cusumano MA, MacCormack A, Kemerer CF, Crandall B. 2003. Software development worldwide: the state of the practice. *IEEE Software* **20**(6): 28-34.
- Dutta S, Lee M, Van Wassenhove L. 1999. Software engineering in Europe: a study of best practices. *IEEE Software* **16**(3): 82-90.
- Dutta S, Lee MA, Van Wassenhove LN. 1998a. Adoption levels of software management best practices in European organizations. in *European Conference on Software Process Improvement*. Monte Carlo.
- Dutta S, Van Wassenhove LN. 1997. An empirical study of adoption levels of software management practices within European firms. Fontainebleau, France: INSEAD.
- Dutta S, Van Wassenhove LN, Kulandaiswamy S. 1998b. Benchmarking European software management practices. *Communications of the ACM* **41**(6): 77-86.
- ESI. 1996. Software Best Practice Questionnaire Comparative Analysis of 1995 and 1996. Bilbao: European Software Institute.
- . 1997. 1997 Software Best Practice Questionnaire: Analysis of Results. Bilbao: European Software Institute.
- . 1998. Software Best Practice Questionnaire: Trend Analysis 1995-1997. Bilbao: European Software Institute.
- Finkelstein A, Kramer J. 2000. Software engineering: a roadmap. in *The Future of Software Engineering*, edited by Finkelstein, A: ACM Press.
- Frank L, Sundqvist S, Puumalainen K, Taalikka S. 2001. Cross-cultural comparison of innovators: empirical evidence from wireless services in Finland, Germany and Greece. in *ANZMAC Conference*.
- Hofstede G. 1980. Motivation, leadership, and organization: do American theories apply abroad? *Organizational Dynamics*(Summer): 42-63.
- Hofstede G, Hofstede GJ. 2005. *Cultures and Organizations: Software of the Mind*. New York: McGraw-Hill.
- Howarth B. 2004. Software shut-out. *Business Review Weekly* **26**(1): 29-31.
- Jones C. 2003. Variations in software development practices. *IEEE Software* **20**(6): 22-27.
- Kahn JS. 1989. Culture: demise or resurrection? *Critique of Anthropology* **15**(1): 32-49.
- Karolak DW. 1998. *Global Software Development: managing virtual teams and environments*. Los Alamitos, CA: IEEE Computer Society Press.
- Khandelwal VK, Ferguson JR. 1999. Critical success factors (CSFs) and the growth of IT in selected geographic regions. in *32nd Hawaii International Conference on System Science*.
- Krutchon P. 2004. Analyzing intercultural factors affecting global software development – a position paper. in *Position paper*. Victoria, Canada: University of British Columbia.
- MacGregor E, Hsieh Y, Kruchten P. 2005. Cultural patterns in software process mishaps: incidents in global projects. in *Human and Social Factors of Software Engineering (HSSE)*. St Louis, Missouri.

- Mahoney D, Trigg M, Griffin R, Pustay M. 2001. *International Business: A Managerial Perspective*: Pearson Education Australia.
- McCarthy J. 2003. Building Apps on the Fly. *Information Age* 18 June: 49-50.
- McSweeney B. 2002. Hofstede's model of national cultural differences and their consequences: a triumph of faith - a failure of analysis. *Human Relations* **55**(1): 89-118.
- Mili H, Mili F, Mili A. 1995. Reusing software: issues and research directions. *IEEE Transactions on Software Engineering* **21**(6): 528-561.
- Myers MD, Tan FB. 2002. Beyond models of national culture in information systems research. in *Human Factors in Information Systems*, edited by Szewczak, E and Snodgrass, C. Hershey, PA: IRM Press.
- Paulish DJ, Carleton AD. 1994. Case studies of software-process-improvement measurement. *Computer* **27**(9): 50-57.
- Phillips M. 2003. CMMI appraisal tutorial. in *Australian Software Engineering Process Group (SEPG)*. Surfers Paradise.
- Pressman RS. 2003. *Software Engineering Resources*. R.S. Pressman & Associates.
- Ronen S, Shenkar O. 1985. Clustering countries on attitudinal dimensions: a review and synthesis. *Academy of Management Review* **10**(3): 435-454.
- Saran C. 2001. The road to software excellence for SMEs. ComputerWeekly.com.
- Sharp H, Robinson H, Woodman M. 2000. Software engineering: community and culture. *IEEE Software* **17**(1): 40-47.
- Shore B, Venkatachalam AR. 1995. The role of national culture in systems analysis and design. *Journal of Global Information Management* **3**(3): 5-14.
- Vanstone A. 2005. Record number of skill stream migrants in 2004-2005. Canberra: Department of Immigration and Multicultural and Indigenous Affairs.

### Author CVs

#### Aileen Cater-Steel

Dr Aileen Cater-Steel is a Senior Lecturer in Information Systems at the University of Southern Queensland (USQ) Australia. Her research interests are software process improvement, especially for small development firms, and IT service management process improvement.

#### Mark Toleman

Dr Mark Toleman is an Associate Professor in the Department of Information Systems at USQ. He has over 20 years research experience in areas ranging from information systems development methodologies to human-computer interaction to information systems in agriculture, to name a few. He now concentrates his research on systems development methodologies and information systems education, and he has published over 90 articles in books, refereed journals and refereed conference proceedings.

# The Human Factor Deployment for Improved Agile Quality

*Kerstin V. Siakas, Errikos Siakas*

*Alexander Technological Educational Institute of Thessaloniki<sup>1</sup>,  
Department of Informatics,  
P.O. Box 141, GR-57400 Thessaloniki, Greece  
E-mail: siaka@it.teithe.gr,  
serik@mailbox.gr*

## **Abstract**

Many new paradigms and methodologies have been developed aiming to find solutions to the persisting software crisis. The most eminent paradigm to make software development more predictable is Software Process Improvement (SPI), which was inspired from other engineering disciplines. SPI is a worldwide movement, which accentuates a disciplined approach embracing repeatable processes and continuous improvements. The conviction in SPI is that a high-quality process will deliver high-quality products. The main critique against SPI is that it creates a lot of bureaucracy and constrains innovation. The fundamental of Agile Methodologies on the contrary, are flexibility and quick response to requirements changes. No detailed plans are made. Instead changing requirements are considered a necessity to sustain and improve customers' competitive advantage. Many companies have adapted, tailored and customised the agile approach to fit their own organisational practices. Simultaneously the society has become increasingly connected and the interactions between people have amplified. It has become increasingly obvious that the human factor play an important role in sustaining and adding business value and competitive advantage.

In this paper we examine the agile methodology from a human point of view. In order to identify the human and cultural factors important for the success of agile methods, parallels are drawn to success factors identified in recognised management approaches, such as the Effective Technical and Human Implementation of Computer Systems (ETHICS) and the Total Quality Management (TQM), which both emphasise the importance of the human (soft) factor, such as empowerment, participation, people motivation and team work.

The parallels indicate that concentration on human factors is not a new issue as such, but more likely that the maturity of the software discipline and the combination of factors taken into consideration with the agile methodology has created an agile culture on its own; a movement with young, enthusiastic software developers who claim that they have solved the problem of frequently changing requirements by taking customers on board in the whole development life cycle, by frequent iterations and high commitment and work morale.

The concept of culture is analysed for deeper understanding of human and cultural dynamics in the agile professional culture. We argue that the development of a flexible organisational culture is required to embrace the agile professional culture for added business value and competitive advantage.

## **Keywords**

Agile Development, Soft issues, Human factor, ETHICS, TQM, Agile Culture

---

<sup>1</sup> This work is funded by the Greek Ministry of Education (25%) and European Union (75%) under the EPEAEK II program "Archimedes II".

### **1 Introduction, Motivation and Perspectives**

Agile techniques have been used since the beginning of software development. As systems grew a chaotic situation appeared often characterised as 'code and fix'. In order to make software development more predictable emphasis became on Software Process Improvement (SPI) inspired from other engineering disciplines, which had acknowledged that a high-quality process delivers products of predictable quality. The organisations did not want to be dependent on uncontrollable champions and thus the repeatable processes were employed by roles or actors or in fact changeable software developers. Although there is evidence that SPI methodologies have improved quality and productivity of software [Fenton et. al., 1994] in particular regarding predictability, stability and high assurance [Boehm and Turner, 2003, Herbsleb et. al, 1994], the software industry is still many years away from becoming a mature engineering discipline [Georgiadou et. al, 2003]. The most frequent criticism and resistance to SPI is bureaucracy. The Agile methodologies, also known as light-weight methodologies [Agile Manifesto, 2006], were developed in the mid 1990s as part of the reaction against the bureaucratic plan-driven process-based approach and the traditional life cycle models, which were considered inadequate [Beck 2000; 2003]. Instead incremental design and rapid prototyping was introduced promising lower defect rates and faster development times together with a solution to frequently changing requirements [Boehm and Turner, 2003]. In Agile development the phases are overlapping with design starting during analysing and coding starting during detailed design. The test-driven development embraces test scenarios before development and testing along with implementation. Most agile methods empower developers to respond to changing customer requirements, even late in the life cycle. This promises increased customer satisfaction [Beck, 2001; Holcombe, 2005]. Acceptance of changes also empowers the customers and end-users. However, the high iteration frequency has consequences for contracts variables, such as scope, price and time creating a need for flexible contracts [Geschi et al., 2005], which in turn, may be a drawback for the customer's cost-analysis plans.

For increased understanding of the significance of the human factor in Agile development and its evolution from know-how and proven practices in software development, we draw parallels to success factors identified in Effective Technical and Human Implementation of Computer Systems (ETHICS) and the Total Quality Management (TQM). Both these widely recognised management approaches highlight the significance of soft human issues, such as motivation, empowerment, participation and team work. Furthermore, our study unfolds the incentives behind the new agile professional culture by analysing layers of culture and by revealing its artefacts. Finally we claim that democratic type of organisations, which have horizontal hierarchy emphasising flexibility and spontaneity, are the most suitable organisational cultures to embrace the Agile professional culture.

### **2 The Human Approach**

The multidisciplinary character of Information Systems prompts for a soft human approach and effective Human Resource Management (HRM). A socio-technical approach recognises the interaction of technology and people. It produces work systems that are both technically efficient and have social characteristics which lead to high job satisfaction [Mumford, 2000]. Such an approach takes account of the fact that different individuals and groups have their own needs, interests and values. Below we analyse two management approaches, which emphasise a socio-technical viewpoint, namely the Effective Technical and Human Implementation of Computer Systems (ETHICS) [Mumford, 2001] and the Total Quality Management (TQM) [Deming, 1986] and simultaneously we draw parallels to agile development in order to understand the strengths and weaknesses of agile development for continuous improvements.

## 2.1 ETHICS – parallels with Agile development

Mumford [2000], one of the pioneers in the socio-technical movement created the ETHICS (Effective Technical and Human Implementation of Computer Systems) methodology in the 1980s. The ETHICS philosophy evolved from the organisational behaviour science and is based on the principles of participation, both by members of the team, the department and the organisation in the relevant decision making process as well as by user participation. Management theories that emphasise participation consider it as morally right; people should be able to determine their own destinies. Also they may consider that participation will make it easier to create a committed workforce. Another view is that participation is a valuable educational experience and provides understanding and knowledge that can assist an organisation to more effectively realising its objectives.

Although ETHICS was developed to be used by an expert group (software professionals) to design a system for a non-participating client or user, its most important advantage lies in the fact that it can assist users to become partners in the design process, since it provides a systematic methodology and set of analytical tools which enable the users to analyse their own efficiency and job satisfaction needs, set design objectives to improve efficiency and job satisfaction and develop strategies which will achieve these objectives. Mumford [2000] argues that participation brings job satisfaction, which in turn creates commitment.

In agile development participation is one of the key issues. Collective code ownership, pair programming, user involvement and team rotation are examples of participation. Our perception of the reason for the enthusiastic software developers in agile development is that they have high levels of job satisfaction because of broadened participation and their enthusiasm is an expression of their job satisfaction. However, more scientific evidence needs to be collected to statistically prove this relationship.

In today's competitive world of constantly fluctuating demands on organisations in terms of rapidly changing technology and amending customer needs, change is an inevitable process all employees in a organisation are involved in, either as every-day small-scale changes (incremental), such as adopting a new version of a software system, or in a large-scale changes (radical), such as a cultural change or change of strategy with subsequent changes in structure and operations. ETHICS has three objectives related to the management of change:

- The first objective seeks to legitimate a value position in which the future users of computer systems at all organisational levels play a major part in the design of these systems. The argument is that the likelihood for both job satisfaction and efficiency increases when people are able to influence the design of their own work situations. If we draw parallels to agile development we can conclude that the collective ownership of the code and empowerment of software engineers make them feel that they can influence their own work situation.
- The second objective is to enable groups concerned with the design of computer systems to set specific job satisfaction objectives in addition to the usual technical and operational objectives. Here it is argued that unless job satisfaction and quality of working life objectives are made explicit, the human impact will be unpredictable because it has not been consciously planned for. The result can be that staff may respond in a negative way including absenteeism and increased labour turnover. All of these responses can impose high financial costs on management. If we draw parallels to agile development we find for example that some of the 12 principles in Extreme Programming concern job satisfaction. As examples we can mention the following XP principles and their job satisfaction objective:
  - *40 hours week*: happy programmers working in a stress-free environment
  - *Pair programming*: shared responsibility, shared knowledge, improved communication, increased code quality and productivity (coding standards are followed closer and code reviewed simultaneously with creation)
  - *On-site customers*: real life users providing scope, setting priorities, resolving ambiguities and providing test scenarios
- The third objective is to ensure that any new technical system is surrounded by a compatible, well functioning organisational system. This covers the specification of roles and relationships within the department and the design of work procedures, individual jobs and workgroup ac-



tivities. If we draw parallels to agile development we can find that there are specific rules regarding roles, relationships and work procedures [Robinson and Sharp, 2003]. In agile development the everyday stand-up meetings (problem formulation and participated decision making take place and guide action), story cards (components that have to be implemented during next iteration) and system metaphors (a set of small metaphors from which responsibilities, class, method and variables are derived; a fundamental facilitator for sustaining shared vision) are examples of work procedures and workgroup activities.

Table 1 below is a summary of the success-factors of the ETHICS model and the parallels drawn to Agile Development.

**Table 1: Comparison of success factors in ETHICS and Agile Development**

	<b>ETHICS</b>	<b>Agile Development</b>
<b>(HRM)Human Resource Management</b>	- Job satisfaction & quality of working life objectives are made explicit	- Individuals & interactions over processes and tools - 40 hours week
<b>Participation</b>	- Considered to create job satisfaction and commitment	- Collective code ownership - Pair programming - Team rotation
<b>Influence on own work</b>	- Considered to increase job satisfaction and efficiency	- Collective ownership of the code - Empowerment of software engineers
<b>Knowledge sharing</b>	- Considered a process of negotiation & reconciliation of interests	- Pair Programming - Team rotation
<b>Communication</b>	- Communication between different stakeholders is supported	- Increased Communication
<b>Customer Involvement</b>	- The analytical tools & the systematic methodology enable user involvement	- On-site customers
<b>Organisational System</b>	- Specification of roles & relationships - Design of work procedures, individual jobs and workgroup activities	- Specific rules regarding roles, relationships & work procedures

By analysing the influence of the human factors in ETHICS that bring job satisfaction and subsequently commitment and improved efficiency and by comparing them with agile development we found similar factors and therefore we draw the conclusions that participation improves job satisfaction, commitment and efficiency, which are considered success factors in agile development. However, in future a more scientific approach is needed to statistically prove the relationships.

## 2.2 Total Quality Management features in Agile Methodologies

The Total Quality Management (TQM) concept [Deming, 1986] was originally developed after the Second World War in Japan for manufacturing industries. Successful implementation of TQM methods contributed to Japan's industrial success especially in quality and reliability. The TQM philosophy adopts a disciplined and structured approach focusing on customers and continuous improvement emphasising the importance of people, culture and process management by documentation and monitoring of processes, compliance of results to plans according to Deming's plan-do-check-act circle [Deming, 1986]. The principles of TQM were adapted gradually in the software industry [Schulmeyer et. al, 1992] and SPI models, such as ISO9001:2000 [ISO, 2006], Capability Maturity Model Integrated [CMMI, 2006], Bootstrap [Kuvaja et. al, 1994] and ISO-15504 [Dorling, 2006] are all build on the principles of the Total Quality Management (TQM) philosophy.

TQM requires a cultural change in the organisation [Durrant, 1990]. In the TQM approach Deming [1986] emphasises management responsibility and the intrinsic link between leadership and the quality of processes, which in turn should lead to quality in products and services. Management commitment and leadership are the driving factors for motivating employees to strive for continuous process improvement [Mauro, 1999; Mellor, 2005]. The trend today seems to be to integrate product and process quality and their interactions. The TQM principles require quality awareness throughout the whole organisation and a change in attitudes towards quality by imposing a constant thinking of improving

the process and the product.

The Agile methodologies try to congregate plus points and advantages from existing methodologies without bringing about the disadvantages. Some of the features in Agile development are totally in contrast with TQM, such as the plan-driven approach vs. agility, process emphasis vs. product emphasis and much documentation vs. executable lean documentation, others have many similarities, such as customer focus, employee commitment and satisfaction. Below are different characteristics of Agile methodologies discussed in the highlight of TQM.

### 2.2.1 Flexibility

The distinction between predictive (plan-driven / TQM) and adaptive (agile) approaches recognises the fact that achieving plan milestones does not necessarily equate to customer success. Mellor [2005] draw attention to the basic strategic notion in knowledge intensive companies, which is to realise the fact that the future cannot be predicted. Detailed organisational plans for the future do not make sense. A strategic decision in this situation is to base the activities on the company's own resources and keep on developing them [Conner and Prahalad, 1996]. This also means creating a flexible and open organisational culture, which can quickly respond to various business situations [Rantapuska et. al, 1999]. The agile approach emphasises individuals and interactions over processes and seems to have recognised the fact that future cannot be predicted. A strategic competitive advantage in agile development is flexibility to changing requirements with a quick response. No detailed plans are made. Instead changing requirements are considered a necessity to sustain and improve customers' competitive advantage.

One of the central features in the Agile methodologies is cyclic and incremental development. The intangible nature of software [Siakas, 2002] and the difficulties of software professionals to capture and understand the business domain and requirements [Rantapuska et. al, 1999] have created the need for cyclic and incremental development with increased user involvement and early versions of software delivery in order to improve software quality. The high iteration frequency also has consequences for contracts variables, such as scope, price and time and thus the contracts need to be flexible. This may create a drawback for the customer's cost-analysis plans.

### 2.2.2 Employee Satisfaction and Commitment

In TQM management commitment and leadership are the driving factors for motivating employees to strive for continuous process improvement [Mauro and Mauro, 1999; Mauro, 1999]. Subsequently the TQM principles require quality awareness throughout the whole organisation and a change in attitudes towards quality by imposing a constant thinking of improving the process and the product. The TQM philosophy worked well in Japan, because the Japanese culture was well-suited to these kinds of concepts, but when it was transferred to other countries, e.g. the US and Europe, the results were not always as expected [Kaneko et. al, 1995].

The people issues, the most important asset in software development, which is based on brainpower and capability to produce software, are embraced in the agile approach. The basis of agile development lies in small teams working in co-located development environment developing non-safety critical software (Abrahamsson, 2005). Agile development relies on self-directed teams consisting of highly skilled, motivated and innovative software engineers, who are collaborative in team work and self-organised, active learners. To further motivate the team members, they are provided the support they need, and they are trusted in terms of the accomplishment of the required tasks. Trust has shown to be a key factor in dynamic teams [Siakas and Balstrup, 2005]. Trust enables co-operated behaviour by promoting open exchange of information, effective responses to crisis, reduction of conflict [Rousseau et al, 1998; Sydow, 1998] and job satisfaction (the creation of a good fit between the employee's job expectations and the job requirements as defined by the organisation) [Mumford, 2000]. The prime condition for trust is uncertainty, as well as, lack of information and control [Giddens, 1990]. If the activities would be visible and easy to understand no trust would be necessary. Trust is related to absence in time and space and is often discussed in relation to concepts, such as risk, control and power [Imslund, 2003, Das and Teng, 2001].

The commitment required by the Agile methodologies seems to have similarities with TQM, which requires a cultural change by emphasising commitment at all levels in the organisation. The first step in TQM is to convince every employee of his/her own role in total quality. The difference is that in TQM people are thought of as being roles instead of individuals and the processes are designed to work whoever will be in the role, whilst in the Agile methodologies people are considered to be unique, highly creative professionals. In order to obtain commitment empowerment of software developers, communication horizontally and vertically and end-user participation are supported.

In Agile development different processes are created depending on the project. Thus every project can have a different process depending on user needs. Processes imposed by management are by nature resisted. Management and software developers have an equal role in the leadership of projects [Agile Manifesto, 2006]. Also at regular intervals the team reflects on how to become more effective, tunes and adjusts its behaviour accordingly. This could be compared with the continuous quality strive in TQM, which as driving factors comprise management commitment and leadership, communication and collaboration between and within department both horizontally and vertically. In the Agile methodologies the dynamics within the team are the factors determinant for improvement.

Empowerment and employee satisfaction are common themes in both TQM and in Agile Methods. User involvement and frequent iterations increases domain knowledge in the development process, which in turn increases developer motivation, commitment and satisfaction, key elements for success [Abrahamson, 2002; Siakas and Georgiadou, 2003].

### 2.2.3 *User Satisfaction*

One of the basic principles in TQM is customer/user satisfaction, as it seems to be in the Agile approach as well. How can user satisfaction be obtained? The reasons for buying a product depend on the customer. Different customers are likely to have quite different views about what a quality product is [Siakas et. al, 1997]. The customer's concerns on quality factors are rather different from those of the developers and the managers. Some quality factors are important to all stakeholders. Customers are mainly requiring a correct software that fits its purpose, easy to use, ready in time to a price that gives value for money. The behaviour of the software after release is in the customer's interest (not the development process), which in addition to a fit for purpose everyday use of software involves ease of improvements, adaptation to changing requirements and connectability to other systems. Quality attributes considering the end product is for the user of greatest interest. However, quality is a complex condition depending on the opinions and attitudes of the stakeholders concerning different quality attributes. The different quality views may conflict with each other [Siakas and Georgiadou, 2005; Siakas et. al, 1997]. To obtain satisfied customers and repeat orders customers/users are usually put first recognising that user satisfaction and fitness for purpose is the ultimate measurement for high quality. The view of the customer in terms of quality attributes has to be taken seriously into consideration for ensuring user acceptance and satisfaction [Ishman, 1995, Siakas and Georgiadou, 2005]. Quality plays a vital role in achieving a competitive advantage, based on the notion of continuous improvement throughout the entire organisation. In a survey carried out by the authors in four countries, namely Denmark, Greece, Finland and the UK in total 306 software developers on different levels answered to a questionnaire regarding software quality [Siakas, 2002]. On a question regarding the meaning of software quality 85.7% of the respondents considered that software quality means user satisfaction.

TQM requires emphasis on the customer. Trompenaars [1997] explains the customer orientation to be closely connected to the inner-direction or outer-direction of a culture. In outer-directed cultures, like Japan and Singapore, it is a natural behaviour of adaption to try to satisfy the customer. He describes it like a surf-rider who responds to the waves and keeps their balance where others loose theirs.

### 2.2.4 *User Involvement*

The literature has suggested that higher customer involvement also results in higher quality, especially in terms of meeting requirements [Berki et. al, 1997]. In order to ensure conformance to requirements, user satisfaction and competitive advantage agile development entail the user in the entire development process. However, customer identification can be difficult and may require the identification of suitable internal customer representative(s) providing a single point of contact both for the team and senior management on a daily basis. The high iteration frequency in agile development also provides

opportunities for product feedback [Karlström and Runeson, 2005]. It is argued that agile methods make the key business users a very strong partner in assuring quality [Beck, 2003; 2000; Beck and Fowler, 2001]. Rather than completely leaving quality to the IT professionals the key IS users are co-responsible for conformance to requirements. In fact the only part that has adequate extensive domain knowledge needed for verifying fit for purpose (user satisfaction) is the key business user. Thus it is inevitable that the more user involvement in the development process the higher the possibility for conformance to requirements, fit for purpose and user satisfaction and ultimately quality of end product. It is important that IS stakeholders state clearly their objectives and expectations from the software products so that agile software developers can respond to the characteristics by incrementally developing the agile final product and by adopting agile work processes with features that reflect these required objectives.

The TQM approach has a very strong customer/user emphasis. In order to ensure user acceptance and satisfaction the customer has to be put first and quality attributes are determined in the view of the customer/user. The TQM is a user centred approach. Similarly the agile development has a user centred approach. The difference is that the TQM emphasises that a standard process will produce a predictable outcome. In agile development on the contrary the emphasis is on user viewpoints relating to the characteristics of the final product in combination with a daily feed-back mechanism, which increases the rate of feedback on performed work and the speed of discovering erroneous functionality at an early development stage [Karlström and Runeson, 2005; Mellor, 2005]. In Agile methodologies continuous access to business expertise is a main feature [Beck, 2003].

A field-study carried out in Greece [Sfetsos et. al, 2004] showed that customer involvement increased domain experience. However, customer involvement by on-site customer was difficult in practice and other ways of communication had to be invented. Customer involvement is in general considered a potential regarding customised projects, but in software package projects the customer is unknown in the stage of software development. In the future this part of the software market is also likely to increase with eBusiness customers who buy off-the-shelf software. The question seems to be if customers are willing to spend the time required by Agile Methodologies in order to train the developers in the business domain or will they in the future request more multi-skilled software developers.

### 2.2.5 Summary of success factors

Table 2 summarizes the TQM success factors analyzed and parallels are drawn to Agile Development

**Table 2: Comparison of success factors in TQM and Agile Development**

	<b>TQM</b>	<b>Agile Development</b>
<b>Leadership</b>	- Emphasis on management responsibility and the intrinsic link between leadership and the quality of processes	- Management and software developers have an equal role in the leadership of projects
<b>Customer Focus</b>	- Strong customer/user emphasis - Customer satisfaction focus	- Customers on board - Increase customer satisfaction
<b>Flexibility</b>	- Predictive approach - Long term plans - Emphasis on processes - Compliance to standards - Bureaucracy - Extended documentation	- Adaptive approach - No detailed plans, short term plans - Flexibility to changing requirements - High iteration frequency - Test driven development - Executable lean documentation
<b>Motivation Job satisfaction</b>	- Management commitment and leadership are the driving factors for motivating employees to strive for continuous process improvement - Replaceable roles	- Self-directed teams consisting of highly skilled, motivated and innovative software engineers - Skilled individuals
<b>Empowerment Commitment</b>	- Empowerment emphasises a cultural change by highlighting commitment at all levels in the organisation	- Empowerment of software developers, communication and end user participation supported
<b>Collaboration</b>	- Collaboration horizontally and vertically between and within departments	- Small teams working in co-located development environment reflecting on efficiency and tuning accordingly

By analysing the human factors in TQM and by drawing parallels to agile development we found that the TQM approach is very bureaucratic in comparison to the flexible agile development, which fast respond to changing requirements. Similarities are found in the emphasis on empowerment of employees, job satisfaction and commitment on all levels, which are considered to improve the total IS quality. The user in both approaches is considered to be the fundament judge of quality and therefore user satisfaction and fitness for purpose are considered to be the fundamental measurement for high quality.

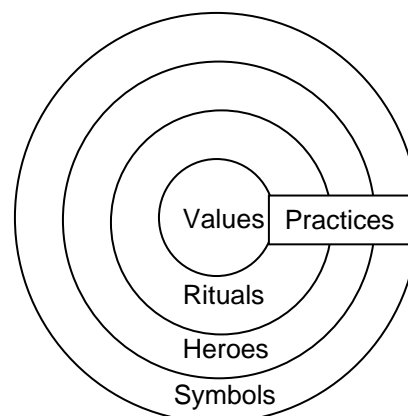
### 3 The Agile Culture

In this section we investigate the agile culture for deeper understanding from a viewpoint of other disciplines, such as anthropology, sociology and organisational behaviour.

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 Stefan 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.

However, there are similarities in the different approaches to identifying and defining culture. Researchers [Kluckhohn, 1951; Kroeger and Kluckhohn, 1952; Kluckhohn and Strodtbeck, 1961, Inkeles and Levison, 1969] relate differences between cultures to different approaches to solving common human problems. Many more recent researchers [Hofstede, 2001; Schein, 1985] seem to have adopted this approach. There are cultures of, e.g. a family, a tribe, a region, a national minority or a nation. There is cultural variation within and among cultures and there are different levels and forms of cultures.

Hofstede [1994, 2001] depicts different layers of culture as an onion (figure 1), indicating that values represent the deepest and symbols the most superficial manifestation of cultures.



**Figure 1: Layers of a culture [adopted from Hofstede, 2001]**

*Symbols* are characteristics of a cultural group, such as words, objects, conditions, acts or characteristics of persons that have a deeper meaning for an individual or a group. Symbols are considered superficial and can easily be changed and copied by other cultural groups.

*Heroes* are persons, real or imaginary, living or dead, who possess characteristics highly prized in a culture - the person everyone will count on when things get tough.

*Rituals* are collective activities carried out for their own sake and are considered as socially essential. Ways of greeting and paying respect to others are examples of rituals.

Symbols, heroes and rituals are visible to outsiders and are collectively called practices. However,

their cultural meaning is invisible and lies in the way these practices are interpreted by the insiders.

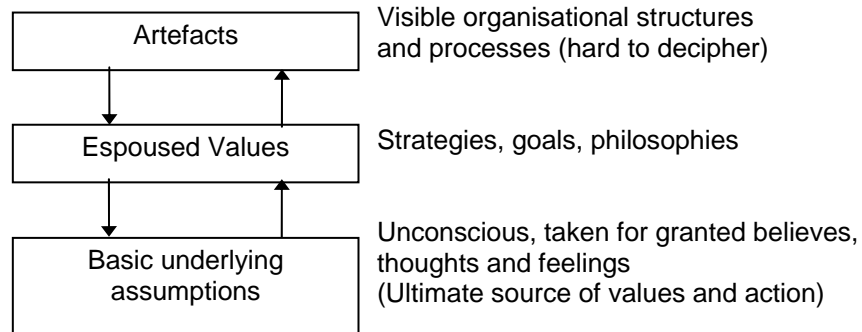
Many business and political meetings have ritual purposes in addition to rational purposes.

*Values* are tendencies to prefer certain states of affairs over others. Values are learnt implicitly since childhood and are taken for granted. They are qualities, principles or behaviours considered morally or intrinsically valuable or desirable; they remain unconscious to those who hold them.

Hofstede [2001] stated that everyone belongs to a number of different groups and categories of people at the same time. He calls this mental programming within people, corresponding to different levels of culture, for example at a national, professional and organisational level.

In agile development there are many examples of the layers above, such as planning games [Robinson and Sharp, 2003], story cards, metaphors instead of requirements ‘release’ machine (a celebration every time some code is released) and other signals for opening, closing and progress [Reifer, 2002]. All these features strengthen the feeling of group belonging and shared values - the culture. Mumford [2000] considers that shared stakeholder values in organisations (strong organisational culture) are important for commercial success in tomorrow’s competitive globalisation.

Similarly with layers of cultures (figure 1) Schein (1985) has depicted levels of cultures and their interaction (figure 2). Levels of cultures apply to organisational culture. The main difference between the two figures lies in the distinction between artefacts (visible daily activities) and espoused values (goals and vision).



**Figure 2: Levels of cultures and their interaction (Adopted from Schein, 1985)**

Basic assumptions according to Schein [1985] are solutions taken for granted to identifiable problems. They evolve as solutions to problems, are repeated over and over again and refer to the implicit, deeply rooted assumptions people share. Basic assumptions are held unconsciously and guide perceptions, feelings and emotions about situations. They are complex, non-confrontable and non-debatable aspects of human group psychology, and involve beliefs (and their interpretation), values and emotions. Basic assumptions can be compared to values in figure 1.

Espoused values refer to a normative or desired vision of the organisation as opposed to activities carried out on a daily basis. An example of espoused value is the mission statement, which defines the long-term vision of the organisation in terms of “where it wants to be and whom it wants to serve”. In mission statements usually the purposes of the organisation and principal business aims are stated together with the key beliefs and values of the organisation, definitions of major stakeholders and ethical principles, which influence code of conduct. Schein [1985] proposed that initially the stakeholders who prevail, namely founders and the leaders, influence groups in the organisation and thus slowly shared values become shared assumptions.

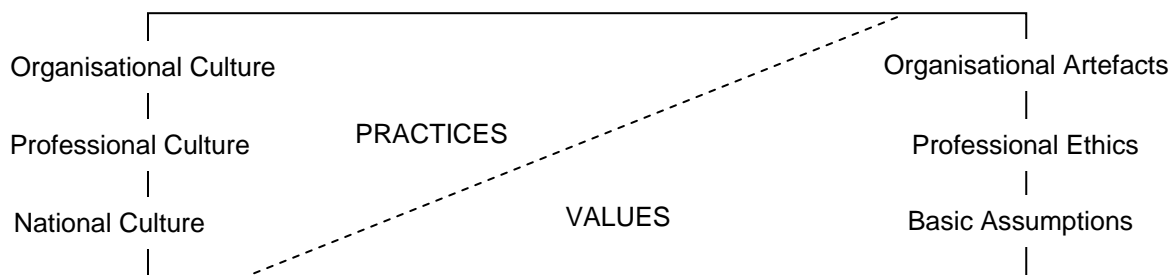
*Artefacts* refer to the physical and socially constructed environment of the organisation. Artefacts are the most visible and most superficial manifestations of an organisation’s culture [Brown, 1998] and include language, symbols, heroes, behaviour patterns, rules and procedures, material objects, physical layouts and technology. Artefacts can be compared with practices in figure 1.

The agile approach can be considered to be a culture on its own. It has the characteristics of a group of people that differentiate themselves from others through a whole set of shared practices including visions, principles, ideals, etc. that emerges in the interaction between members of a group. The eX-

Extreme Programming (XP) for example draw attention to four XP values, namely, communication, simplicity, feedback and courage, the underlying basis for the twelve principles which are translated into practices. These practices are the artefacts for the XP culture.

Having a consistent culture is important in order to create consensus and agreement and to avoid culture clashes friction in the group/team and the whole organisation. The social characteristics of the team members are important. Employment of technically highly competent and competitive software professionals generates the basis for the creation of a strong culture. A project with a very complex problem domain affects even architectural choices, in the form of generating a system that is partitioned into subsystems. Architecture is important, it emerges from the principles and values of the team culture – and culture by its very nature cannot be planned; it can merely be guided. However, it is argued that the agility may lead to more complex and not well-documented systems through a fragmented software development process [Boehm and Turner, 2003; Marciniak, 1994].

Figure 3 shows the relationship between values and practices in organisations.



**Figure 3: Values vs. Practices in Organisations**

Values are deep-rooted basic assumptions expressed by the national culture. The influence from the professional culture is less deep-rooted than the national culture (the way we have been brought up) and is dictated by our profession (agile culture). If the organisational culture supports the professional culture by suitable reward/recognition and career development systems a potential outcome is motivated and satisfied agile software developers.

The agile culture requires active involvement of all team members and seems to be most suitable in democratic-type organisations, which have horizontal hierarchy emphasising flexibility and spontaneity. This type of organisations generates initiative and responsibility approaches. The leadership style is that of co-ordination and organisation. The organisation has flexible rules and problems are solved by negotiations. Employees are encouraged to make contribution to the decision-making process and to the development of the organisation in general. Democratic organisations can be said to be people-oriented. Examples of countries belonging to the Democratic type are Scandinavia, Anglo-Saxon countries and Jamaica [Siakas, 2002; Siakas et. al, 2005].

## 4 Conclusion

The purpose of this paper was to make more explicit the human and cultural dynamics that bear on the success of agile development. Success factors in agile methodologies are recognised to lie in the motivation and exploitation of the dynamics of the human factor. Parallels were drawn to similar success factors (which are considered to improve IS quality and provide added business value) identified in ETHICS (empowerment, job satisfaction, work procedures and workgroup activities) and TQM (employee satisfaction, commitment, user satisfaction and user involvement). Agile methodologies emphasise user satisfaction through user participation, recognition of and response to continuous changing requirements and frequent delivery of products together with adaptive and iterative software development by self-organising teams that recognise that team-members are committed competent professionals able to choose and follow an adaptive process.

The cultural dimension of agile development was investigated and the cultural characteristics for an agile culture were identified. The agile culture imposes a highly competitive environment with cultural,

political and social implications. The organisational culture suitable for agile development was identified and examples of countries, which promote this kind of organisational culture, were proposed.

There is an obvious need for more scientific evidence and further research in order to understand the requirements of agile quality in different cultural and social contexts and to identify controllable and uncontrollable factors for agile IS development. IS quality requires knowledge of different organisational and national cultures, the methods and tools used, the ways they are used and, most importantly, the ways people perceive quality and quality assurance.

## 5 References

- Abrahamsson, P. (2005): Project Manager's Greetings - Agile Greetings, Agile Newsletter 1, 2004, p. 1
- Abrahamsson P. (2002): *The Role of Commitment in Software Process Improvement*, PhD Thesis, University of Oulu, Department of Information Processing Science and Info-tech Oulu
- Agile Manifesto (2006): <http://www.agilemanifesto.org/> retrieved on 20.03.2006
- Beck K. (2003): *Test Driven Development: By Example*. Boston, MA. Addison Wesley
- Beck K. (2000): *Extreme Programming Explained – Embrace Change*. Reading, MA: Addison Wesley Longman, Inc.
- Beck K., Fowler M. (2001): *Planning Extreme Programming*, Boston, MA: Addison Wesley
- Berki, E., Georgiadou, E., & Siakas K., (1997). A Methodology is as Strong as the User Involvement it Supports, International Symposium on Software Engineering in Universities - ISSEU, Rovaniemi, 7-9 March, pp.36-51
- Boehm Berry and Turner Richard (2003): *Balancing Agility and Discipline: A Guide for the Perplexed*, Addison Wesley Professionals
- Brown Andrew D. (1998): Organisational Culture, *Financial Times Management*, Pitman Publishing, 2nd Ed.
- Checkland Peter, Scholes Jim (1990): *Soft Systems Methodology in Action*, Chichester, Wiley
- CMMI (2006): Retrieved from <http://www.sei.cmu.edu/cmmi/> on 21.03.2006
- Conner Kathleen R and Prahalad C. K. (1996): A Resource-based theory of the Firm: Knowledge Versus Opportunism, *Organization Science*, Vol. 7, No. 5, Sept./ Oct.
- Das T. K., Teng B. S. (2001): Trust, Control and Risk in Strategic Alliances, An Integrated Framework, *Organization Studies*, February, 22 (2), pp. 251-283
- Deal T. E., Kennedy A.A. (1982): *Corporate Cultures: The Rites and Rituals of Corporate Life*, Reading, Mass., Addison-Wesley
- Deming W. Edwards (1986): *Out of the Crisis: Quality, Productivity and Competitive Position*, Massachusetts, USA
- Durrant P. W. (1990): *TQM and the Cultural Changes Required*, 3rd Conference Total Quality Management, June
- Georgiadou E., Siakas K., Berkis E. (2003): Quality Improvement through the Identification of Controllable and Uncontrollable Factors in Software Development, EuroSPI 2003 (European Software Process Improvement Conference), Graz, Austria, 10-12.12.2003, pp. IX 31-45
- Giddens A. (1990): *The Consequences of Modernity*, The Stanford University Press
- Groeschl Stefan, Doherty Lis (2000): Conceptualising Culture, Cross Cultural Management, *An International Journal*, Vol. 7, Number 4
- Fenton Norman, Whitty Robin, Iizuka Yoshinori (eds) (1994): *Software Quality Assurance and Measurement, A Worldwide Perspective*, International Thomson Computer press, London
- 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
- Holcombe, M., (2005). Extreme Programming and Agile Processes in Software Engineering, 6th International Conference, XP 2005, Sheffield, UK, June 18-23, Proceedings



## Session 4: SPI and Human Factors

- Holmes Monica C. (1005): *The Relationship of Cross-cultural Differences to the Values of Information Systems Professionals within the Context of Systems Development*, PhD Dissertation, Denton, Texas
- Hofstede Geert (2001): *Culture's consequences: comparing values, behaviours, institutions, and organisations - 2nd Ed.* - Thousand Oaks, Calif.; London: Sage Publications
- Imsland Vegar (2003): *The Role of Trust in Golbal Outsourcing Relationships*, Candidate Science Thesis, Oslo University, Department of Informatics
- 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
- Ishman Michael Dean (1995): *A Cross-cultural model of User Satisfaction with Information Systems*, PhD dissertation, University of New York, Buffalo
- ISO (2006): <http://www.iso.org> , retrieved on 20.03.2006
- Kaneko Ryuzo, Kadota Yasuhiro, Ohba Shogo (1995): Behaviour Analysis Makes the Company Mature, Chap. 10, in Norman Fenton, Robin Whitty, Yoshinori Iizuka (eds): *Software Quality Assurance and Measurement, A Worldwide Perspective*, International Thomson Computer press, London
- Karlström and Runeson (2005), Combining Agile Methods with Stage-gate Project Management, *IEEE Software*, May/June
- Kluckhohn F., Strodtbeck F. (1961): *Variations on Value Orientations*, Evanstone, Illinois: Row, Peterson and Company
- Kluckhohn F.(1951): Values and Value-Orientations 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, *Harvard Business Review*, Cambridge
- 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
- Marciniak John J. (ed.) (1994): *Encyclopaedia of Software Engineering, Software 2000: A view of the future*: Eds: Brian Randell, Gill Ringland and Bill Wulf. ICL and the Commission of European Communities, John Wiley & Sons
- Mauro Joyce A., Mauro Nicholas J. (1999): The Deming Leadership Method: A Behavioural and Technical Approach in *Cross-Cultural Management*, Vol. 6 Num. 3
- Mauro Nicholas J. (1999): The Deming Leadership Method and Profound Knowledge: A Global Prescription in *Cross-Cultural Management*, Vol. 6 Num. 4
- Mellor Stephen J. (2005): Adapting Agile Approaches to Your Project Needs, *IEEE Software*, May/June 17-34
- Mumford Enid (2000): Socio-technical Design: An Unfulfilled Promise or a Future Opportunity? In Baskerville Richard, Stage Ian, DeGross Janice I (eds.): *Organisational and Social Perspectives on Information Technology*, IFIP TC8 WG8.2 *International Working Conference*, June, Aalborg, Denmark, Kluwer Academic Publishers
- Rantapuska Torsti, Siakas Kerstin V., Sadler Chris, Mohamed Walaa E. (1999): Quality Issues of End-user Application Development, *The fourth International Conference on Software Process Improvement - Research into Education and Training, INSPIRE '99*, Crete, Greece 9-11 Sept
- Reifer Donald J, (2002): How Good Are Agile Methods, *IEEE Software*, July/August 16-18
- Robinson and Sharp, (2003): XP Culture: Why the twelve practices both are and are not the most significant things, *Proceedings of the Agile Development Conference (ADC'03)*
- Rousseau D. M., Sitkin S. B., Burt R. S., Camerer C. (1998) : Not so different after all : A cross-discipline view of trust, *The Academy of Management Review*, July, 23 (3), pp. 394-404
- Schein E. H. (1985): How Culture Forms, Develops and Changes, in Kilmann R.H., Saxton M.J. Serpa R. (eds.): *Gaining control of the Corporate Culture*, San Francisco, Jossey Bass
- Schulmeyer G. Gordon, McManus James I (1992): *Total Quality Management for Software*, London, Chapman and Hall

- Sfetsos, P., Angelis, L., Stamelos, I., Bleris (2004): G. Evaluating the Extreme Programming System - An Empirical Study, *Fifth International Conference on Extreme Programming and Agile Processes in Software Engineering*, Garmisch-Partenkirchen, Germany, June
- Siakas Kerstin V. and Georgiadou Elli (2005): PERFUMES: A Scent of Product Quality Characteristics, *The 13th Software Quality Management Conference, SQM 2005*, March, Gloucestershire, UK (2005)
- Siakas Kerstin V., Balstrup Bo: Global Software (2005): Sourcing by Virtual Collaboration? *EuroSPI 2005 (European Software Process Improvement and Innovation, Conference)*, 9-11 November, Budapest, Hungary
- Siakas Kerstin V., Elli Georgidaou, Eleni Berki (2005): Agile Methodologies and Software Process Improvement, *IADIS (International Association for development of the Infor-mation Society) International Virtual Multi Conference on Computer Science and In-formation Systems (MCCSIS 2005) - SEA (Software Engineering and Applications)*, 26 April
- Siakas Kerstin (2002): *SQM-CODES; Software Quality Management – Cultural and Organ-isational Diversity Evaluation*, PhD dissertation, London Metropolitan University
- Siakas Kerstin V., Georgiadou Elli (2003): The Role of Commitment for successful Software Process Improvement and Software Quality Management, *The 11th Software Quality Management Conference, SQM2003*, 23-25 April, Glasgow, UK , pp. 101-113
- Siakas Kerstin V., Berki Eleni, Georgiadou Elli, Sadler Chris (1997): The Complete Alpha-bet of Quality Software Systems: Conflicts and Compromises, *7th World Congress on Total Quality & Qualex 97*, New Delhi, India, 17-19 February, pp. 603 – 618
- Sydow J. (1998): Understanding the Constitution of Interorganizational Trust, in C. Lane, R. Bachman (eds). *Trust Within and Between Organizations*, Capter 1, pp. 33-64, Oxford University Press
- Trompenaars Fons, (1997): *Hampden-Turner Charles: Riding the waves of Culture*, Nicholas Brealey Publishing Limited (1997)

### Author CVs

Kerstin V. Siakas

Kerstin Siakas is an Assistant Professor in the department of Informatics at the Alexander Technological Educational Institute of Thessaloniki, Greece. Her teaching includes Management Information Systems, Software Quality Management and Project Management. She is engaged in research in Information Systems Engineering, Multidisciplinary Approaches of Software Engineering, Knowledge Management and Software Quality Management. She has a particular interest in human and cultural approaches. She has an extensive industrial experience (since 1975) in software development on different levels from many European countries and mainly from multinational organisations. She has a number of academic and industrial project partners in many countries and she has published around 50 papers related to her research.

Errikos Siakas

Errikos Siakas has graduated in 2003 as a Software Engineer from the department of Informatics at the Alexander Technological Educational Institute of Thessaloniki. He is teaching computing to children in the secondary schools and simultaneously he is engaged in research in Multidisciplinary Information Systems Engineering. His particular research interest is in Agile Software Development and Global Outsourcing.



# Managing Multi-Cultural and Multi-Social Projects in SPI

Miklos Biro<sup>1</sup>, Richard MESSNARZ<sup>2</sup>, Janos Ivanyos<sup>3</sup>

<sup>1</sup>Corvinus University of Budapest, Hungary

Email: [mbiro@informatika.bke.hu](mailto:mbiro@informatika.bke.hu)

<sup>2</sup>ISCN GesmbH, Schieszstattgasse 4, A-8010 Graz, Austria

Email: [rmess@iscn.com](mailto:rmess@iscn.com)

<sup>3</sup>Memolux, Budapest, Hungary

Email: [ivanyos@memolux.hu](mailto:ivanyos@memolux.hu)

**Abstract:** This paper discusses the required competencies of EU project managers in general, and outlines that the success of projects by two thirds depends on human and social factors in the organisations when projects are running on a multinational basis.

It will outline some major highlights of the underlying studies and a set of rules promoted for European projects (<http://www.manageur.com>) about how to be successful. Corvinus University, ISCN, and Memolux were partners in these projects and studies which involved many hundred firms from 1998 – 2003.

The paper describes how a two dimensional space model (social + cultural) is currently being used to interpret rules for a better management of project collaboration.

## 1. Background – EU Project Management

In the project ManagEUr ([www.manageur.com](http://www.manageur.com)) the following materials have been developed and approx. 700 managers have been trained in 2005/2006.

- A defined skills set for EU project managers
- An online browsing and self assessment of skills portal.
- A set of best practice team models for managing EU projects.
- A set of course materials with exercises and example projects.

For structuring the skills set the EU Leonardo da Vinci project ManagEUr followed the EU standards for skills descriptions [3] (see Figure 1).

<p>A <b>Domain</b>, contains</p> <ul style="list-style-type: none"><li>• <b>Job Roles</b>, which contain</li><li>• <b>Units</b>, which contain</li><li>• <b>Elements</b>, which contain</li><li>• <b>Performance Criteria</b>, which must be proven by</li><li>• <b>Evidences</b>.</li></ul>
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Figure 1: Basic elements of the skills definition model

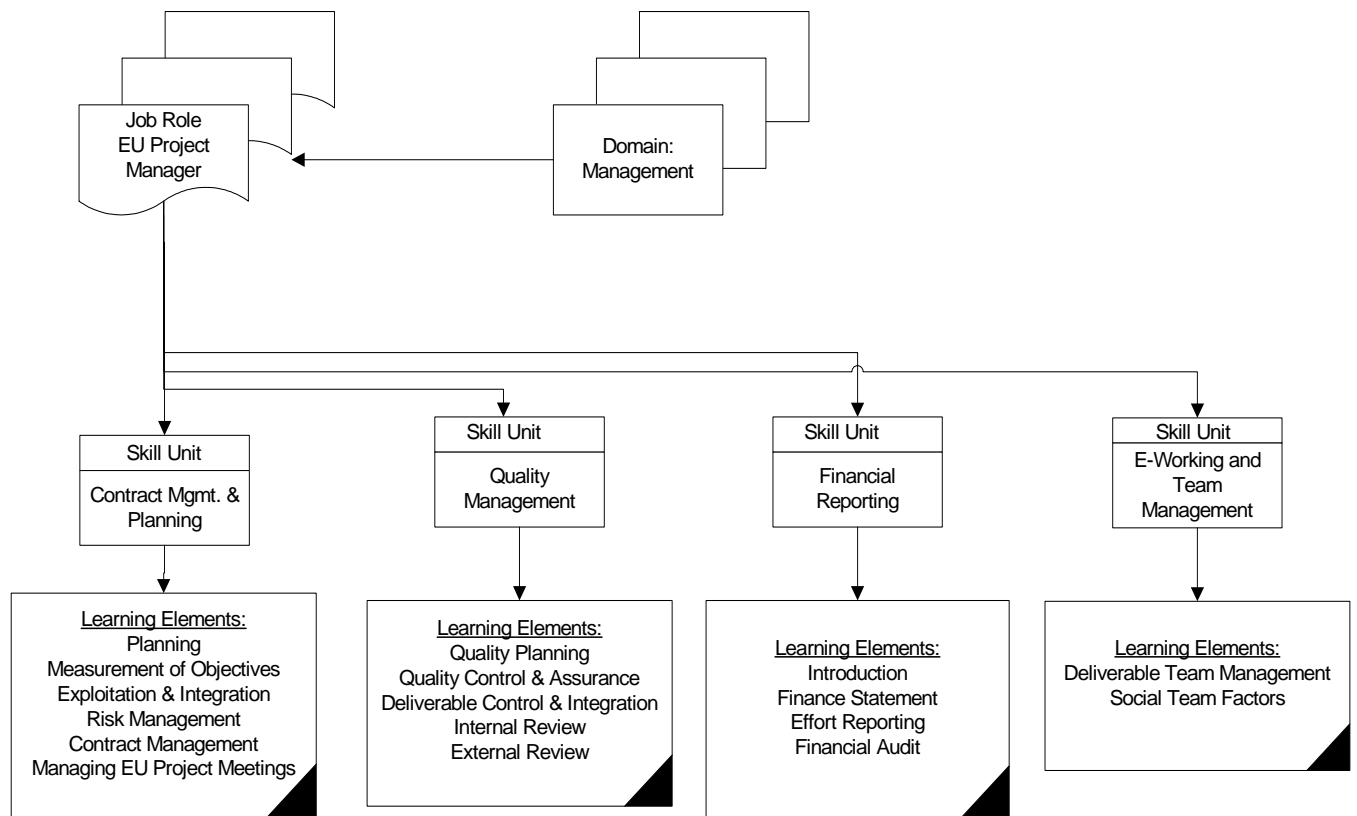


Figure 2: Skills Card of an EU Project Manager Job Role

**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 [9] 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.

For each learning element there is course module with exercises and reference projects. The skill unit “E-Working and Team management” is like a supporting set of competencies for all other competencies and thus has a large impact on the successful implementation.

## 2. Importance of Team Skills - Underlying Studies

Improvement in general relates to successfully changing and adapting products, services, and processes in a continuous cycle. Software Process Improvement can be seen as the process related innovation cycle.

The BESTREGIT study (BEST Regional Innovation Transfer white paper, 1998) analysed 200 innovative firms in Europe and highlighted at this time the major success principles of innovative firms (see Figure 1). The study outlines that innovative companies are capable of creating a multi-cultural network of creative minds (supported by networking technologies) and can focus their efforts in areas where they are better than any competitors.

The TEAMWORK study (TEAMWORK project, 2001 – 2003) [2], [3], [4] used a platform to provide the networking infrastructure (as required according to the 1998 study) and analysed 44 networked projects in 59 participating European organisations. A social analysis team (experts who usually do social patterns analysis for national elections) performed interviews and feedback analysis to extract the typical social barriers and potential success factors to run multinational joint European projects.

- **Study of 50 innovative regions in the EU with 200 innovative companies (1998)**
  - by Dr. Sean MacCarthy.
  - How the innovative organizations operate?
  - How the individuals in the organizations respond to continuous change?
  - Main findings: Innovative Organisations ...
    - invest time and money into the understanding of the fundamentals of the forces of change
    - understand the different cultures through personal contacts using networking at a personal level
    - study trends and they are always up-to-date
    - concentrate their energy in areas where they excel or where no one else can operate
    - outsource all other non-core activities
    - are practical users of information technologies and have information technology strategies in place

**Figure 3: Core Competencies of Successful Innovative Firms (1998)**

The TEAMWORK study (2001 – 2003) [2] created an environment as proposed by the first study and observed the typical patterns of behaviour of experts and teams in European projects. It defined a so called four social dimensions which interfere with the success of projects.

1. Core Organisational Patterns
2. Organisation Growth Patterns
3. Organisational Style Patterns
4. People and Behaviour Codes

These 4 dimensions (even if the environment according to the BESTREGIT study had been set up) largely influenced the success of teams. The study used statistical techniques to only propose those patterns where the typical pattern across all teams showed a high statistical

significance (can be predicted with a high likelihood to occur). The different social success principles are discussed in chapter 3 below.

The multinationals study (128 multinational companies, 2002) [7] analysed multinational companies and their approach to innovation. The result is displaying again the importance of the social dimension. (see Figures 4 and 5). The study clearly showed that also in large companies nearly 60% of the success of innovation (including SPI based process innovation) relies on organisational and social success patterns.

### • Motivation for Innovation

• Technological developments	31%
• Capacity in the industry	15%
• Pressure from customers	18%
• New entrants to the market	12%
• Price sensitive market	10%
• Regulatory changes	4%
• Shorter product life cycle	2%
• Don't knows	8%

Figure 4: Areas where large multinational companies invest into projects

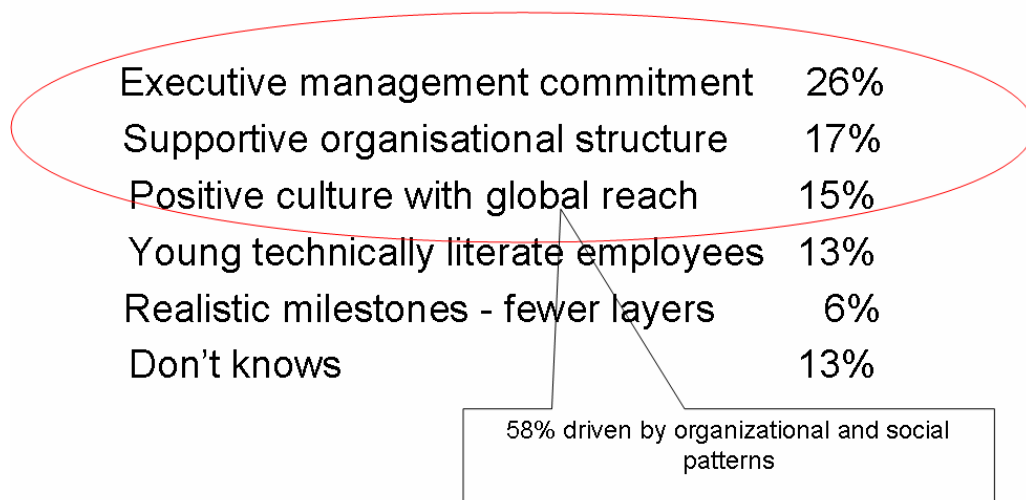


Figure 5: Success factors in large multinational companies works

### 3. The Social Dimension Architecture

The social study [2], [4], [5], [6], [7] outlined the following success principles as a typical team pattern in EU projects.

#### 3.1 Core Organisational Patterns

The following core organisational patterns showed a high significance in the TEAMWORK study in 2001 – 2003:

Social Pattern	⊗	Projects were more likely to successfully achieve the goals if ...
Community of Trust	○	<ul style="list-style-type: none"> <li>The teams had a joint vision, no conflicts of interest, and a team-learning behaviour.</li> </ul>
Stability by Baselines	○	<ul style="list-style-type: none"> <li>The teams had agreed baselines of work as intermediary results to check that all have the same basis of knowledge to build upon.</li> </ul>
Empowerment of Productive Roles	○	<ul style="list-style-type: none"> <li>The teams allowed that young and productive (without power conflict) team members were empowered and even allowed that they become the center of information flows (the project manager not controlling but becoming an empowerment and knowledge coordinator).</li> </ul>
Skill Based Task Distribution	○	<ul style="list-style-type: none"> <li>The teams were formed on the basis of complementary skills sets (skill analysis at start).</li> </ul>

**Table 1: Core Organisational Patterns**

#### 3.2 Organisation Growth Patterns

The following organisational growth patterns [2], [4], [5], [6], [7] showed a high significance in the TEAMWORK study in 2001 – 2003:

Social Pattern	⊗	Projects were more likely to successfully achieve the goals if ...
Size of the Organisation	○	<ul style="list-style-type: none"> <li>The teams had a size of 7 – 10 people.</li> </ul>
Engage Customers	○	<ul style="list-style-type: none"> <li>The teams had a direct involvement of the customer to jointly review and validate innovation ideas.</li> </ul>
Surrogate Customer	○	<ul style="list-style-type: none"> <li>In the case of teams having no chance to directly engage a customer they invented a special role internally to play the customer role.</li> </ul>
Use Team Scenarios	○	<ul style="list-style-type: none"> <li>The teams agreed how to collaborate in specific issues (based on scenarios of work) to reduce any communication conflicts.</li> </ul>
Firewalls	○	<ul style="list-style-type: none"> <li>The teams had a social firewall manager who in</li> </ul>



Social Pattern	⊗	Projects were more likely to successfully achieve the goals if ...
		a critical development phase kept the researchers from being interrupted by extraneous influences and special interest groups.
Unity of Purpose	○	<ul style="list-style-type: none"> <li>The teams had a common defined purpose and mission.</li> </ul>
Roles Assignment by Expertise	○	<ul style="list-style-type: none"> <li>The assignment of duties was based on the competencies of the team members (not just by power hierarchies).</li> </ul>
Engage Quality Assurance	○	<ul style="list-style-type: none"> <li>The teams had internal and external reviews in periodical cycles.</li> </ul>
Engage Group Validation	○	<ul style="list-style-type: none"> <li>The teams involved in periodical cycles all stake holders to group validate the team results.</li> </ul>

Table 2: Organisational Growth Patterns

### 3.3 Organisational Style Patterns

The following organisational style patterns [2], [4], [5], [6], [7] showed a high significance in the European project's study in 2001 – 2003:

Social Pattern	⊗	Projects were more likely to successfully achieve the goals if ...
Few Roles	○	<ul style="list-style-type: none"> <li>Organisations identified a max. of 16 different roles per organisational unit.</li> </ul>
Producers in the Middle	○	<ul style="list-style-type: none"> <li>Organisations empowered the productive roles in the team and shifted work to productive roles. (assignment by productivity and not just by hierarchy)</li> </ul>
Stable Roles	○	<ul style="list-style-type: none"> <li>Organisations had accepted the role based team structures and agreed that certain roles must have a stable assignment of a person to the role (not changing too often).</li> </ul>
Divide and Conquer	○	<ul style="list-style-type: none"> <li>Organisations had a principle to divide big tasks into teams which collaborate.</li> </ul>
Result Focus	○	<ul style="list-style-type: none"> <li>Organisations had to deal with labour forces representing different political, religious, etc. opinions and they could create a mission independent from politics and religion.</li> </ul>
Face to Face	○	<ul style="list-style-type: none"> <li>Organisations understood that beside the e-team structures still a social and face to face contact is needed.</li> </ul>
Deloading central Roles	○	<ul style="list-style-type: none"> <li>Organisations supported role based team models and added further persons to a role (if a role is overloaded , role and not person).</li> </ul>
Decoupling Stages	○	<ul style="list-style-type: none"> <li>Organisations had a planning capacity to analyse the dependencies between teams and a process to</li> </ul>

Social Pattern	⊗	Projects were more likely to successfully achieve the goals if ...
		decouple phases and stages.

**Table 3: Organisational Style Patterns**

*3.4 People and Behaviour Codes*

The following people patterns [2], [4], [5], [6], [7] showed a high significance in the TEAMWORK study in 2001 – 2003:

Social Pattern	⊗	Projects were more likely to successfully achieve the goals if ...
Architect Controls the Product	○	<ul style="list-style-type: none"> <li>Teams had a creative mind who had an overview about the whole project driving the team.</li> </ul>
Architecture Team	○	<ul style="list-style-type: none"> <li>In case of a very complex project a well selected set of creative minds had a joint overview about the whole.</li> </ul>
Smoke Filled Room	○	<ul style="list-style-type: none"> <li>Teams organised creative workshops to elaborate different solution scenarios in a “smoke filled room” environment.</li> </ul>
Social Drive	○	<ul style="list-style-type: none"> <li>All members of the team were committed to the mission and the project leader continuously illustrated who has or has not delivered so far (outlining who belongs to the team and the team status).</li> </ul>
Standards Linking Locations	○	<ul style="list-style-type: none"> <li>In case many involved locations the teams agreed standards to exchange materials and knowledge.</li> </ul>

**Table 4: People and Behaviour Codes**

*3.5 Summary*

The study involved teams from 7 European countries. Of course, the study cannot claim to be totally independent from the culture dimension, nor it can conclude that if 7 European countries showed these typical patterns then it will be applicable for all European countries. However, what makes the study interesting is that it was done by non-technicians and social science experts (an observation group totally independent from the technical group) and it used a social science patterns analysis which based on statistical significance. This means that the above social dimension factors have been identified with a statistical 95% probability (remaining 5% error probability) across the European teams.

#### 4. Short Introduction to the Cultural Dimension

In this chapter we presume that basics of the Hofstede study are known, because many papers in EuroSPI have already published about these dimensions (see Figures 6 and 7) [9], [10], [11], [12]. In this paper we select the special case of collaborating with Russian teams as they increasingly become involved in the projects of ISCN and because EuroSPI 2006 takes place near the Russian border. The same approach can be followed using any of the other national results from Hofstede.

ISCN's development department is in Graz, Austria, so that we basically compared the German with the Russian features. Cooperations in different EU and related projects exist now since 2001.

Figure 6 highlights that Russians have a 50% higher uncertainty avoidance than a typical German culture would have. This meant that while others in our projects (Spanish, Irish, etc.) felt that the Austrian project leader imposed a lot of rules, the Russians (even after receiving detailed plans) were asking ISCN to be more precise and define more detailed tasks for them.



Figure 6: Uncertainty Avoidance

Figure 7 highlights that Russians have only one third of long term orientation than a typical German culture would have. This meant in the projects that that while the German teams defined long term release plans the Russians were very detailed in their technical knowledge and wanted to change more often (after each potential technical improvement occurred).

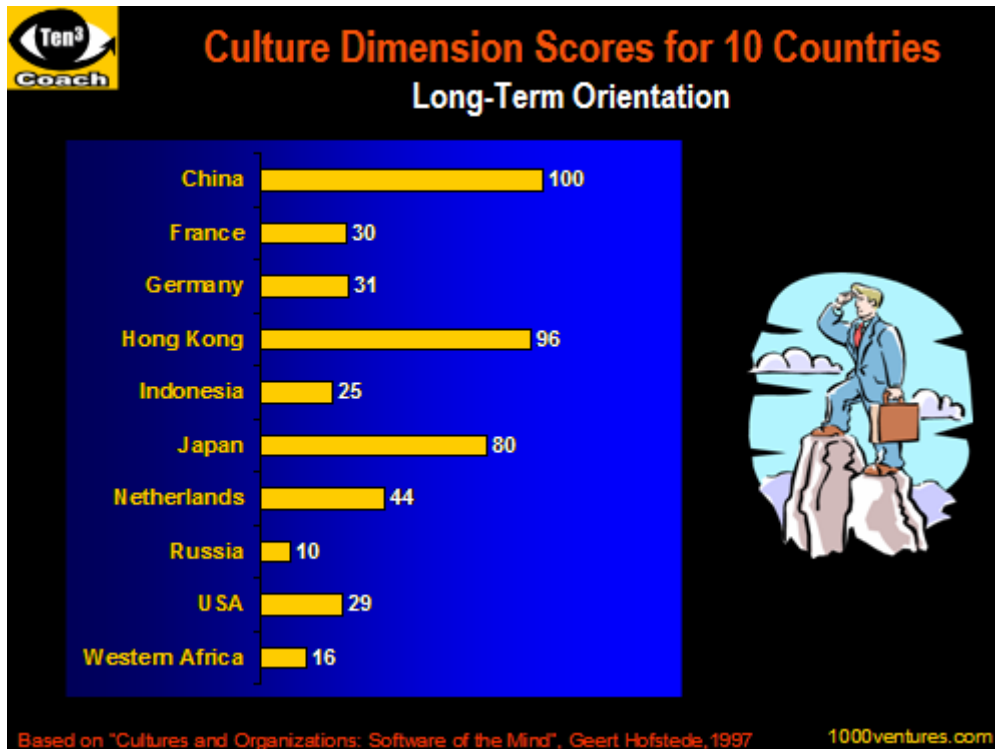


Figure 7: Long Term Orientation

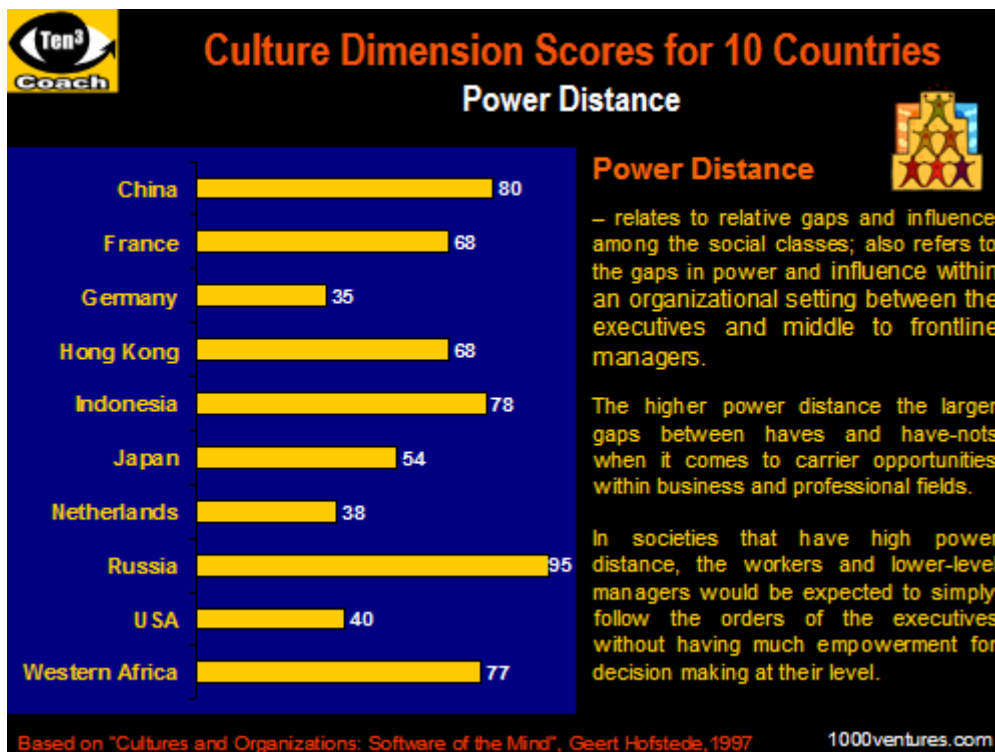


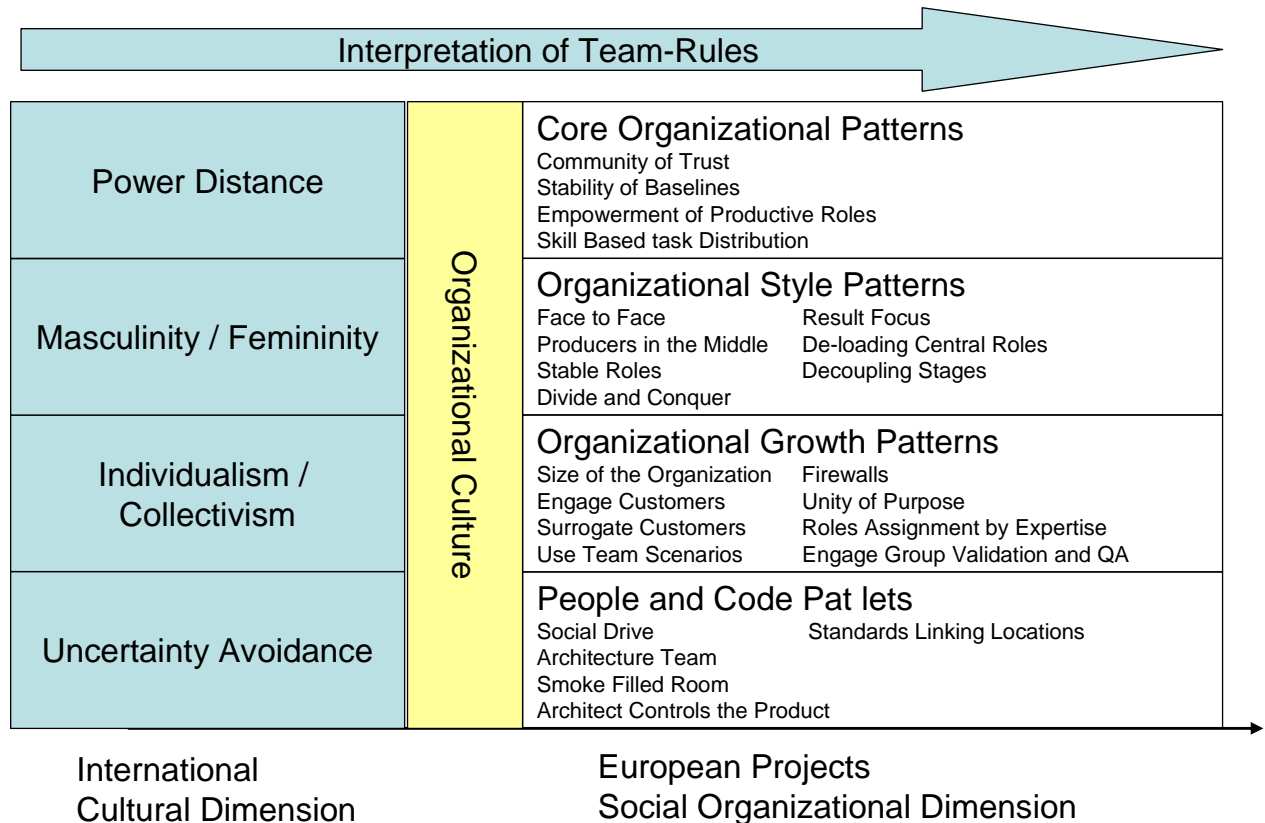
Figure 8: Power Distance

Figure 8 highlights that Russians have a three times higher power distance than a typical German culture would have. This meant in the projects we needed to agree with very high

level people (head of departments in Russian academy of sciences) first before the researchers were actually starting to support the project (even if they were motivated before they strictly waited for higher support to do so).

### 5. The Two Dimensional Space (Social + Cultural)

Any cross-national collaboration is typically a mixture of both dimensions, the international cultural dimension and the social organisational dimension (see Figure 9).



**Figure 9: The Two-Dimensional Space**

In the years 2003 – 2006 we elaborated further on the guidelines of the previous TEAMWORK project and in the EU-Manager project we started training managers across Europe to apply the above described model in Figure 9. To illustrate its usage we will continue to elaborate how we applied the model for the Russian collaborations.

To establish a guideline a social and cultural matrix is being established (see Figure 9).

Hofstede Dimension	Russian Collaboration from Austria	European Social Dimension	Impact	Further guidelines for Russian Collaboration from Austria
<b>Power Distance</b>	Russians have a three times higher power distance than a typical German culture would have. This meant in the projects we needed to agree with very high level people (head of departments in Russian academy of sciences) first before the researchers were actually starting to support the project (even if they were motivated before they strictly waited for higher support to do so).			
		<b>Community of Trust</b>	yes	involve a senior Russian person (very high level) in the project mission
		<b>Empowerment of Productive Roles</b>	yes	the senior Russian person should act as the project representative while others (technically very competent researchers) shall be selected as experts to work daily on the joint project
		<b>Skill Based Task Distribution</b>	yes	it is important to assign the selected researchers to tasks due to their strength, while the representative will be included in the project (mission) board; the worst case happened when the senior person becomes the project expert and later talks to his experts (losing information and productivity)
<b>Uncertainty Avoidance</b>	Russians have a 50% higher uncertainty avoidance than a typical German culture would have. This meant that while others in our projects (Spanish, Irish, etc.) felt that the Austrian project leader imposed a lot of rules, the Russians (even after receiving detailed plans) were asking ISCN to be more precise and define more detailed tasks for them.			
		<b>Use Team Scenarios</b>	yes	Define 50% more detailed tasks for roles to get them working with you
		<b>Standards Linking Locations</b>	yes	Define 50% more detailed content guidelines and checklists for expected results to get them working with you
		<b>Engage Group Validation and QA</b>	yes	Define 50% more quality guidelines and criteria on more detailed level to get them working for you
		<b>Roles Assignment by Expertise</b>	yes	They have proven to be very high experts on a detailed technical level in the projects. However, due to the many rules they expect to be able to work, they usually do not become the producer in the middle (will change over the years).
...				

Figure 9: Social – Cultural Interpretation Matrix

### 6. Conclusions

The European and international studies (1998 – 200 firms, 2002 – 128 multinational firms, 2003 – 59 European organisations) illustrate that innovation and SPI project's success depends to 1/3 on the technical issues and 2/3 on the social cultural issues in the organisation and involved cultures.

SPI is a quite significant investment and reaching an ROI takes time (years). Organisations working in a distributed sense thus need guidelines about how to deal with cultural and social issues in teams.



**Figure 10: The EU Project Manager Team**

In 2005/2006 more than 700 managers have been trained. Above 500 performed a test and received certificates. EU Manager will join the EU Certificates Association at a founder conference in December 2006 creating the basis for Europe wide certification.

A valorisation conference for EU manager takes place on 2.-4.10.2006 in Budapest Hungary, approx. 100 managers are expected.

The goal is to achieve nearly 1500 trained EU managers within the next 2 years.

### References

- [1] M. Biro, R. Messnarz, A. Davison (2002) The Impact of National Cultures on the Effectiveness of Improvement methods - The Third Dimension, in Software Quality Professional, Volume Four, Issue Four, American Society for Quality, September 2002
- [2] Feuer E., Messnarz R., Wittenbrink H., Experiences With Managing Social Patterns in Defined Distributed Working Processes, in: Proceedings of the EuroSPI 2003 Conference, 10-12 December 2003, FTI Verlag, ISBN 3-901351-84-1

- [3] 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
- [4] 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
- [5] Messnarz R., R. V. Horvat, K. Harej, E. feuer, ORGANIC - Continuous Organisational Learning in Innovation and Companies , in: Proceedings of the E2004 Conference in Vienna, E-Work and E-commerce, eds. Brian Stanford Smith, Enrica Chiozza, IOS Press, Amsterdam, Berlin, Oxford, Tokyo, Wash-ington, 2004
- [6] Messnarz R., G. O'Suilleabhain, R. Coughlan, From Process Improvement to Learning Organisations, in: Wiley Sodftware Process Improvement and Practice – Selected papers from EuroSPI 2005, May, 2006
- [7] O'Keeffe, T., & D. Harrington, 2001. Learning to Learn: An Examination of Organisational Learning in Selected Irish Multinationals. Journal of European Industrial Training, MCB University Press, Vol. 25: Number 2/3/4
- [8] Gemünden H.G., T. Ritter, Inter-organisational Relationships and Networks, Journal of Business Research, 2001
- [9] Siakas Kerstin V, Balstrup Bo, Georgidaou Elli, Berki Eleni. 2005. Global Software Development; the Dimension of Culture, IADIS.International Association for Development of the Information Society) International Virtual Multi Conference on Computer Science and Information Systems (MCCSIS 2005) - SEA.Software Engineering and Applications, 26 April 2005
- [10] Siakas Kerstin V, Berki Eleni, Georgiadou Elli. 2003. CODE for SQM: A Model for Cultural and Organisational Diversity Evaluation, EuroSPI 2003.European Software Process Improvement Conference), Graz, Austria, 10-12. 12. 2003, pp. IX 1-11
- [11] Siakas Kerstin V. 2002. SQM-CODE: Software Quality Management – Cultural and Organisational Diversity Evaluation, PhD Thesis, London Metropolitan University, UK
- [12] 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 November



Hofstede's Dimension of Culture Scales					
Country	Power Distance	Individualism	Uncertainty Avoidance	Masculinity	Long term orientation
Arab countries	80	38	68	53	
Argentina	49	46	86	56	
Australia	36	90	51	61	31
Austria	11	55	70	79	
Belgium	65	75	94	54	
Brazil	69	38	76	49	65
Canada	39	80	48	52	23
Chile	63	23	86	28	
China, Mainland					118
Colombia	67	13	80	64	
Costa Rica	35	15	86	21	
Denmark	18	74	23	16	
East Africa	64	27	52	41	
Equador	78	8	67	63	
Finland	33	63	59	26	
France	68	71	86	43	
Germany FR	35	67	65	66	31
Great Britain	35	89	35	66	25
Greece	60	35	112	57	
Guatemala	95	6	101	37	
Hong Kong	68	25	29	57	96
India	77	48	40	56	61
Indonesia	78	14	48	46	
Iran	58	41	59	43	
Ireland	28	70	35	68	
Israel	13	54	81	47	
Italy	50	76	75	70	
Jamaica	45	39	13	68	
Japan	54	46	92	95	80
Malaysia	104	26	36	50	
Mexico	81	30	82	69	
Netherlands	38	80	53	14	44
New Zealand	22	79	49	58	30
Norway	31	69	50	8	
Pakistan	55	14	70	50	
Panama	95	11	86	44	
Peru	64	16	87	42	
Philippines	94	32	44	64	19
Poland					32
Portugal	63	27	104	31	
Salvador	66	19	94	40	
Singapore	74	20	8	48	48
South Africa	49	65	49	63	

<b>Hofstede's Dimension of Culture Scales</b>					
<b>Country</b>	<b>Power Distance</b>	<b>Individualism</b>	<b>Uncertainty Avoidance</b>	<b>Masculinity</b>	<b>Long term orientation</b>
South Korea	60	18	85	39	75
Spain	57	51	86	42	
Sweden	31	71	29	5	33
Switzerland	34	68	58	70	
Taiwan	58	17	69	45	87
Thailand	64	20	64	34	56
Turkey	66	37	85	45	
Uruguay	61	36	100	38	
USA	40	91	46	62	29
Venezuela	81	12	76	73	
West Africa	77	20	54	46	16



# Using Multi Dimensional Scaling to analyse Software Engineers' De-motivators for SPI

Nathan Baddoo<sup>1</sup>, Tracy Hall<sup>1</sup> and Ciaran O'Keeffe<sup>2</sup>

<sup>1</sup>School of Computer Science, University of Hertfordshire, UK

<sup>2</sup>Department of Psychology, Liverpool Hope University, UK

## Abstract

We present analysis of factors that de-motivate software practitioners from supporting Software Process Improvement (SPI) initiatives, using a Multidimensional Scaling (MDS) technique. Our findings are based on an empirical study involving nearly two hundred software practitioners from thirteen UK companies. Our aim is to provide managers with a better understanding of the obstacles that prevent practitioners from supporting SPI. This insight should help SPI managers establish more effective SPI implementation strategies. In this paper we re-introduce the use of Multidimensional Scaling (MDS) techniques in SPI research. MDS is a social science data analysis technique designed to generate a rich visual understanding of human issues. By using MDS we found evidence to suggest distinct clusters of de-motivators exist for different staff groups.

**Keywords:** Software process improvement, practitioner, de-motivators, MDS, SSA

## 1. Introduction

In this paper we report on the use of a Multidimensional Scaling (MDS) technique [Shye *et al.*, 1994] to analyse data on software practitioners' de-motivators for Software Process Improvement (SPI). This data was collected from focus groups in thirteen UK companies encompassing almost two hundred software practitioners. We looked separately at the de-motivators of developers, project managers and senior managers. We first reported the original findings of this study in 2003 [Baddoo and Hall 2003]. In this paper we report further analysis carried out on the original findings using MDS.

The aim of this extended analysis is to understand the relationship between software practitioners' de-motivators for SPI by showing how these de-motivators co-occur to each other. We use MDS to represent this co-occurrence. MDS is a research technique used increasingly by psychologists to uncover such attitudinal co-occurrence. Our results should help SPI managers develop a richer understanding of the association between the de-motivators of SPI for different staff groups of software practitioners. This improved understanding should enable SPI managers to implement SPI more effectively.

## Session 5: SPI and Learning Factors

Another aim is to replicate a demonstration, carried out in 2002 [Baddoo and Hall 2002], of how MDS can be applied in software engineering to achieve a richer understanding of some of the human factors of this discipline. We believe that such replication gives the analysis method efficacy.

In Section Two we introduce Multidimensional Scaling. In Section Three we describe our research methods. Section Four presents results and discussions from the plots of practitioner de-motivators for SPI. In Section Five we conclude this paper.

### 2. Multidimensional Scaling

Multidimensional Scaling is a multivariate analysis procedure that represents, geometrically, the relationship between variables within an Euclidean space. It is defined as:

A set of procedures that allow for information contained in a set of data to be represented by a set of points in space, arranged in such a way that the distances between the points reflect the empirical relationship. [Guttman, 1968].

Traditionally, MDS has been used in conjunction with facet theory [Guttman, 1968]. Facet theory is an empirical research design and analysis technique that Guttman and Greenbaum [1998] describe as:

1. Having a definitional framework for a universe of observations in a study.
2. Establishing an empirical structure of observations within this framework.
3. Searching for correspondence between the definition framework and the empirical structure.

Facet theory proposes ways of exploring and understanding a concept, for example attitude, intelligence or de-motivators in this instance, by depicting the inter-relationship between the elements of that concept [Shye *et al.*, 1994]. This is achieved through the empirical observation of many variables of this concept. Multidimensional scaling is a procedure for representing and analysing the observed empirical elements of a facet.

In this particular study the *concept* is de-motivation for SPI. The *facets* are the classification of de-motivation according to whether they are personal, organizational or process de-motivators. Our *variables* (i.e. *elements*) are the de-motivators for SPI as cited by three groups of software practitioners [Baddoo & Hall 2003].

#### 2.1 Studies using MDS

Meads' review of the development of MDS indicates that most initial use of the technique was by scientists working in psychophysics and sensory analysis [Mead, 1992]. Today, MDS is predominately being used in investigative and occupational psychology. For example, Donald [1994] used MDS to explore the structure of office workers' experience of work environments. Donald's study used the non-parametric MDS technique of Smallest Space Analysis (SSA) to analyse attitudinal data collected from 215 office workers. Donald's findings showed that in addition to functional aspects of the office environment other aspects of the environment were equally important, for example, group cohesion and a *spirit de corps* [Donald, 1994].

Other studies done with MDS in the UK have been in the field of crime research. For example, in interviews with sexual offenders, the motivations, excuses, reasons and justification given by the offenders for their actions form the variables of the analysis. Researchers then try to ascertain how these variables co-occur with each other in multi-dimensions and establish a criterion for grouping them together [e.g. Alison *et al.*, 1998; Canter & Alison, 1999; Canter & Heritage, 1990]. Indeed, in Canter and Heritage’s model of sexual offence behaviour, they encapsulate the behavioural structure of sexual assaults by presenting the different facets of the concept as deduced from a sample of various cases and offenders. One important advantage of MDS is its accessibility to empirical scrutiny due to the concentration on the overt behavioural constituents of rape, especially in comparison with other approaches that partly rely on motivational and/or psychodynamic inferences [e.g. Groth, 1979].

The use of MDS in software engineering research is rare. In 2002, we used MDS to analyse the association between motivators of SPI [Baddoo and Hall 2002]. Since then, few other studies have applied MDS in similar software engineering research, for example [Young 2004]. We continue this trend by analysing de-motivators to SPI with MDS in this study.

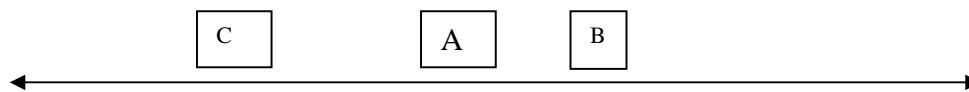
**2.2 An overview of MDS**

A basic tenet of MDS is that the stronger the statistical relationship between variables, the closer they will appear in geometric space. Table 1 shows the statistical relationship between three variables, A, B and C.

	A	B	C
A	*	.8	.5
B		*	.1
C			*

**Table 1. Multi-variate correlations: 3 variables [Baddoo and Hall 2002]**

Table 1 shows that variables A and B have the highest correlation (.8). Variables A and C show a weaker correlation than between A and B but a stronger correlation than between B and C. These correlations can be represented in one-dimensional space as in Figure 1.



**Figure 1. One dimensional representation of statistical relationship [Baddoo and Hall 2002]**

The limitation of this one-dimensional representation becomes apparent if Table 1 is expanded to include a fourth variable. Table 2 shows a four variable correlation matrix. To represent the relationship between all four variables as distances will be impossible to do on a one-dimensional scale. This is because it is impossible to place D anywhere in Figure 1 that truly reflects its statistical relationship with the other three variables. Representing the statistical relationships will also be difficult in two dimensions, i.e. in an area. However, conceptualising the space in which these variables occur in three dimensions (within a cube) makes it possible to adequately present this statistical relationship.

	A	B	C	D
A	*	.8	.5	.3

B	*	.1	.6
C		*	.4
D			*

Table 2. Multi-variate correlations: 4 variables

Multidimensional scaling comes from a need to represent multiple variables that are related to one another. As the examples in Tables 1 and 2 show, when there are more than three variables, this representation becomes difficult to show either on one or two-dimensional scales. A three dimensional scale, therefore, becomes the best option of representing the statistical relation between several variables.

### 2.3 Smallest Space Analysis (SSA)

SSA is one of the three MDS techniques recommended by Guttman [1968]. It is a non-parametric MDS technique, used for cases where the data being analysed is nominal or categorised ordinal data. The greater the similarity between two variables, the greater their proximity in the corresponding geometric space.

#### 2.3.1 Overview of SSA

The technique is termed *smallest* space analysis because it uses the rank order of the correlation coefficients to plot the closeness of points in geometric space. This ensures that the smallest space possible is used in the depiction of relationships.

Traditionally, correlation between quantitative variables is measured by the degrees of change over cases. Correlation is measured in non-parametric data by the presence or absence of variables over a series of cases.

The following is a summary of the steps involved in SSA [Guttman 1968]:

- 1 Establish a Content Category Dictionary of all variables.
- 2 Create a data matrix based upon how many occurrences of the variables in the content dictionary appear in each 'case'.
- 3 Run the SSA tool to plot the variables and their relationships.
- 4 Partition the SSA spatial plot into regions. The researcher should be able to argue the inclusion of variables in the region they appear.

When a SSA spatial plot is produced, a coefficient of alienation (commonly Guttman-Lingoes or Jaccard's) is used to measure the "fit" between the geometric representation and the original correlation matrix. A coefficient of 0 represents a perfect fit and anything up to 0.25 is considered good.

#### 2.3.2 Applying SSA in this study

The partitioning of SSA's is done in relation to particular "facets". The regions in a SSA plot are the components (i.e. the facets) of the concept under investigation. Partitioning takes place according to how the variables represent themselves in the plot. When partitioning a SSA, the researcher is trying to show how components of a concept are mapped out on the plot and how variables are represented within the components. Established theory defines the components of

each concept and the variables within that component. It should therefore be possible to defend the inclusion of any variable in a particular region/component/facet of a plot with theory.

In this study we have chosen to classify de-motivators to SPI into the facets of *personal*, *process* or *organisational*. This choice is discretionary, but the partitioning should be done such that the inclusion of every variable in a particular region is supported by theory suggesting that the variable can be so defined.

### 3. Methodology

#### 3.1 Data collection - focus groups

We used focus groups as our main approach to collecting data on de-motivators for SPI. Focus groups are a tried and tested technique in the social sciences. They involve assembling small groups of peers to discuss particular topics. Discussion is largely free flowing, but is discreetly orchestrated by a researcher. Focus groups have the advantage of being a flexible but sophisticated research technique that allows soft issues to be explored. Conventional software engineering research methods are usually ineffective at collecting the rich data that focus groups generate. Indeed focus groups have been described as "*a way to better understand how people feel and think about an issue*" [Krueger & Casey, 2000]. Also, Morgan and Krueger say that "*the comparisons participants make among each other's experiences and opinions are a valuable source of insights into complex behaviours and motivations*" [Morgan & Krueger, 1993]. Focus groups are, therefore, an ideal vehicle for exploring motivation for SPI.

Focus groups also elicit data that allows a better understanding of the differences between groups of people [Krueger & Casey, 2000]. This makes them ideal in our study where we are interested in exploring the different de-motivators for SPI of practitioner groups.

From September 1999 to March 2000 we visited 13 software companies and conducted 49 focus groups. The groups comprised between 4 to 6 members and altogether 200 software practitioners took part in this study. Participating companies were selected from a larger sample of companies who responded to a detailed questionnaire giving broad information about their software development activities and company demographics. They were chosen to provide our research project with a cross-section of company maturity levels and software applications.

Each focus group lasted approximately 90 minutes. Based on our previous experiences of using focus groups [Hall & Wilson, 1997], we separated senior managers, project managers and developers into separate focus groups. Kitson and Masters in their 1993 study of software process assessment in 59 companies collected data from similar staff groups [Kitson & Masters, 1993]. They report that because they were collecting data from managers and practitioners who were confronting real issues on a daily basis, they had high confidence in the accuracy and validity of the data. In total, we conducted the following focus groups: 21 developer focus groups, 16 project manager focus groups, and 12 senior manager focus groups.

Two of the questions we asked focus groups to discuss were:

- What are the de-motivators to SPI in your company?
- What would stop you from supporting SPI?



## Session 5: SPI and Learning Factors

Practitioner responses to the questions were audio recorded and subsequently transcribed. The initial analysis of the data collected from this study have been reported in [Baddoo and Hall, 2003]. This paper represents the secondary analysis of that data, using the MDS technique of SSA.

### 3.2 Data analysis – SSA

We followed the SSA steps outlined in Section 2.3.1 and established a Content Category Dictionary of all SPI de-motivator variables. The Content Category Dictionary is a collation of the significant factors identified by software practitioners as de-motivators to SPI, across the three practitioner groups. An inter-rater reliability was established for this dictionary by choosing 4 random scripts. Table 3 shows low to moderate kappa statistics for these randomly chosen scripts. Table 3 also shows that two of these statistics are not significant. The Content Category Dictionary is provided as Appendix A. We used this dictionary to produce data matrices for each of the 3 practitioner groups. The matrices are provided in Appendix B. The SSA tool used these matrices to calculate multivariate correlations between de-motivators. Finally, the SSA tool used the correlations to plot the Euclidean distances between de-motivators.

De-motivator Scripts	kappa	Sig.
A	.307	.096
B	.216	.241
C	.525	.001
D	.439	.016

**Table 3:** Overview of the kappa statistics and significance results

### 3.3 Study limitations

We note the following issues in our study methods:

- The data we have collected characterises practitioners' perceptions of factors that de-motivate them from supporting SPI. We have not verified these perceptions directly. It may be that what practitioners believe de-motivates them is not what actually de-motivates them. Furthermore, what people say de-motivates them may not be accurate.
- The data we collected does not show the strength of feeling about a de-motivator within individual focus groups. If a de-motivator came up in a group it was recorded. So one person may have cited a de-motivator and whether everyone agreed or no one else agreed it was recorded. We have measured the importance of a de-motivator by it occurring in multiple groups.
- Although nearly 200 practitioners were involved in this study our data collection points were the 49 focus groups. This means that our data set is relatively small. We chose not to scale up the data points to reflect participants rather than focus groups, but this means that it is sometimes difficult to generate significant relationships with such a relatively small data set.

## 4. Relationship between de-motivators

The following are SSA plots showing the relationship between de-motivators cited by developers, project managers and senior managers.

- SSA: Developers de-motivators

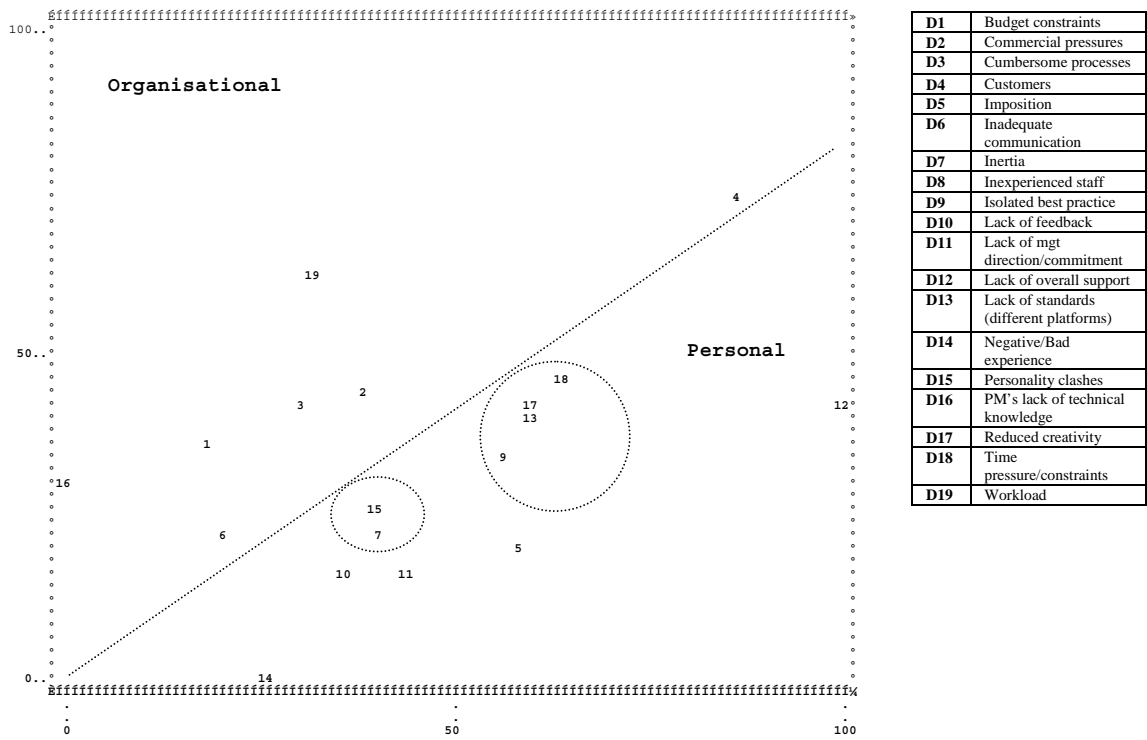


Figure 2: SSA of developers' de-motivators for SPI

## Session 5: SPI and Learning Factors

Figure 2 presents a SSA plot of developers de-motivators for SPI. Coefficient of alienation is 0.07245, suggesting a very good fit. It shows that even though there is a loose association between a majority of the de-motivators, depicted by a sparse cluster at the centre of the plot, there are a few close associations. In fact, three sets of developer de-motivators appear very tightly close together. Inexperience staff(8) and isolated best practice(9) are a perfect fit - appearing on the same spot. Lack of standards(13) and reduced creativity(17) are also very closely shown together. Finally, inertia(7) and personality clashes(15) seem closely associated.

These results show that the likelihood of developers citing these sets of de-motivators together is high. That is: if developers indicate that they are de-motivated by inexperienced staff, they are highly likely to indicate that they are de-motivated by isolated best practices, too. Also, developers who indicate that reduced creativity de-motivates them from supporting SPI are also likely to indicate that a lack of standards de-motivates their SPI effort. Similarly, developers who report inertia de-motivating for SPI are also highly likely to report personality clashes de-motivating for SPI.

Also, an important finding in this developer SSA is that there appears another cluster of de-motivators that all relate to the pressures and constraints of time. These de-motivators are:

- Time pressures and constraints(18)
- Reduced creativity (17)
- Lack of standards(13)
- Isolated best practice(9)
- Inexperienced staff(8)

This finding indicates that developers are in strong agreement about these de-motivators, which are directly or indirectly related to time as a resource. For example, time pressures may result in using staff for jobs they are not properly trained for. Also, time constraints can frustrate the evolution and application of standards within companies. This is because practitioners are more likely to ignore standards and procedures when under pressure.

However, the likelihood of co-occurrence of most of the de-motivators in the plot is very low because most of the de-motivators appear sparse in the plot. For example, the likelihood of developers citing *inertia* in conjunction with *imposition* is less than that of citing *inertia* in conjunction with *personality clashes*. This likelihood is even less when the comparison is made with *inertia* and *lack of overall support*.

Furthermore Figure 2 also shows that there are some developers' de-motivators that appear far removed from any of the other de-motivators. These de-motivators appear on the outskirts of the plot. These de-motivators are customers(4), lack of overall support(12), negative/bad experience(14), project managers' lack of technical knowledge(16) and workload(19). Their positions in the plot indicate that they have little association with each other and also with the remaining de-motivators that are situated in the middle of the plot.

Overall, it is difficult to establish issues that bind many of the very closely associated de-motivators together. For example, even though Figure 2 shows that inexperienced staff and isolated best practice appear as identical plots, it is difficult to establish a particular issue that binds them together. The only concept that sufficiently partitions de-motivators in Figure 2 is one that categorises de-motivators as either organisational or personal. Hence the partition employed.

• SSA: Project manager's de-motivators

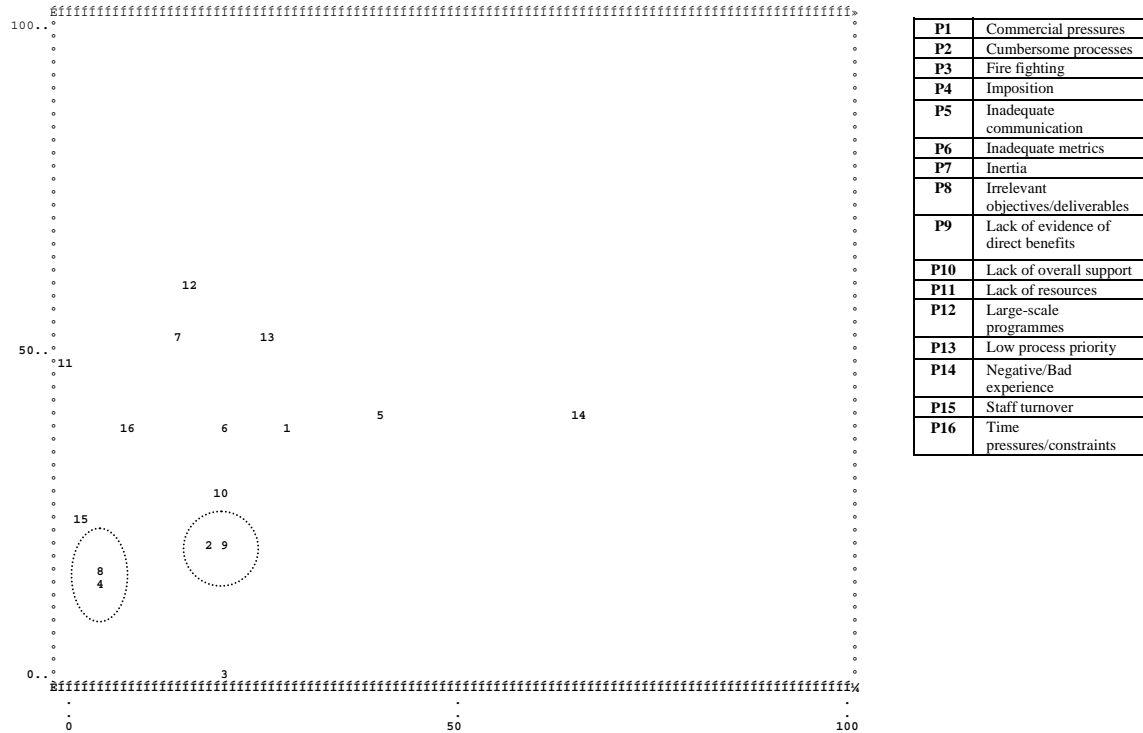


Figure 3: SSA of project managers' de-motivators for SPI

Figure 3 shows a SSA plot of project managers' de-motivators of SPI. This plot has a coefficient of alienation is 0.0662, indicating a very good fit. This figure shows that most of the de-motivators cited by project managers appear in the same region of the plot. With the exception of negative/experience(14) and fire-fighting(3), most de-motivators form a cluster in one region of the plot. This cluster, however, is not a very close one even though some individual de-motivators appear very close to each other. Among these there are imposition(4) and irrelevant objectives(8) and cumbersome processes(2) and lack of evidence(9).

This plot shows that where project managers perceive imposition as de-motivating to their SPI effort, they are also highly likely to cite irrelevant objectives as de-motivating. Likewise, where a lack of evidence de-motivates project managers from supporting SPI, they are highly likely to be de-motivated by cumbersome SPI processes.

The strong association between *irrelevant objectives* and *imposition* suggest that project managers' dissatisfaction with how SPI is implemented is highly likely to be cited together.

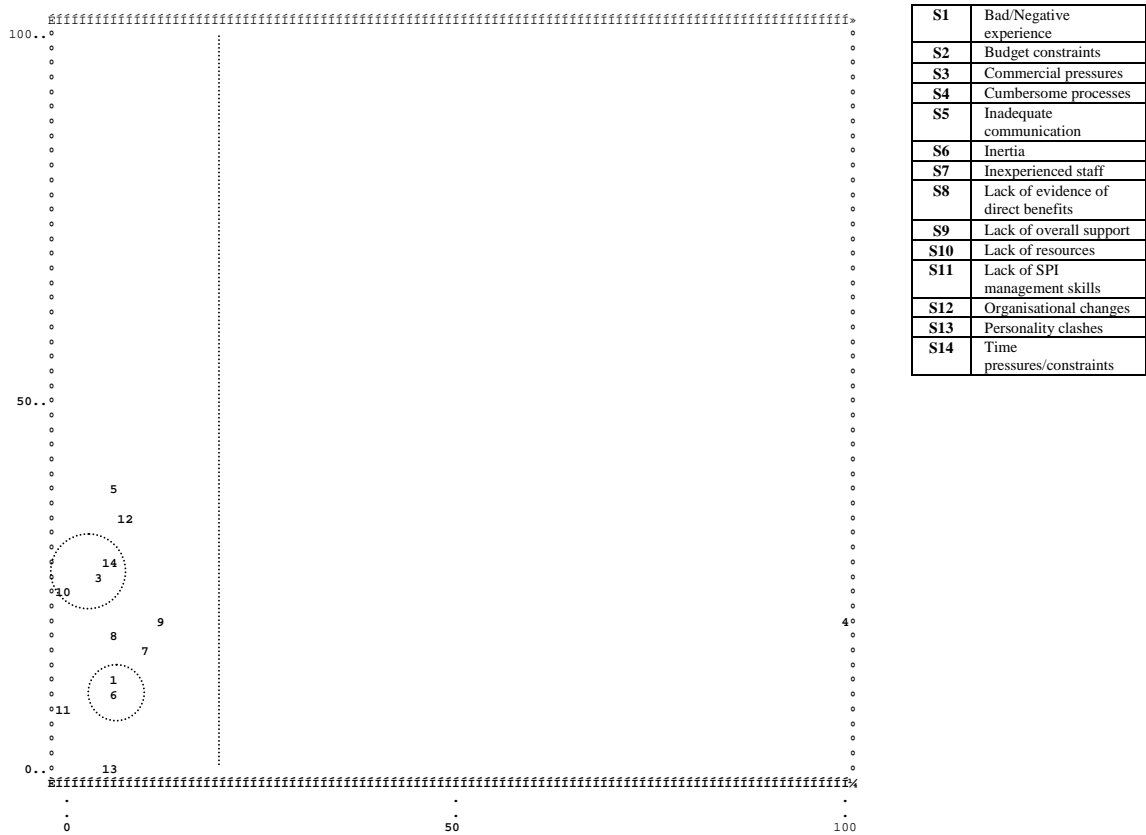
However there appears no direct issue that links *cumbersome processes* and *lack of evidence*. The only possible link is an indirect one related to resistance to SPI. Cumbersome processes may cause dissatisfaction with practitioners and when practitioners are dissatisfied with programmes they stop supporting them. Also, as established in the previous chapter, evidence of SPI success

## Session 5: SPI and Learning Factors

encourages practitioner support, therefore a lack of evidence may make practitioners resistant to SPI.

Overall, the plot of the remaining de-motivators suggests that the likelihood of a strong association between them is comparatively low. The further any two de-motivators appear, the less likely it is that project managers will cite them together. The two least likely de-motivators to be cited by project managers together are fire-fighting(3) and negative/bad experience(14). These de-motivators appear on the fringes of the plot of de-motivators. These two de-motivators also seem the least associated with any of the other de-motivators.

- **SSA: Senior managers' de-motivators**



**Figure 4: SSA of senior managers' de-motivators for SPI**

Figure 4 presents a SSA plot of senior managers' de-motivators for SPI. This plot has a coefficient of alienation of 0.02924, which indicates a very good fit. Figure 4 shows that most of de-motivators appear close to each other in the plot. In fact all the de-motivators, bar one, appear in one region of the SSA plot. Initial indication from this plot is that most of these de-motivators have close associations with each other. Further investigation of Figure 24 shows that within this cluster of de-motivators, some are very close to each other. For example, bad/negative experience (1) and Inertia (6) are very close to each other. Similarly, lack of resources(10), commercial pressures(3) and time pressures/constraints(14) are very close to each other. Budget constraints(2) and inexperienced staff(7) are so much close together that they appear as one plot.

The results from senior managers' SSA indicate that senior managers perceive a strong relationship between their de-motivators for SPI. Even though collectively these de-motivators represent different issues, senior managers' perceptions of them are similar and are very likely to cite them together.

The results of the SSA plot also show that there are subsets of de-motivators related to specific issues that appear even closer to each other.

- First, the issue of resources appears in a tighter cluster of three specific resource related de-motivators. Senior managers indicate that when they are de-motivated by a lack of resources, they are equally likely to be de-motivated by commercial pressures and time constraints. This suggests that resource issues de-motivate senior managers together.
- Secondly, the issue of resistance to SPI is shown by the de-motivators of bad/negative experience of SPI and inertia. The SSA results show that where senior managers identify that bad experience is de-motivating to the SPI effort, they are also most likely to perceive inertia as de-motivating.
- Thirdly, senior managers perceive a near exact closeness between a resource issue and a skills issue. The results show that budget constraints and inexperienced staff both share a common profile. This profile suggests that where senior managers perceive budget constraints as de-motivating to the SPI effort, they are likely also to see inexperienced staff.

Figure 4 also shows that one de-motivator, only, cumbersome processes(4) is far removed from the other de-motivators cited by senior managers. This indicates that this particular de-motivator possesses little in common with the other de-motivators and is unlikely to be cited in tandem with these other de-motivators. We suggest that cumbersome processes does not appear closely related to the other de-motivators cited by senior managers because it is a low level de-motivator whereas the other de-motivators are higher level de-motivators. Also, the other de-motivators cited by senior managers are related to peripheral issues with the processes, whereas this particular de-motivator addresses the processes directly. We suggest that since senior managers tend to abstract their concerns at a higher level than the technical level of the nature of the processes, it becomes improbable that they will be citing cumbersome processes together with the other de-motivators for SPI.

## 5. Conclusion

In this study we have used the MDS technique of Smallest Space Analysis to show that there are clusters of de-motivators for SPI for different practitioner groups, and that these clusters can form under particular themes. For example, some developer de-motivators strongly cluster around the theme of pressures and constraints of time. On the other hand, Project manager de-motivators form a loose cluster, even though it is possible to isolate some of the least coupled de-motivators like *firefighting* and *negative experience* from the rest of the set of de-motivators. Compared to project managers, senior managers generally have a tighter cluster of de-motivators, but then nearly all of these come under high level concerns that senior managers have for processes. The only lower level process concern that senior managers mention as a de-motivator, *cumbersome processes*, appears far removed from any of the other de-motivators.

Using Smallest space analysis in this study, we have explored the inherent assumption of the technique that any underlying structure or common theme in behaviour will be most readily appreciated by examining the relationship each variable has with every other variable. For this reason alone, the utility of an MDS procedure, specifically SSA, in examining de-motivators of SPI has been proven. However, the use of MDS in this research has been as an exploratory approach to analysing categorical data. As a result only few strong conclusions have been drawn from the findings. Nonetheless, some of these conclusions from this study should help SPI managers establish more effective implementation strategies for SPI.

Finally, we suggest that the use of the technique in this research extends the use of MDS in software engineering research and lays a step for future software engineering research to apply this technique to gain a richer understanding of other software engineering concepts.

### References

- Alison, L. J., Reddy, S., Canter, D. V. & Stein, K. C. (1998). *Behavioural circumplexes in sexual assault and armed robbery*. (Internal document). Liverpool, UK: The University of Liverpool, Centre for Investigative Psychology.
- Baddoo N and Hall T (2003) De-motivators Of Software Process Improvement: An Analysis Of Practitioners' Views. *Journal of Systems & Software* 66(1):23-33
- Baddoo N and Hall T (2002) Software Process Improvement Motivators: An Analysis Using Multidimensional Scaling. *Empirical Software Engineering* 7(2):93 –114
- Canter, D. V. & Heritage, R. (1990). A multivariate model of sexual offence behaviour: Developments in "offender profiling" – I. *Journal of Forensic Psychiatry*, 1(2), 185-212.
- Canter, D. V. & Alison, L. J. (Eds.) (1999), *The Offender Profiling Series: Vol. 5. Profiling rape and murder*. Aldershot: Dartmouth.
- Donald I. (1994). The Structure Of Office Worker's Experience Of Organisational Environments. *Journal Of Occupational And Organisational Psychology*. 67(3): 241-258
- Groth, N. A. (1979). *Men who rape: The psychology of the offender*. New York: Plenum.
- Guttman L. (1968). *A General Nonmetric Technique for Finding the Smallest Coordinate Space for a Configuration of Points*. *Psychometrika*. 33: 469-506
- Guttman R., Greenbaum C.W. (1998). Facet Theory: It's Development And Current Status. *European Psychologist*. 3(1)
- Hall T., Wilson D. 1997. Views Of Software Quality: A Field Report. *IEE Procs On Software Engineering*. 114(2): 111-118
- Kitson D.H., Masters S.M. (1993). An Analysis of SEI Software Process Assessment Results: 1987-1991. *15th International Conference on Software Engineering*. Baltimore, Maryland, May 17-21
- Krueger R.A., Casey M.A. (2000). *Focus Groups: A Practical Guide For Applied Research* (3rd ed.). Sage Publications
- Mead A. (1992). Review Of The Development Of Multidimensional Scaling Methods. *The Statistician*. 41(1): 27-39
- Morgan D.L., Krueger R.A. (1993). *When To Use Focus Groups And Why*. In DL Morgan (Ed.), *Successful Focus Groups: Advancing The State Of The Art*: 3-19. Sage
- Shye S., Elizur D., Hoffman M. (1994). *Introduction to facet theory*. Sage: Thousand Oaks, CA
- Young, S. M., (2004) *An Exploratory Investigation Into Personality*, SCAT, University of Sunderland, Sunderland



## Session 5: SPI and Learning Factors

### Appendices A Definition of de-motivators (Content Category Dictionary)

Plot codes			De-motivator	Definition
D1		S2	Budget constraints	Budgets do not allocate resources specifically to SPI, therefore SPI work becomes a drain on overall budget
D2	P1	S3	Commercial pressures	Pressure to satisfy commercial/financial objectives of company
D3	P2	S4	Cumbersome processes	Processes that are bureaucratic and difficult to implement
D4			Customers	Direct interference from customers
	P3		Fire fighting	A policy of tackling problems as they occur as opposed to a proactive long term strategy for <u>tackling problems</u>
D5	P4		Imposition	Imposing SPI as a dictate, without prior consultation with practitioners
D6	P5	S5	Inadequate communication	Lack of communication between different levels in a company and between different functional areas
	P6		Inadequate metrics	Not collecting sufficient metrics to guard improvement
D7	P7	S6	Inertia	Resistance to practices new
D8		S7	Inexperienced staff	New and temporary staff who are not sufficiently knowledgeable of company processes
	P8		Irrelevant objectives/deliverables	SPI objectives are not tailored to "real" needs that practitioners can identify with
D9			Isolated best practice	Best practices are kept within departments/teams/groups and not shared within the company
	P9	S8	Lack of evidence of direct benefits	Practitioners do not have or are not provided evidence of the success of SPI.
D10			Lack of feedback	Practitioners are not given feedback of the SPI outcomes, or of contributions they make towards SPI
D11			Lack of mgt direction/commitment	Senior management do not demonstrate understanding nor commitment to SPI
D12	P10	S9	Lack of overall support	SPI is not overwhelming supported by the practitioner involved in it. There is apathy amongst <u>certain groups</u>
	P11	S10	Lack of resources	The company does not have the resources -staff, time, tools - to properly fund SPI. This is dissimilar to <i>Budgets</i> .
		S11	Lack of SPI management skills	There are insufficient personnel with the appropriate skills to drive (manage) SPI in the company
D13			Lack of standards (different platforms)	The software development function operates across different platforms. There are no overall standards to SW development
	P12		Large-scale programmes	The SPI initiative is too big for the company. To many facets going on at the same time. It creates co-ordination problems
	P13		Low process priority	SPI is given low priority with respect to other project activities
D14	P14	S1	Negative / Bad experience	Previous negative experiences of SPI negates against SPI uptake amongst practitioners
		S12	Organisational changes	Organisational changes that imply re-allocation of staff and responsibilities impact negatively on <u>ongoing SPI programmes</u>
D15		S13	Personality clashes	Individuals and personal politics frustrates the SPI effort
D16			PM's lack of technical knowledge	Project managers do not possess technical knowledge of SW production hence are unable to appreciate the merits of SPI
D17			Reduced creativity	Practitioners perception of SPI procedures is that it takes away their individual creativity and flair
	P15		Staff turnover	High staff turnover frustrates the nurturing of SPI culture. Ever having to teach new people the established processes
D18	P16	S14	Time pressure/constraints	Pressure to deliver product on time frustrates SPI initiative
D19			Workload	Practitioners have too much work, thus are unable to devote sufficient effort to SPI

**Appendices B** Data matrices for developers, project managers and senior managers' de-motivators

Company	Group	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19
A1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
A2	1	1	1	0	0	0	1	1	0	0	0	1	0	0	0	1	0	0	0	0
	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1
A3a	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0
A3b	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
A4	1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
A5	1	0	1	0	0	0	0	1	1	1	0	1	0	1	0	1	0	0	1	0
	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
A6	2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
	1	0	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0	0	0
A7	2	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
A8	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0
A9	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0
	1	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
A10	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
A11	1	1	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1
	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
A12	2	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0

Data matrix of developers' de-motivators

Company	Group	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16
A1	1	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0
A2	1	1	1	0	0	0	0	1	0	1	0	0	0	0	0	1	1
	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
A3a	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1
A3b	2	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0
A4	1	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0
A5	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A6	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0
	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
A7	2	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	1	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1
A8	2	1	0	0	0	0	0	1	0	0	0	0	1	1	0	0	1
A9	1	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	1
A11	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1
A12	1	1	0	0	0	1	1	0	0	1	1	1	0	1	0	0	1

Data matrix of project managers' de-motivators

Company	Group	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14
A1	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0
A2	1	0	0	1	0	0	0	0	0	0	1	0	0	0	1
A3a	1	1	1	0	0	0	1	1	0	1	1	1	0	0	0
A3b	1	1	1	0	0	0	1	1	0	1	1	1	0	0	0
A4	1	0	0	0	0	1	0	0	0	1	1	0	1	0	1
A5	1	1	0	0	0	0	1	0	1	0	0	0	0	0	1
A6	1	0	0	0	0	0	1	0	0	0	1	1	0	0	1
A7	1	1	0	0	0	0	1	0	0	1	1	0	0	0	1
A8	1	0	0	1	0	0	0	0	0	1	0	1	0	0	0
A9	1	0	0	0	0	1	0	0	0	0	1	0	0	0	1
A10	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0
A12	1	0	1	1	0	0	0	1	0	1	1	0	0	0	1

Data matrix of senior managers' de-motivators

### Author CVs

#### **Nathan Baddoo**

Nathan Baddoo is a senior lecturer in the School of Computer Science at the University of Hertfordshire. He is also a member of the Systems and Software Research (SSR) group. His research focuses on human factors in software processes. He holds many publications in this area. Nathan completed a PhD in Motivators and De-motivators in Software Process Improvement, in 2001. His current research interests are in the general areas of software quality, software process improvement and software practitioner motivation. Nathan is also exploring how to construct prediction models of software engineers' motivation by using concepts like Agent-based simulation and fuzzy logic. Nathan is a programme committee member of EUROSPI<sup>2</sup>2006

#### **Tracy Hall**

Tracy Hall is a principal lecturer in the School of Computer Science at the University of Hertfordshire. She heads the Systems and Software Research (SSR) group, previously she headed the Centre for Systems and Software Engineering (CSSE) at South Bank University. For the past ten years she has been active in the area of software quality, measurement and software process improvement. She has many publications in this area. All of the research that Tracy undertakes is in direct collaboration with companies in the software industry. To date she has collaborated with over 25 companies. Over the last eight years she has been a programme committee member of the British Computer Society's annual Quality Management conference.

She is also an active member of the UK think tank: the Centre for Software Reliability Council. She is the co-editor of a book on professionalism of software practitioners, 'The Responsible Software Engineer', Springer, 1997 and has chaired highly successful international conference in that area.

#### **Ciarán O'Keeffe**

Ciarán O'Keeffe is currently employed at Liverpool Hope University where he specialises in researching, and teaching, three areas of psychology: Parapsychology, Criminal Psychology & Music Psychology. Following an MSc. Investigative Psychology at Liverpool University where he specialized in using Multi Dimensional Scaling techniques to analyse fraud cases and various aspects of cybercrime, he recently completed his PhD from University of Hertfordshire, supervised by Professor Richard Wiseman. The title of the PhD was "Assessing the content of advice given by practitioners who claim a paranormal ability". He's also been the lead psychologist in a project examining Infrasound which incorporated a live concert at the Purcell Room in London. Additional parapsychological research includes: spontaneous cases research, mediumship; psychic criminology; and 'Christian' parapsychology (i.e. exorcism, possession and stigmata). He has participated in numerous bizarre projects (e.g. ghost investigation of Hampton Court & Edinburgh Vaults, Dream project at ancient sites, ghost investigation training day at Oxford Prison – the site of filming for the movie SpyGame!). Aside from academic research in the field, which he has published and presented at numerous conferences, Ciarán attempts to provide a more sceptical perspective on various 'paranormal' shows in the UK (e.g. Most Haunted, Jane Goldman Investigates).

# The Craving for External Training in Small and Medium-sized Software Companies - A Trigger Effect towards Software Process Improvement

*Hanna-Miina Sihvonon, Paula Savolainen, Jarmo J. Ahonen*  
*Department of Computer Science, University of Kuopio*  
*P.O.Box 1627, FI-70211 Kuopio, Finland*  
*{hanna-miina.sihvonon,paula.savolainen,jarmo.ahonen}@uku.fi*

## **Abstract**

In this paper, we discuss how motivation towards software process improvement (SPI) activities can be triggered through training in small and medium-sized software companies. Training needs and their connection to SPI were surveyed during two different training and development projects at two distinct periods under different economic and industrial situations. We compare and analyze the experiences and results from these two projects. We discuss how the knowledge and awareness gained through training can trigger motivation towards SPI. We also present how the focus of the training has changed during the years from technology-oriented training to more SPI oriented and conclude some reasons behind the change.

## **Keywords**

Software process improvement, small and medium-sized companies, training

### 1 Introduction

The software business evolves rapidly and new techniques, architecture, and networking solutions for software development are constantly introduced to the market. All of these new tools and solutions make it possible to build more complex and larger software products. However, these changes have led to a lengthier and more involved software development process. Therefore, it is essential for the employers to support their employees in developing themselves and keeping their skills up to date.

Research has been done on how IT-workforce capability could be maintained and enhanced, for example in [1] and [2] but usually the research concerns higher education and what it has to offer. The attractiveness and usefulness of MSc programs for a software company's employees have improved but they are quite demanding. In general, the goal of higher education is a degree, which requires long-term commitment, perseverance, flexibility from employee's and employer's schedules, commitment from company's management and perhaps monetary support.

Instead of higher education, short-term training would seem an attractive alternative for companies to increase market value and their employees' human capital and capability. There are several benefits of using short-term training, for example, it responds to current and specific needs that emerge from daily business and offers a wide scale of different types of training, from many different technology courses, to more in-depth process- and business-level courses. The benefits are more essential to small and medium-sized software companies than to large companies. Adjusting and scheduling the needed short-term training that derives from business needs is important, because in small companies the knowledge gained from training may affect the whole business.

Small and medium-sized companies play a crucial role in the industry, because they are a wellspring of innovation, exploit new technologies, create job opportunities and keep established firms on their toes [3] [4] [5] [6]. Since the majority of software companies are small [7] [8] [6] and their resources are often limited, their competitiveness can be improved by support for additional training and knowledge. In small and medium-sized companies, a competent and multi-skilled workforce is valuable. Their employees' skills and educational backgrounds can be very heterogeneous and people have different bases for their work. Due to heterogeneity issues, new training needs occur continually along with development needs within a company's software development process.

The problem in small or medium-sized companies is to get the needed training in a timely and cost-effective fashion. For example, the training is too expensive, time-consuming, or arranged in distant locations. Considering the importance of small and medium-sized software companies in the market, it is essential to acknowledge the significance of supporting and encouraging them to improve their employees' competence to respond to new challenges. As it is presented later in this article, the companies' training needs are extensive, but considering the problems presented, they may need an external trigger to begin determined knowledge acquisition and to perceive their needs for SPI.

Externally coordinated training and development projects can be very useful for small and medium-sized companies to get the needed training in time. The needed background information for the trainings and the training arrangements can be coordinated by the project group and the projects provide financial support from external financiers in organizing company-specific, short-term training from commercial training organizations.

In addition, that in the projects the training needs are defined, and the needed training is arranged, the projects can be used as tools to research the SPI activities in small and medium-sized software companies. The purpose of this article is to describe the training needs in the small and medium-sized companies and to discuss the impact of arranged training how it contributes to take actions towards SPI.

In Section 2, we describe the research questions and present two training and development projects that have been used for observing training needs in small and medium-sized software companies. In Section 3, we contemplate the changes in the economic situation and software industry and we discuss the common and different aspects of the two projects. In Section 4, we examine how the awareness that training gives can trigger SPI activities. In Section 5, we summarize and discuss our conclu-

sions and present future research intentions.

## **2 The Research Questions and the Analyzed Projects**

In the Section 2.1, we concentrate on presenting the research questions. In Sections 2.2 and 2.3, we provide some background information of the projects that have been used for observing training activities in small and medium-sized software companies.

### **2.1 The Research Questions**

Our focus is on the following research questions: "Have changes in economic the situation and the software industry had an effect on the emphasis of training needs and contents?" and "Does training trigger SPI activities in the small and medium-sized software companies?". We will also assess, whether the support and facilitation in organizing training that the projects provide, have an effect on the awareness and interest to contribute to SPI activities.

We have had an opportunity to observe two different training and development projects in geographically restricted regions in Finland. The first project, hereafter referred to as the "Mars project", was carried out at the turn of the century in 2000 when the dot-com mania was at its hottest. The goals of the project were to train the employees to use new technologies and methods and to give consultative training on how to apply these new skills to daily work. At the beginning of 2006, quite a similar kind of project, hereafter referred to as the "Venus project", was started and is currently running. Both of the projects aid companies in clarifying their short-term and long-term training needs, so that it will be easier to predict and react to these needs. The research questions are considered from the perspective of the experiences and results of these two projects.

### **2.2 The Mars Project**

At the end of year 1999, the training and development Mars project was established to provide training for small and medium-sized software companies. The project was to last 23 months and it finished at the end of 2000. The project was partly funded by the companies involved and partly by external funding. The goal of the project was to enhance companies' technological capability and to develop companies' software processes through training and consulting. During the project, the focus concentrated around technical training directly related to software development tools and techniques. However, also the need for training that aimed at process and method development emerged towards the end of the project.

The target population of the project was small and medium-sized software companies. The population included 14 companies of different sizes, varying from 8 to 85 employees. The companies answered a training needs questionnaire at two different phases of the project. The companies that were interested in participating in the project were surveyed in the first phase of the project. Company-specific training needs were traced through a questionnaire and companies were asked to provide suitable schedules for arranging the training. In the early stages, it was perceived that the training should be organized in a company-specific manner and timely, depending on each company's business needs.

In the second phase, the questionnaire was characterized differently. In addition to identifying training needs, a satisfaction questionnaire concerning the previously organized training was included, as well as a questionnaire of future training intentions. From both questionnaires, the data was combined and assembled into comparable form.

### 2.3 The Venus Project

In the beginning of 2006, a local training and development project Venus, was established to provide training for small and medium-sized software companies. The project will last until the end of June 2007. Like the Mars project, the Venus project is partly funded by the companies involved, and partly funded by external resources. The goal of the project is to awaken companies' interest towards more profound thinking of processes through technology, process, and business level training. Several companies were interviewed and from them, the ones who had a real craving to increase their human capital were assorted in the project. There are now 11 companies involved in the project, sizes varying with 4 to 130 employees.

The companies have eagerly seized the opportunity to enhance their human capital and they are actively involved. The process how to obtain training needs from the companies is still evolving. Some experiences and preliminary results of the training needs are presented in Section 3.3.

## ***3 Changes in the Software Industry and the Common and Different Aspects of the Projects***

Training needs usually depend on timing and the current state of affairs. Therefore, the first thing we discuss in the Section 3.1 is the difference in economic situation during each of these projects, and how it may have affected the emphasis in the contents of training. In Sections 3.2 and 3.3 we review the experiences and information from the training needs questionnaires of the two projects.

### 3.1 The Economic Situation and Changes in the Software Markets

During the late 1990s and early 2000s, the dot-com mania was in full swing. As presented in [9] and [10], the Internet and eBusiness growth was rapid and made many believe that the old rules of business no longer applied. Several new companies were established on the basis that the Internet boom would transform the way consumers and businesses behaved overnight. Many of these dot-com companies had unrealistic and unstable business models, but the fever of investors to eagerly cash in on the speculative growth of these companies led to an enormous stock market boom, which led to an equally enormous crash in early 2000. Also the predicted Y2K problems temporarily increased IT budgets in 1997 through 1999, thereby adding to the demand for maintenance services [11]. The Mars project was carried out during this turbulent economic period.

On the other hand, the Venus project was started in the beginning of 2006, and now the starting point from an economic perspective is quite different. The oddities of the dot-com boom have faded into the background, and the economic situation has returned to a somewhat realistic level. Now, IT companies have to return to their roots, back to basics, to maintain market share and to ensure long range survival and continuity. They have to provide real value to clients, understand their needs, and make the delivery process more successful. Instead of selling the latest technological fad without considering the benefits or genuine business need, they should seek to improve value from existing IT infrastructure and systems that most organizations struggle to control. Projects should be delivered more effectively, and real business value should be squeezed out of current system [12].

Despite the evolution towards less technology-focused software business, the possibility to innovate new technology solutions in other industry branches exists. These new business opportunities still require dot-com types of actions in the whole business, e.g. investments from venture capitalists, a quick development process, use of high technology, or perhaps even releasing unfinished products to the market. [13]

The experiences from the Mars project and the Venus project are presented and analyzed from a time- and situation-relative perspective. We will describe the starting point for both of the projects and the observations made during the projects.

### 3.2 The Mars Project

It was surprising how many unique training needs only 14 companies can provide. Overall, the amount of unique trainings that were arranged was 118. The trainings are divided in Table 1 to business, process, and technology categories. The subcategories of trainings, for example different database product and tools, are excluded. The emphasis of the trainings was on technology category. 19 training subjects were from technology category and only 4 from process or business category. Especially Java related courses and different database products and tools were very popular, arranged altogether 40 times. The need for the business or process related trainings was insignificant. There were only few business category trainings and those were arranged only once. The process category trainings, project management and UML related trainings were arranged altogether 5 times.

**Table 1.** Organized trainings in the Mars project.

Training	Training category	Amount of arranged training	Training	Training category	Amount of arranged training
Software Quality Management	Business	1	Internet technologies	Technology	8
Outsourcing	Business	1	Data Warehousing	Technology	6
Project management	Process	3	Mobile technologies	Technology	5
UML	Process	2	IBM products	Technology	4
Java	Technology	19	SQL Server	Technology	4
MS Excel	Technology	9	CORBA	Technology	1
Visual Basic	Technology	8	Architectures	Technology	1
Crystal Reports	Technology	5	Linux	Technology	1
XML	Technology	7	Windows 2000 Security	Technology	1
Databases	Technology	21	Active Directory	Technology	1
Operating systems	Technology	7	Telecommunications and Data Security	Technology	1
C / C++	Technology	2			

The participation activity varied widely. From a small company almost everyone participated in arranged training, on the other hand, from a much larger company only few people participated in arranged training. There was no obvious reason that would explain the varied participation. The differences between companies might have derived from the company's training needs or the training subject.

As was mentioned earlier in Section 2.2, a satisfaction questionnaire was sent to all companies. At this point, there was still half a year remaining until the end of the project. About 60% of planned trainings had been organized and 35 different training organizations had arranged training. However, only 5 different training organizations had arranged two thirds of the training. Because of this, in the satisfaction questionnaire, the evaluation mainly concerns these 5 organizations.

The focus was to get answers regarding the following quality topics: how well the contents met the companies' training needs, and whether the price level as well as the training arrangements were satisfactory. According to the questionnaire, the participants were satisfied with the contents of the trainings. Even though the role of project management training was not significant, the questionnaire showed that the participants were the most pleased with this training. It is also important to note that the most arranged training, Java and database related trainings, received also the most criticism. Furthermore, the training for the usage of programming tools was especially criticized.

The last part of the questionnaire explored future training intentions three years in the future. The an-



swers showed that the companies intended to use more money in training and to add more employees training. It is noteworthy though that the training days do not divide evenly between the employees in a company, since the starting point and length of training derives from current assignments and changes in job description. The results from the training needs questionnaires, satisfaction questionnaires, and the expression of future intentions to use more money for training indicate that interest towards SPI has been aroused.

### 3.3 The Venus Project

In the first phase of the Venus project, the companies were asked to provide a preliminary list of their training needs. In the second phase, the project members visited each company and performed interviews based on each company's preliminary list to find out the training contents in more detailed level. The preliminary lists were rather technology oriented, but as these lists were refined in company-specific interviews in collaboration with company's representative and project group members, it turned out to be a false assumption. It became clear that the presented trainings were only a glimpse of what was really needed. In many of the suggested trainings, the real need derived from the companies' processes, business needs and the needed training is to be arranged from process perspective, not technology perspective.

The trainings organized and planned to be organized in the Venus project are presented in Table 2. Most of the trainings will be arranged in the future. Therefore, each of the trainings has a priority level and training needed occurrence. The companies have prioritized the trainings and the priority level is average of these. The training needed occurrence shows how many companies are interested in the training. Based on the priority level and training needed occurrence it is can be seen what subjects are highly prioritized and how many companies are interested in those subjects.

From Table 2, it can be seen that over half of the suggested training subjects, 13, are from business or process categories. Only 10 subjects are from technology category. The most highly prioritized and the most occurred trainings are training towards international markets, testing, UI Design and usability and project management. However, the project management is the most needed training, since 8 out of 11 companies are interested in it.

**Table 2.** Trainings organized and planned to be organized in the Venus project.

Training	Training category	PL	TNO	Training	Training category	PL	TNO
Training towards international markets *)	Business	H	5	Technical writing	Process	L	2
Negotiation skills	Business	A	2	XML	Technology	L	2
Incorporeal rights	Business	L	2	SOA *)	Technology	A	3
Technology management *)	Business	A	3	SIP	Technology	L	1
Software business	Business	A	3	Flash	Technology	L	2
Software architectures	Process	A	4	SQL Server	Technology	A	4
Testing	Process	H	6	JAVA	Technology	H	4
Version management	Process	L	2	Active directory	Technology	A	3
UML	Process	H	4	.NET	Technology	H	5
Effective use of project management software	Process	H	3	Data security and telecommunication *)	Technology	H	4
UI Design and usability *)	Process	H	6	Linux	Technology	L	1
Project management *)	Process	H	8				

PL = Priority level (H=High, A=Average, L=Low)

TNO = Training needed occurrence

\*) = At least one training in this topic has been arranged

Based on the refined lists and the interviews, the companies are now ready to see training as a tool and external push towards process improvement. They are interested in training that supports the software engineering process, for example project management, testing and UI Design and usability. The current summary of training needs shows that the trend is moving from technology training to process and business level training.

Furthermore, the companies expressed remarkable interest towards training that prepares for certification. In [14], Prabhakar et al. claim that the employers nowadays prefer job applicants who have certification in the required area of expertise. Certification provides a greater sense of confidence to job applicant's abilities and it is a measure of professional expertise [2]. Nowadays the employers seek and value certified workforce and that indicates that they also support their employees training and development. The employees appreciate professional development opportunities almost as much as financial rewards. Therefore, the employee turnover may decrease and this supports the long-term SPI efforts. Considering the importance of certifications, in the Venus project, the basis for planning the training is that it will prepare the attendants for certification.

In addition to training, the Venus project also includes a development phase during which the companies get small-scale consultative aid in defining their processes, information flows, and problems in software engineering. Based on the information that consulting provides, the companies can be assisted in determining their long-term SPI supportive training needs.

### **4 The Trigger Effect towards SPI**

As was described in the beginning of the Section 3.1, the economic and industrial situations have varied during these projects. The late 1990s and the beginning of year 2000 were peculiar period. All of the following, the dot-com mania, venture capitalist's enthusiasm to invest on dot-com companies, predicted Y2K problems, mobile technology boom and the growth of the Internet and eBusiness, have had an indelible effect overall software industry. In this frenzy, it was easy to cast aside the old business rules and go with the flow. The capability to adopt new technologies quickly was a necessity and the quality issues and well-managed business processes were not so emphasized.

All of the factors listed above may have had an impact on the focus of the trainings that were arranged during the Mars project. The trainings concentrated on a wide variety of technology subjects. The process-level training was insignificant at the early stages of the project. As the effects of dot-com boom and other factors diminished during the passage of time and the companies became aware of the training benefits, the need for the process improvement training increased. Especially the project management training was in great demand, which was a sign of companies' interest towards SPI efforts. Since the Mars project was already at its end and it was not possible to organize more training, a new two-year project was established that concentrated strictly on project management related training.

At present, as the economic and industrial situation has settled to a more stable level, the focus of the training needs has changed. This change was observed already in the beginning of the Venus project. Now the focus seems to be not only in technological training but even more on training that supports process improvement in all levels starting from best practices and standards in programming activities to understanding of customer's business processes. As it was already mentioned in the Section 3.3, the companies' real training needs behind the technological course names actually derived from the process and business level needs.

It is worthwhile to point out that the information and knowledge gained from the trainings triggers companies to contemplate and analyze their own processes. Before the companies are able to improve current processes, they have to identify what the processes are and where the problems in the processes occur. The newly learnt information makes the companies more aware of the flaws and weaknesses in their own processes and this observation may initiate the improvement actions.

For example, in the Venus project's project management training, the companies' knowledge of the process activities involving the software engineering projects was increased remarkably. The training included several exercises and examples of software projects. During the training, the CMMI was introduced and quality issues discussed. The awareness obtained through this training may arouse in-

terest towards spontaneous SPI activities and increase the motivation to acquire more in-depth SPI oriented training.

In order to get the SPI activities going on, the small software companies require external support to initiate software process improvement [15]. During the Mars project, the companies' readiness to develop SPI activities increased towards the end of the project, as for example the project management trainings gained ground. However, in the Venus project this tendency is already seen at the early stages of the project. In addition to that and to increase the motivation towards SPI, the training exercises are planned from process perspective and the exercises force the participants to contemplate their companies' processes.

In comparison to SPI activities in large software companies, where it is a long-term and heavy process [16], in the small and medium-sized companies the preparation and implementation of SPI activities is different. Even though it is a long-term journey in small and medium-sized companies too, it is an agile, cyclical process and carried out relatively fast [17]. The changes in companies' business orientation, growth, clientele, and economic situation affect the current SPI needs. Training and development projects, such as Mars and Venus projects, can give a push towards development activities especially in small and medium-sized software companies. The support that the projects provide should last for a longer period so that the SPI activities could be carried out continuously.

### **5 Discussion**

In this paper, we have studied training needs in small and medium-sized software companies in two different periods and under different economic situations. The companies need for training was significant. The training needs were endless and new needs will emerge, as the business needs change. The companies in the Mars project had 118 unique trainings during the project and they estimated to invest even more on training in the future. In Venus project, the companies' training needs were so far exhausting, and it is not possible to arrange all of the needed training in the short run.

The Mars and Venus projects provided the companies with external support in organizing and financing the trainings. Often, in the small and medium-sized software companies the lack of resources limits their possibilities to acquire training. Without the support, the companies could invest less on training and development efforts. We observed that the support that the training and development projects provide, might also give an external push to the companies to contemplate their own processes.

In the Mars project, the trainings were technology oriented but towards the end of the project, the desire to invest on process oriented trainings, such as project management, aroused. In the Venus project, the training needs were derived from business and process development needs already from the beginning of the project. The training needs have changed during the different periods of time. The reason behind the change in the Mars and Venus projects is, at least partly, the considerable changes in the economic situation and software industry.

The need to improve the processes clearly exists in many companies and they will likely take actions towards improvements eventually. However, an external push can speed up the decision to begin the SPI activities. The small and medium-sized software companies need external support in the training, which increases awareness and knowledge and this may catalyze the SPI motivation.

In conclusion, the results and experiences from these projects indicate that the focus is moving towards training that supports process thinking and the gained awareness may trigger SPI activities. From the experiences of the projects, it can be concluded that externally coordinated training and development projects can have a trigger effect towards SPI. However, it is not possible to separate what other factors than training might have contributed to favorable direction towards SPI. The presented conclusions were based on the reported data from the Mars and Venus projects. Some data from the Mars project was incomplete, which limits the possibilities to present results that are more detailed. In order to get more exact data to support the conclusions in the future research, some metrics and key figures may be applied in the Venus project. Considering the future research in SPI activities, we will use the development phase of the Venus project.

## 6 Literature

1. Bills, D. P.: Marketing Higher Education in Information Technology. SIGITE'04, October pp. 28-30, 2004.
2. Montane, R., Khan, Z.: Specialized Certification Programs in Computer Science. SIGCSE 2001, 2/01, pp. 371-375, 2001.
3. Wheelen, T. L., Hunger, D. J.: Strategic Management and Business Policy. 9th ed., Pearson Education, 2002.
4. Chin In Sing, A.: 10 Factors on Fostering Innovation in Small and Medium-sized Organizations. ICMIT 2000, pp. 473-477, 2000.
5. Vähäniitty, J., Rautiainen, K.: Towards an Approach Managing the Development Portfolio in Small Product-Oriented Software Companies. Proceedings of the 38th Hawaii International Conference on System Sciences, 2005
6. Baskerville, R., Pries-Heje, J.: Knowledge Capability and Maturity in Software Management. The DATA BASE for Advances in Information Systems, Vol. 20, No. 2, pp.26-40, 1999.
7. Cater-Steel, A. P: Process Improvement in Four Small Software Companies. Proceedings of the 13th Australian Software Engineering Conference (ASWEC'01), pp. 262-272, 2001.
8. Fayad, M. E., Laitinen, M., Ward, R. P.: Software Engineering in the Small. Communications of the ACM, vol. 43, no. 4, pp. 115-118, 2000.
9. [http://www.philipson.info/files/a\\_short\\_history\\_of\\_software.pdf](http://www.philipson.info/files/a_short_history_of_software.pdf), referred 31st March 2006.
10. Tschritzis, D.: Life Beyond the Bubbles. Communications of the ACM, Vol. 46, No 5, pp. 25-27, 2003.
11. Greenstein, S.: The Ride Before the Fall. An Inside Scoop on the High-Tech stock market bust. IEEE Micro Economics, Vol. 22, Issue 1, pages 4-5, 91, 2002.
12. Holmes, A., Ryan, J.: The End of the Long Boom. Communications of the ACM, Vol. 46, No. 12ve, 2003.
13. Cusumano, M.A.: The Business of Software: What Every Manager, Programmer, and Entrepreneur Must Know to Thrive and Survive in Good Times and Bad, New York, Free Press, 2004.
14. Prabhakar, B., Litecky, C. R., Arnett, K.: It Skills in a Tough Job Market. Communications of the ACM, Vol. 48, No. 10, pp. 91-94, 2005.
15. Saastamoinen, I., Tukiainen, M.: Software Process Improvement in Small and Medium Sized Software Enterprises in Eastern Finland: A State-of-the-Practice Study. EuroSPI 2004, LNCS 3281, pp. 69-78, 2004.
16. Zahran, S.: Software Process Improvement: Practical Guidelines for Business Success, Addison-Wesley Professional, 1998.
17. Ward, R. P., Fayad, M. E., Laitinen, M.: Software Process Improvement in the Small. Communications of the AMC, Vol. 44, No. 4, pp. 105-107, 2001.

### **7 Author CVs**

#### **Hanna-Miina Sihvonen**

Hanna-Miina Sihvonen is a PhD student at the University of Kuopio. Her research interest is software process improvement from the employees' point of view. She received her MSc in software engineering from the University of Kuopio. She has worked as a software engineer in Tekla Corporation in various projects.

#### **Paula Savolainen**

Paula Savolainen is a project manager at the University of Kuopio. Her research interest is software business especially in small companies. She received her M.Sc.Econ. from the University of Jyväskylä. She worked in TietoEnator over 10 years in different positions and in Deio, which is now a part of GE Healthcare. She has also worked as IT-coordinator in the North-Savo region.

#### **Jarmo J. Ahonen**

Jarmo J. Ahonen is a professor of software engineering at the University of Kuopio.

# European Quality Network (EQN) – A European Human Resource Strategy for Improvement

[www.eu-certificates.org](http://www.eu-certificates.org)

Richard MESSNARZ<sup>1</sup>, Franz Piller<sup>2</sup>, Bruno WÖRAN<sup>2</sup>

<sup>1</sup>*ISCN GesmbH, Schieszstattgasse 4, A-8010 Graz, Austria*

*Tel: +43 316 811198, Fax: + 43 316 811312, Email: [rmess@iscn.com](mailto:rmess@iscn.com)*

<sup>2</sup>*IMC Krems, University of Applied Sciences, A-3500 Krems, Austria*

*Email: [franz.piller@imc-krems.ac.at](mailto:franz.piller@imc-krems.ac.at)*

<sup>3</sup>*DANUBE, Zieglergasse 28, 1070 Vienna, Austria*

*Email: [bwoeran@danube.or.at](mailto:bwoeran@danube.or.at)*

**Abstract:** This short paper discusses the importance of job role based qualification strategies to achieve a higher capability level.

It describes the objectives and strategies of the EQN project (2005 – 2008) which establishes a shared European job role based qualification strategy for the IT & Services sector.

## 1. The Need for Human Resource Strategies in SPI

Processes usually are defined according to underlying standards (ISO 15504, ESA ECSS, ISO 9001, ...) and are described with process steps to be performed by roles and producing results (outputs) from well defined inputs, methods and tools to support the process steps, and activities to be done and skills to be covered by roles [5].

However, European studies (1998 – at 200 firms [7], 2002 – at 128 multinational firms [8], 2003 – in 59 networked European organisations [1], [2], [3]) illustrate that approx. 2/3 of the factors influencing improvement success are human and organisational factors.

Figure 1 shows the relationship between process assessment and human resources. Actual people will take the ownership of defined roles and thus will require to achieve a certain skills profile, and the achievement of the human resources skills is equally important than achieving a defined process in technical terms.

**Processes and Human Resources**

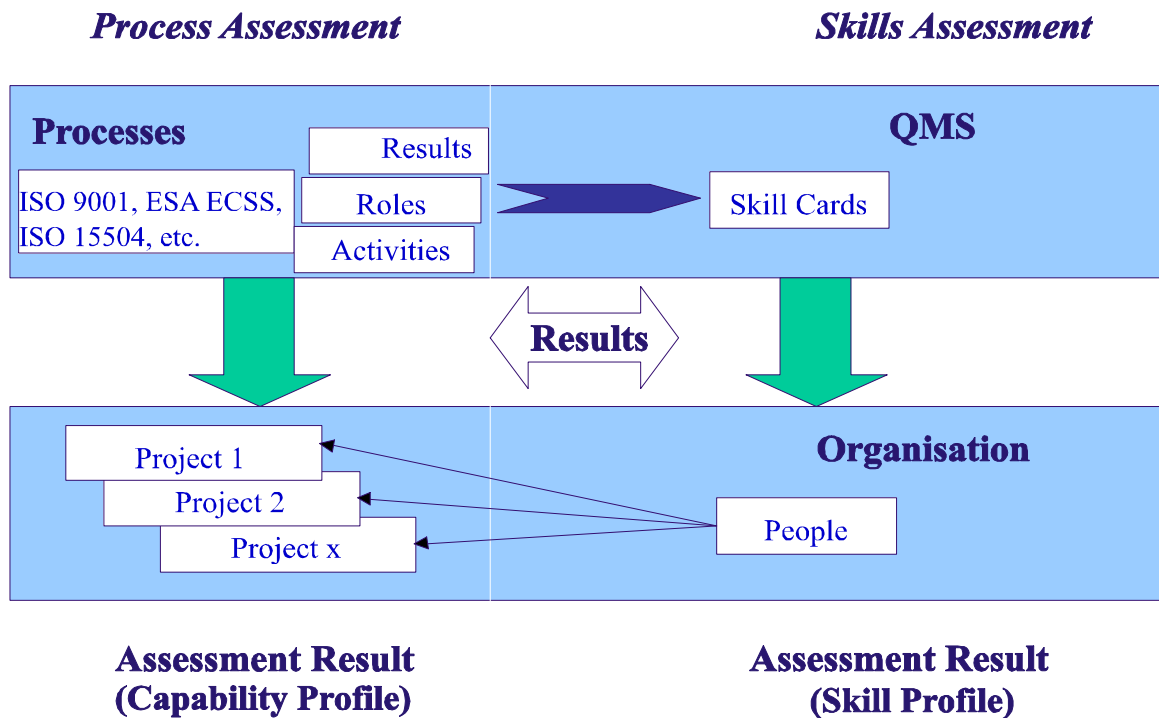


Figure 1: Combination of Process Improvement and Skills Management

**2. Known European Problems with Role Qualification Based Strategies**

At a capability level 3 (ref. ISO 15504 – defined level) organisations have an established set of roles and defined skills profiles required by the roles. When a project is being staffed people are assigned to the roles based on their existing skills profile and competencies.

More and more firms addressing a capability level 3 in the future therefore establish roles such as:

- Project manager
- Configuration manager
- Requirements Manager
- Tester
- Process Improvement Manager
- Innovation Manager
- Etc.

... and establish training plans for their human resources to be trained as such roles.

Major problems in the past were

1. The traditional university programs just focus towards diploma and do not offer specific job role education.
2. Even if semi-public institutes started to offer job role based courses, the content of the courses was not comparable across regions of Europe.

Thus the EuroSPI group with partners joined a consortium and received EU funding to establish a European qualification strategy for job roles. Key job roles are being identified, and all job roles need to fulfil certain European quality criteria to become complete and comparable across Europe.

### 3. The European Qualification Pool – Job Role Based Strategy

At the moment many institutions offer different certificates and job role qualifications in the IT and service sector and many of the certificates are just offered in a limited scope: e.g. just by a consortium of partners of an EU project, by some national qualification institutes, just by a certain university etc.

The value of a certificate is much higher if:

- a. it is based on a modular course architecture (Bologna process in Europe) where training modules can be mapped onto university lectures as well as onto industrial course modules, thus being used across universities and industries.
- b. it is linked to an accreditation association with members of many countries in the board, so that a certificate becomes automatically valid across different countries.
- c. it is based on a shared European knowledge so that all training schemes draw from the same knowledge basis, and participants have gained comparable skills [9].

#### (a) Modular Course Architecture

EQN established a set of quality criteria to assess whether new developed job roles can be accepted for being integrated into the [www.eu-certificates.org](http://www.eu-certificates.org) platform. One major criteria will be a clear and modular course architecture, with a course syllabus and a clear skills definition [2], [9] based on European skills definition norms (skill card standards).

#### (b) Association and Recognition

EQN designs a joint model of how to integrate a selected set of previous Leonardo projects consortia (representing certain acceptable job role qualifications – fulfilling the quality criteria) , industrial qualification institutions, universities, and national training institutes into an accreditation association, with a defined process for issuing certificates.

#### (c) Shared European Knowledge

EQN integrates representatives from a set of major European conferences in the IT and Services sector. Using and extending the software ([library.eurospi.net](http://library.eurospi.net)) from the EuroSPI initiative to administer and make accessible publications online, we plan to create an online knowledge space with many hundred experience reports and papers, from which students and trainers can draw Europe wide.

It is a strategic objective that EQN is exploiting synergies between all different groups and is NOT establishing something competing with existing structures.



#### 4. First Pool of Professions

##### 4.1 Analysis of Quality Criteria and First Key Job Roles to be Supported

The following job roles have been selected to be supported in a first phase until end of 2007:

Nr.	Title	Short Description
1	SPI Manager	SPI manager Project leaders and project workers in small and medium-sized software companies need to improve the work in their own company. They need practical information on how to get started with and implement improvements in their company.
2	FiSMA Scope Manager	FiSMA Scope Manager 1. have good understanding of information system requirements 2. be capable to help project staff to complete and adjust the requirements 3. be competent in functional size measurement using at least one of accepted FSM methods 4. be capable to analyse project circumstances and understand their influence to project effort and schedule 5. have access to at least one project database, where he or she can find and determine the applicable delivery rate (h/fp) 6. be working for an organisation, which is member of FiSMA Experience Division 7. have licence an be competent to use FiSMA data collection tool 8. have proved evidence of earlier data collection and good knowledge of data validation rules.
3	Innovation Manager	The Certified Innovation Manager Course is specifically designed for providing managers (of IT and service companies, not for profit innovation associations, and innovation decision makers) with the knowledge and skills needed to establish: Creative Organizational Learning Environments, Knowledge Management Concepts, Customer Relationship based Learning and Planning, Market Research based Strategies, Team Learning Environments, Team Motivating Environments, Concepts to favour and empower personal skills, systemic innovation processes with dynamic feedback loops, project management that unleashes innovative powers of team members, organizational concepts allowing innovative persons to reach acceptance for new ideas and strategies. Delegates who pass tests for the selected learning elements will receive an innovation manager certificate
4	EU Project Manager	The Certified EU Project Manager Course is specifically designed for providing managers (of IT and service companies, not for profit EU service associations, and SME managers) with the knowledge and skills needed to establish: Planning an EU project (objectives, measurement, exploitation etc.), Quality Management in an EU Project, Financial management and Control in an EU Project, Deliverable Design, Integration and Review Strategies in EU Projects, Team- and E-Working Examples rom Successfully Managed EU Projects, Overview of Related Materials - PMBOK, Prince2, ISO 15504, etc. The training material is

		based on two previous IST projects which developed and tailed systems for e-smart organizations in EU projects (44 projects) involving 59 organizations. Delegates who pass tests for the selected learning elements will receive an EU project manager certificate
5	IT Consultants	IT consultants for SMEs. Within this label, the job role can be organised into 3 sub-job roles : - IT consultant for SMES - IT strategy consultant - IT consultant for SMEs - Package selection - IT consultant for SMEs - IT maturity assessor (particularly adapted for Very Small Enterprises, aiming at assessing their maturity for all IT processes - ITIL oriented)
6	Information & Communication Systems Engineer	Job Role Title: Information and Communication Systems Engineer Description: The job role contains five major units: 1. ICSE Fundamentals 2. Information Systems 3. Communication Systems 4. ICSE Supportings 5. ICSE Electives
7	ISQI Requirements Manager	Standardised education for methods and techniques for requirements analysis, requirements management and tracking and its usage in systems engineering.
8	ISQI SW Architect	Standardised education for basics in software architectural design and its importance for the project life cycle.
9	ISQI Project Manager	Standardised education for project planning, project organisation, project management, and project control.
10	E-Business Manager	The job role contains three major units (Understanding E-Business, Managing E-Business, Managing E-Support Measures) and 12 learning elements with learning objectives.
11	Configuration Manager	An important role is Configuration Manager. This is an very important role in relation to maturity and include the most difficult process Configuration Management - to reach level 2. This proces is very often recommended as improvement initiative. This is an process area which has to be strengthen, and education and certification is a necessity.

**Table 1: Professions Proposed for a Phase 1**

*4.2 Online Portal and Knowledge Library*

EQN integrates existing online services, such as

- Existing skills assessment learning portals for job roles like innovation manager, e-business manager, configuration manager, etc.
- Existing online publication library systems (e.g. library.eurospi.net)
- Links to existing job role qualifications (developed outside the EU funding)

into a portal which acts as a door opener for people searching for IT and services qualifications.

EQN also re-uses the accreditation portal systems (previously developed in an EU project, 1998 – 2001) where (like in ECDL) **tests are done online, tests results are automatically displayed and all certificates are administered.** Some of the job roles use this already.

This results in:

- An online central portal for job role based qualification guidance and strategies and access to skills cards
- An online available case study library
- Access to above 600 experience reports structured according to the key success factors analysed

### 4.3 EQN Association Foundation

The EU-Certificates organisation could be founded as an accreditation association, seated in Vienna, managed by Danube and IMC Krems. All partners of EQN become founding board members, plus those who will be invited to the founding conference. Later all project partners from participating EU projects can join as members as well. Every 2 years a director is elected from all members who will be heading the management team (those managing the EU certificates and the test portal systems) and members for 2 years.

The services are

- Accreditation of training institutions who offer specific job roles and publishing the list of accredited training institutions
- Accreditation of trainers who offer specific job roles and publishing the list of accredited trainers
- Certification of students and publishing the list of certified students (list of all innovation managers who are certified...etc.)
- Access to the online knowledge library through a flat fee per year

The core group contains 17 organisations, while through the founding conference the founding members will comprise approx. 70 European training organisations. The cooperation model is different depending on the organisation type.

#### Example 1: University

A technical college, university of applied sciences, etc. would be accredited that a certain study program covers a specific job role. Then each graduating student can apply for an EU-Certificate valid in industry automatically, at a defined fee. This way students finishing can already approve certain skills needed by industry and increase their skill scope.

#### Example 2: Industrial Body with Existing Job Roles

ISQI ([www.isqi.org](http://www.isqi.org)) already offers a set of certificates in industry, such as certified tester, certified requirements engineer, etc. Then each student who has achieved an ISQI certificate can upgrade his certificate to become valid for a number of countries, represented by EU-Certificates representatives.

ISQI can extend their services to offer job roles (e.g. certified innovation manager,..) from the EU-Certificates pool which they do not have in their portfolio so far.

#### Example 3: National Training Centers

DELTA ([www.delta.dk](http://www.delta.dk)) developed a new job role, such as configuration manager . They can get this new job role accredited and promoted by EU-Certificates, and offer other EU-Certificates partners to become accredited trainers. Then this certificate becomes valid for a number of countries, represented by EU-Certificates representatives.

### Example 4: Chambers of Commerce

Chambers of commerce must stay up-to-date with their services all the time. The EU-Certificates pool gives them access to the new job roles from the Leonardo programme in the IT and services sector and selected (with high demand) industry qualifications, which they can then offer (as accredited EU-Certificates organisation) to the SMEs.

### Example 5: Conferences

European conferences can widen their scope (normally just presentations, workshops, and tutorials) to offer a European training school with valid EU-Certificates around the conference.

## 5. Conclusion and Outlook

EQN has developed

- Accreditation rules for new job roles and professions
  - [http://www.iscn.com/projects/eu-certificates/eqn\\_job-roles-overview\\_and\\_acc\\_req\\_v1.zip](http://www.iscn.com/projects/eu-certificates/eqn_job-roles-overview_and_acc_req_v1.zip)
- Accreditation of training organisations and trainers
  - [http://www.iscn.com/projects/eu-certificates/eqn\\_training\\_acc\\_req\\_v1.zip](http://www.iscn.com/projects/eu-certificates/eqn_training_acc_req_v1.zip)
- Different certification levels and Blooms Taxonomy based education strategy
  - [http://www.iscn.com/projects/eu-certificates/eu-certificates\\_certification\\_process\\_v3.zip](http://www.iscn.com/projects/eu-certificates/eu-certificates_certification_process_v3.zip)
- Skills assessment and examination portals
  - [http://www.iscn.com/projects/piconew\\_skill\\_cards/index.html](http://www.iscn.com/projects/piconew_skill_cards/index.html)
  - <http://www.innovationmanager.org>
  - <http://www.manageur.com>
  - Etc.
- Process models for operating a European certification association
- Legal statutes for founding EU certificates
  - [http://www.iscn.com/projects/eu-certificates/0606\\_EQN\\_Statutes\\_v2.zip](http://www.iscn.com/projects/eu-certificates/0606_EQN_Statutes_v2.zip)

In addition an international founding conference takes place in Krems (near Vienna) on 5.12.2006. If you are a training organisation or university and want to join this strategy please contact [rmess@iscn.com](mailto:rmess@iscn.com) with title EU-Certificates Collaboration.

The conference agenda is described below:

Start : 09.00

09.00 - 09.15 : Welcome by the director of the Univ. of Applied Sciences Krems, Austria

## Session 5: SPI and Learning Factors

09.15 - 09.45: Key Note by Leonardo da Vinci National Agency of Austria – Future Quality Strategies in Life Long Learning

09.45 - 10.15: The EU-Certificates Association – The Strategy

10.15 - 10.35: Coffee Break

10.35 - 11.20: The EU Certificates Accreditation Process, Rules, and Systems

11.20 - 11.50: The EU-Certificates Legal Framework

13.00 - 14.30: Six selected examples of accredited professions

14.30 - 14.50: Coffee Break

14.50 - 15.20: EU – Certificates Test and Examination Systems

15.20 - 15.50: Future Learning Platforms

16.00 - 16.20: Official Signature on Founding Papers

Reception

From 16.20: Christmas market and film bar



**Figure 2: The EQN Core Team**

## References

- [1] M. Biro, R. Messnarz, A. Davison (2002) The Impact of National Cultures on the Effectiveness of Improvement methods - The Third Dimension, in Software Quality Professional, Volume Four, Issue Four, American Society for Quality, September 2002
- [2] 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, Washington, 2002
- [3] Feuer E., Messnarz R., Wittenbrink H., Experiences With Managing Social Patterns in Defined Distributed Working Processes, in: Proceedings of the EuroSPI 2003 Conference, 10-12 December 2003, FTI Verlag, ISBN 3-901351-84-1
- [4] Project EASYCOMP (IST Project 1999-14191, homepage: <http://www.easycomp.org/>)
- [5] 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
- [6] 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
- [7] A Learning Organisation Approach for Process Improvement in the Service Sector, R. Messnarz, C. Stöckler, G. Velasco, G. O'Suilleabhain, A Learning Organisation Approach for Process Improvement in the Service Sector, in: Proceedings of the EuroSPI 1999 Conference, 25-27 October 1999, Pori, Finland
- [8] O'Keefe, T., & D. Harrington, 2001. Learning to Learn: An Examination of Organisational Learning in Selected Irish Multinationals. Journal of European Industrial Training, MCB University Press, Vol. 25: Number 2/3/4
- [9] DTI - Department of Trade and Industry UK, British Standards for Occupational Qualification, National Vocational Qualification Standards and Levels
- [10] Gemünden H.G., T. Ritter, Inter-organisational Relationships and Networks, Journal of Business Research, 2001



# *ImprovAbility*<sup>™</sup> at the Project Level

*Anna Høeberg & Klaus Olsen  
ATP Denmark, Kongens Vaenge 8, DK3400 Hilleroed  
ahb@atp.dk & klo@atp.dk*

## **Abstract**

More than 50% of all SPI-initiative are failing. The project Talent@IT has developed the *ImprovAbility*<sup>™</sup> model in order to improve the organization and each project. ATP has participated in development of the model and has carried out pilot testing and project assessment afterwards.

Project assessments can be used to improve the project early in the projects lifecycle by identifying the most critical parameters for the success of the project. This will give the project manager the opportunity to adjust the direction in time.

The purpose of this paper is to share our experience in using project assessments. ATP has with success carried out project assessments and expects benefit by using project assessments and will in the light of these recommend using of the *ImprovAbility*<sup>™</sup>. In this paper you will also find examples from the project assessments, we have carried out at ATP.

We will present our top 10 list of recommendation in implementation and in using the *ImprovAbility*<sup>™</sup> and how we have implemented project assessment in the process model we use at ATP.

## **Keywords**

Improvability, project assessments, improvement, SPI, practical experience.



### 1 The *ImprovAbility*<sup>TM</sup> Model

The *ImprovAbility*<sup>TM</sup> Model is developed in the Talent@IT project during a 3 years period. Atp participated in the project together with 3 other companies, the It University Copenhagen and DELTA. The results of the project were a model for assessment at the organisation level and a model for assessment at the project level. It is only the model for the project level, we will address in this article.

The research methodology was a quality interview study with more than 50 interviewees representing 16 projects at the four companies.

The findings from the interviews were grouped into 20 parameters at the organization level and only 17 parameters at the project level. The assumption behind the model is that these parameters influences on success or failure.

For more information about the project and the model see [1], [2], [3].

### 2 *ImprovAbility*<sup>TM</sup> at the Project Level

In this section we will give a short introduction to the *ImprovAbility*<sup>TM</sup> assessment at the project level.

*ImprovAbility*<sup>TM</sup> at a project level is concluded in the 4 following steps:

- Preparatory meeting
- Project interview
- Scoring and generation of recommendations
- Closing session

See figure 1.

*ImprovAbility*<sup>TM</sup> at a project level is done by project participants and 2 assessors.

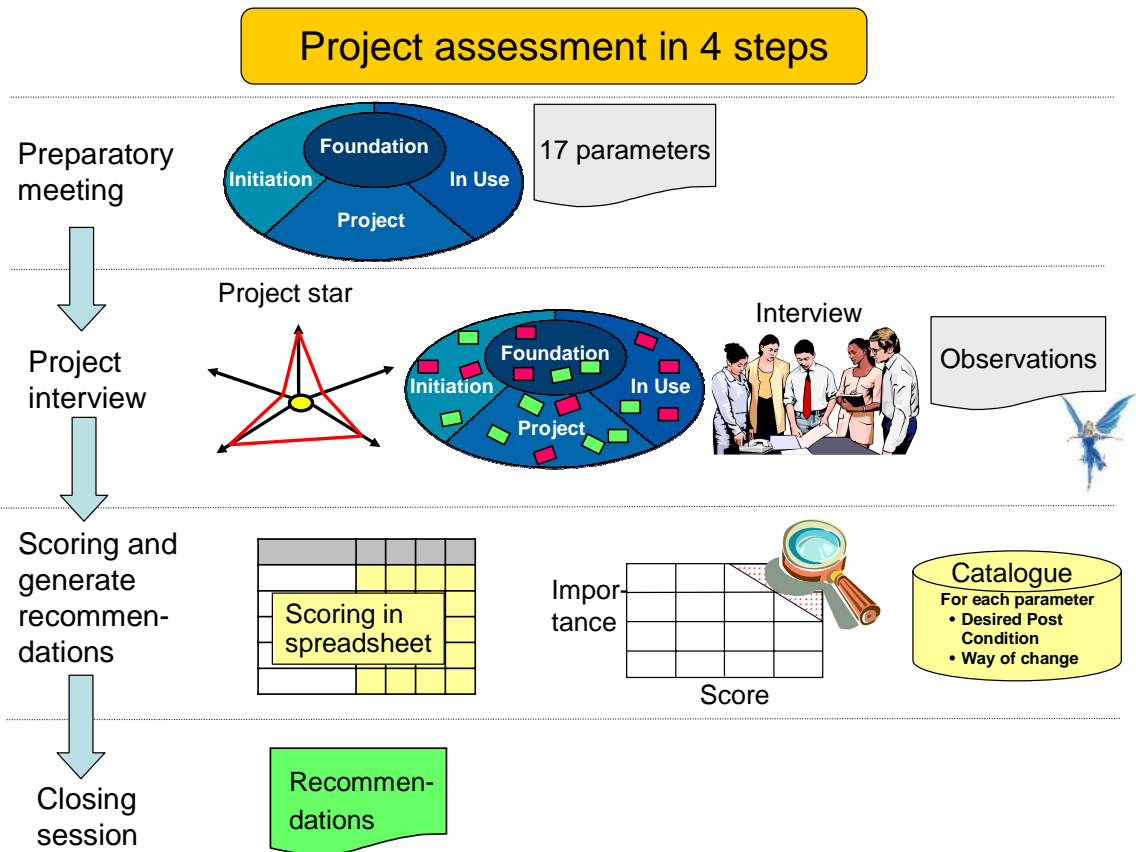


Figure 1 Project assessment in 4 steps

## 2.1 Preparatory meeting

A presentation template exists for use at the preparatory meeting. Included is the purpose of the assessment and a walk through of the 4 steps.

In addition the *ImprovAbility*<sup>TM</sup> model is presented with the 17 parameters in four groups (figure 2). Two pages describing the parameters are handed out.

A preparatory meeting takes half an hour.

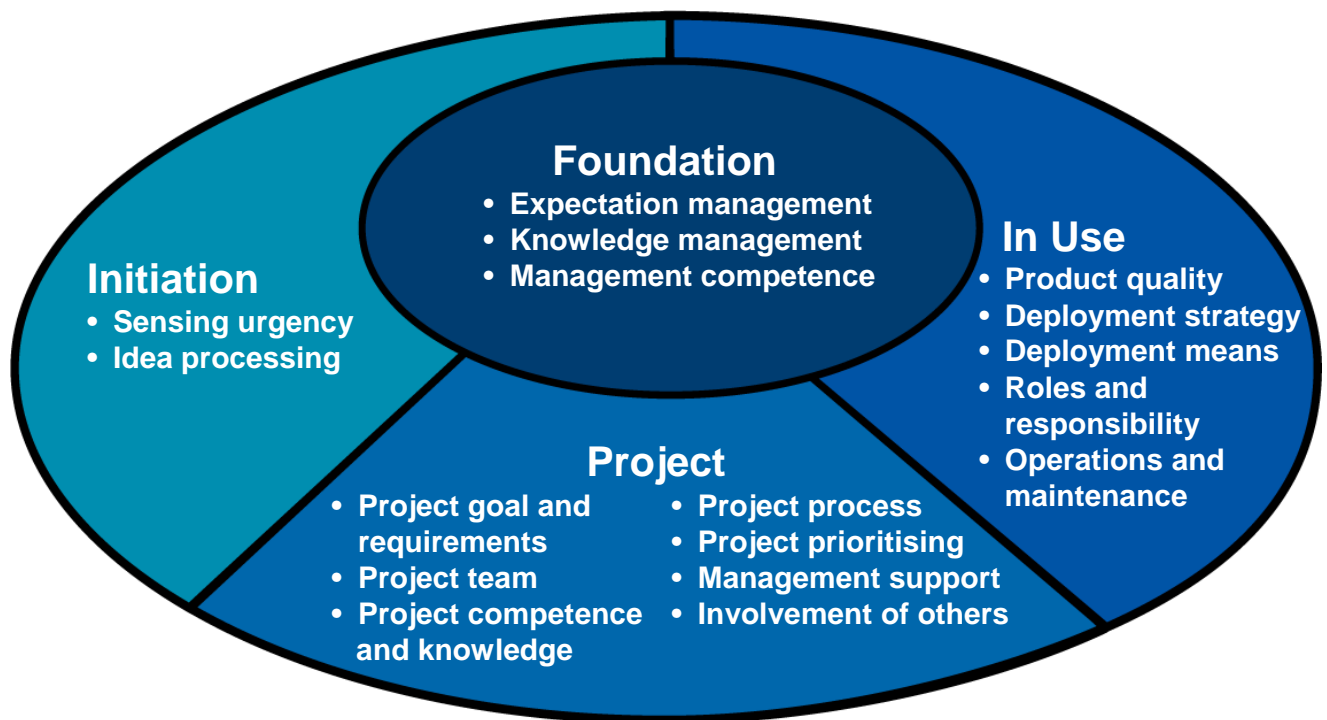


Figure 2 *ImprovAbility*™ with 17 parameters in 4 groups

## 2.2 Project interview

The project interview begins with a characterisation of the project with the help of questions in a questionnaire. The answers create the basis for the project star. The star will be used to determine within which parameters there can be generated recommendations.

Afterwards the project participants fill out 4 red post-it notes with the parameters they are concerned about and 4 green post-it notes with the parameters they feel they have complete control over. The post-it notes are put up on a board with the *ImprovAbility*™ model and are placed within the respective group so that the notes concerning the same parameter are together.

The assessors use the post-it notes during the interview by taking the post-it notes for a single parameter one at a time and then ask about the respective parameter.

The assessors also have an interview guide with questions that ensure that the assessors are able to answer all the questions to the parameters in the scoring sheet.

In the course of the interview the assessors make notes of the statements and observations.

At the end of the interview the assessors ask the project participants which parameters (one or two) they would choose if a Wishing Fairy would be able to fulfil their wishes.

A project interview takes 3 to 4 hours.

### 2.3 Scoring and generation of recommendations

Immediately after the project interview the assessors fill out the scoring spreadsheet supported by the noted statements and observations.

The result of the scoring is compared with the project characteristics in the project star. It is now possible with the help of the prioritisation matrix to identify the parameters that will create the basis for the recommendations.

To set up the recommendations a catalogue is used. The catalogue is used to find suggestions for initiatives that the project can carry out. In the catalogue you will find the Desired Post Condition and the Ways of Changes for each individual parameter. For each parameter the Ways of Changes will create the basis for the recommendations.

The assessors prepare the closing presentation by using the presentation template.

The scoring and generation of recommendations takes 3 hours.

#### Closing session

The closing session is held as quickly as possible after the interview by using the closing presentation.

At the meeting the assessors give their feedback based on what they have observed and present recommendations concerning 3 to 4 of the parameters.

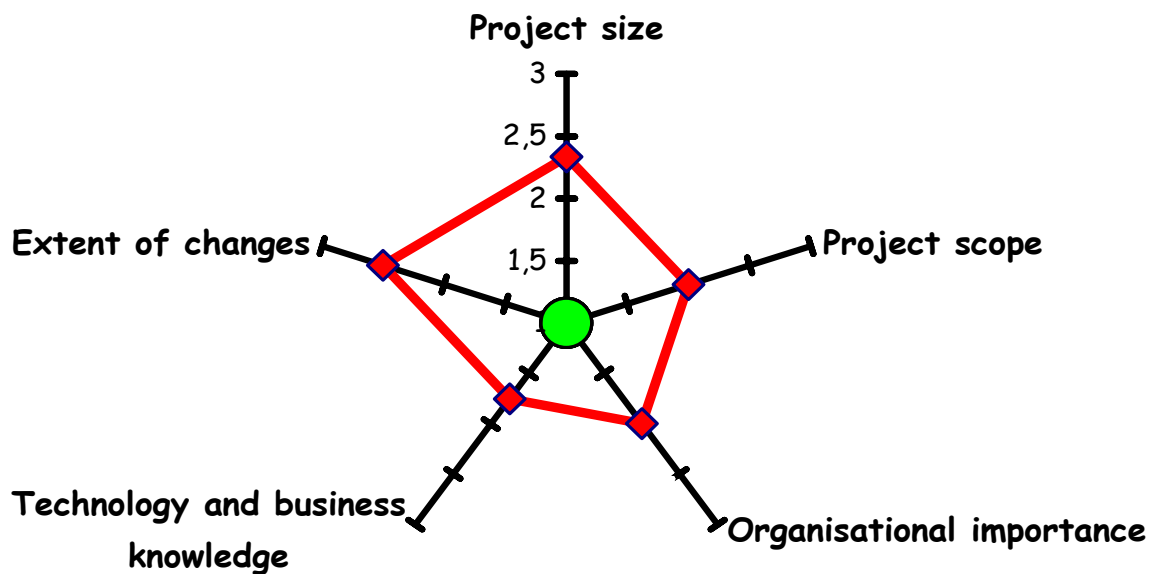
The closing session takes about 1 hour.

## 3 Completions of one assessment at the project level

### 3.1 Presentation of characteristics of the project

The presentation of the project is based on the project star.

The project is a SPI project with the aim to implement automated test at ATP. The project started in October 2005 and was completed July 2006. The assessment was held in January 2006. At this stage the project was manned with 5 resources and approximately 20% of the budget was used.



**Figur 3: The project star for the project**

By looking at the project star for the SPI project in figure 3 it is evident that:

- The project size is estimated to about 2.5. The project is manned with 3 full-time resources and will last for 10 month.
- The project's scope is estimated to 2. This is explained by the fact that only few persons will be involved in automated testing.
- The organisational importance is 2. The project is part of a strategic programme at ATP, but the sponsor is not placed at the top of the management.
- Technology and business knowledge is about 1.7. The technology is well known for the project participants and they know a lot about testing methods.
- The extent of changes is set to 2.5 because automated testing is new at ATP.

### 3.2 Project lessons learned

A project assessment gives a project the possibility to stop and reflect and spend some time evaluating the progress of the project. The projects believe that it was positive that the project assessment forced them "to stop" and it gave them time to reflect.

The recommendations for the projects were not surprising. The scoring and recommendations were created on the basis of input from the projects and therefore the assessors have only held up a mirror for the projects. We think that the exercise with the red and green post-it notes helps the projects to focus on their own strengths and weaknesses. The whole assessment can work as a catalyst towards an acknowledgement of what is necessary to counter the weaknesses in the project.

A project assessment can give good feedback to the project manager.

The summary of the projects' lessons learned gives input to the recommendations and implementations in ATP which we have described in section 4.

### 4 Experiences and recommendations

#### 4.1 Experiences with project assessment

In this section ATP's experiences with ImprovAbility™ project assessments will be described, taking reference in the following activities:

1. Preparatory meeting
2. Project interview
3. Scoring and generation of recommendations
4. Closing session

##### **Preparatory meeting**

The participants at the preparatory meeting were the same people who took part in the project interviews. It proved to be good ballast for the participants in the following interview and the concluding meeting. The participants received a detailed introduction to ImprovAbility™ and the course of events for the project assessment was outlined. Therefore, they were fully knowledgeable and knew why they had been invited and what they could expect from the course of events.

##### **Project interview**

There were allocated three hours for each project interview. The time allocated proved to be adequate for the interviews.

Our experience was that the projects were open, honest and optimistic. For example they found it easier to point out the parameters they felt they had control over (the green post-it notes) than the parameters they thought would give problems (the red post-it notes). They had to predict the future for parameters starting later at the project lifecycle.

The project star was not used in the interviews but in the scoring afterwards. However, the creation of the project star gave the assessors a good understanding of the projects which were beneficial in the interviews. This knowledge made it possible to place the projects answer in the context.

The red and green post-it notes were a good starting point for the interview. It was completed as an open dialogue. As assessors we used the interview guide to ensure that all of the questions were discussed and the answers therefore gave a good base for the scores and recommendations.

##### **Scoring and generation of recommendations**

The scoring for both projects was done immediately after the interviews.

The Wished State and the Change Catalogue gave inspiration to generation of recommendations. The internal assessor's knowledge ensured that the recommendations were made concrete with respect to ATP. This made the recommendations operational and usable for the projects.

##### **Closing session**

We started with general observations about the project we had experienced in the interviews but which were not necessarily included in the scoring. For us as assessors it was important to accentuate the positive observations.

In the concluding presentation the assessors stressed that the choice of the 3 weak parameters was based on the projects own statements. None of the projects were surprised as to where they had weaknesses. It should also be noted that for all projects their wishes were amongst the parameters that had the lowest score.

##### **General**

We experienced that the model was well qualified in connection with both the SPI project and the IT development project.

### 4.2 Top 10 recommendations

From our first test of the project assessments we have drawn out the following recommendations. It should be noted that the recommendations at this stage are based solely on 3 project assessments.

Recommendations:

1. **Project interview:** Complete the project interview with 2 educated and competent assessors, who change roles during the assessment (interview and writing notes).
2. **Preparatory meeting:** Always hold a preparatory meeting with the project. Participants there will take part in the project assessment need to take part in the preparatory meeting. Project participants that have not taken part in the preparatory meeting don't have sufficiently knowledge to be able to take part in the assessment as they amongst other things will not know about the 17 parameters and the *ImprovAbility*<sup>TM</sup> model. Use half an hour for the meeting and hand out a short description of the 17 parameters.
3. **Project interview – dialogue:** Create a good dialogue with the possibility to give detailed answers.
4. **Scoring:** Do a quick scoring immediately after the project interview. Ensure that you have approximately an hour to do the scoring. It is a good idea to do the scoring immediately after the interview as you have an overall picture of the project.
5. **Recommendations and closing presentation:** Should be made as soon as possible after the project interview. Ensure approximately 3 hours for this work. Use the standard template.
6. **Closing session – quick feedback:** Should be held as soon as possible after the project interview. Preferably within 2 days after. This ensures the project has a continuous course and it works well with quick feedback to the project.
7. **Closing session – the mirror:** Create a good atmosphere. Explain that the assessors are holding up a mirror for the project. Ensure that there is no criticism of the project and outline what the project sees as its weaknesses as the basis for the recommendations.
8. **Closing session – leader's participations:** It is a good idea to invite the nearest leader to the closing session. However, if they have not participated in the project interview, they should only listen and not ask questions.
9. **Report on handling of the recommendations:** The project manager ought to report and explain how the project will use the recommendations within a fortnight.
10. **Use a checklist:** Prepare a checklist with all activities during the assessment to make sure, that for example sufficient time is reserved for the closing sessions and preparation time for the assessors,

### 4.3 Implementation at ATP

Project assessment has been included in the Project method adjustment in ATP's IT Process model. The discipline "Project Method Adjustment" is completed in the beginning of a project. It is stated which processes that need to be completed and the artefacts that need to be created. The project assessment needs to be used to ensure the right focus on the correct initiatives.

We will to a large extent use internal assessors because they can give recommendations that are based on our development model. We will recommend that both the internal and the external assessors have the following competencies:

- Expert in the *ImprovAbility*<sup>TM</sup> model, including being confident with the concepts and toolbox that the assessors have available.
- Interview technique
- Presentations technique
- Empathy: ability to put yourself in others situation, instinctive understanding and intuitive perception.
- Objective; neutral and without prejudices
- Clear thinking; sensible and well informed

Completion of the assessor education course is a necessity. The plan is that newly educated assessors complete the first 2 assessments together with an experienced assessor. We will suggest cooperation with other firms which will benefit both firms as we can use their assessors as external assessors and they can use ours. This will provide experience external inspiration.



### 5 Literature

1. Vinter, Otto & Pries-Heje, Jan: På vej mod at blive bedre til at blive bedre. DELTA 2004. ISBN 87-7398-139-7.
2. Pries-Heje and Johansen, Jørn: AIM – Ability Improvement Model. Proceedings from 12<sup>th</sup> European Conference, EuroSPI 2005 Budapest, Hungary, November 9-11, 2005. ISBN 3-540-30286-7.
3. Pries-Heje and Johansen, Jørn: Success with improvement – Requires the right roles to be enacted – in symbiosis. Proceedings from EuroSPI 2006 Joensuu, Finland, October 11-13 2006.

### 6 *Author CVs*

#### **Anna Høeberg**

IT-architect at ATP. B. Com. (Org.) from Copenhagen Business School and Master of Information Technology from the IT-University. Have worked with IT for various companies for more than 15 years. Major competences include development, project management, quality assurance and software process improvement.

#### **Klaus Olsen**

IT-architect at ATP. B. Com. (Org.) from Copenhagen Business School. Have worked with IT for various companies for more than 25 years. Major competences include development, project management, quality assurance and software process improvement.



# ***ImprovAbility***<sup>TM</sup> in SPI projects at PBS

*Arne Staal Ibsen*

## **Abstract**

***This article describes two software process improvement projects in PBS (Payment Business Services) - a Danish service provider. Based on the experiences of the two projects we deliver a first cut evaluation of the factors that have proven to be significant for the success of improvement projects at PBS. The two projects were implemented and evaluated prior to an ImprovAbility<sup>TM</sup> (improvement of abilities) evaluation of PBS.***

***The article describes the initiation, progress and implementation of a project that was initiated with the purpose of improving the test process at PBS. This project is considered a success and the success factors are analysed and described. Parallel to this project another project concerned with the improvement of the requirement phase was completed. The problems and obstacles of this project are described, and the project is evaluated against the test project to give a first cut evaluation of the factors that are critical to secure the success of a software improvement process at PBS.***

***To evaluate the findings of the ImprovAbility<sup>TM</sup> model we have modelled the ImprovAbility<sup>TM</sup> recommendations on the findings of the original evaluation. The comparison shows a remarkable resemblance between the original advice and the findings of the ImprovAbility<sup>TM</sup> evaluation. The comparison indicates that the ImprovAbility<sup>TM</sup> model is valid for the improvement of project success.***

## **Keywords**

Software Process Improvement, Ability

### **1 The company**

The objective of PBS is to develop, sell and operate systems and services in the market for payment cards and payment services, as well as to provide services as a subcontractor to the common infrastructure of the banks. PBS Holding A/S is owned by Danish banks and the Central bank of Denmark. Net turnover in 2004 was €230 million and result for the year €23 million.

In popular terms, PBS moves money and information between citizens, companies and public institutions. In 2004 1.3 billion transactions were processed via PBS. PBS was founded in 1968 and employs approx. 800 people. Our headquarters are located in Ballerup, north of Copenhagen.

PBS is organised in 3 business divisions Payment cards, Payment services and New Business Areas and one IT division. The IT division with the CEO as the top manager employs 400 people. The IT division is responsible for the development, the maintenance and the operation of the business systems. New business systems and major changes to existing systems are processed by 130 developers and project managers. Project size varies from small projects (three to four people involved) to larger ones (more than fifty employees involved). In total more than fifty projects will be developed and implemented each year. The business divisions define and prioritize the development plan.

### **2 The challenge**

Prior to the improvement project the test process in PBS was characterised as a described process with mutual templates and forms. The process and templates had been developed years earlier, and the developers found that the process although structured did not resemble the test process which took place in the actual development processes. Facts and fiction were a major issue. The people in charge of maintaining the process description were in fact alienated from the actual development process. Although the process was structured and well described, there was no adherence to follow the process among the developers. The developers found the definitions unclear, the process description unfit for projects and the forms useless.

The challenge was to build a new process with new templates that would meet the developers' needs. Consensus regarding test concepts was the first major issue. Concepts like test specification, test scripts and test environment were used in the model but not synonymously defined, and subsequently not used and documented in projects and even worse not used in projects in the same domain. Each project and each domain had their own definitions and templates if any.

Prior to the test improvement project PBS consisted of three different business areas: card processing, payment and payroll systems. Although there had been many fission related activities each business area still had its own characteristics and ways of developing and implementing new business functions. This was also the truth in relation to the test process. The description of the test process was at most a way of describing the best possible match between these three cultures. But all in all no common process existed; each area had its own definitions and processes.

Furthermore, each area had its own tools depending on platform, environment and development toolset. There was no common tool for managing of the process (test strategy, scripts and criteria, execution and default checking). The process was managed through templates created in Microsoft Word – a tool unfit for managing numerous test scripts and regression tests.

No formal education or learning took place in relation to the test process. A one-day session where the whole life cycle model was presented was the only training available. No process description, no role presentation and no tool education were present. Project members would pick up knowledge from each other and from working in different projects – and that was the only way knowledge was spread. Due to this way of spreading knowledge the test competence was uncoordinated and dependant on

personal behaviour and skills.

### **3 Project initiation and organization**

Due to the lack of a common supported test process the PBS management, project leaders and developers realized that a common process and shared test tools designed to manage and execute tests were necessary. Software process improvement issues were continuously debated in management and among project leaders in a structured manner and the test process improvement was encouraged and prioritized.

A software process improvement project in PBS is managed in the same way as business projects. As is the case with business projects software process improvement projects will be prioritized and a project leader and project members will be appointed full or part time. The project leader reports to the Head of Development and a project charter and project planning will be prepared. Scope, activities and deliverances are negotiated with all interest groups i.e. project leaders, project members, test groups, maintenance and other process improvement groups and projects.

The target group consisted of project leaders, developers and testers within the development unit. The Head of Development made the decision to initiate the project in adherence with all project leaders. The authority of the Head of Development included all target groups. This was considered one of the primary factors of success.

Project members were appointed according to proven skills in testing, commitment and the fact that they enjoyed a good reputation in testing.

The project leader had experience in process improvement projects and a known ability to create consensus among different groups and opinions.

The project group consisted of two project leaders from successfully tested business projects in two of the business areas and three testers recognized for their abilities in testing. The three testers fulfilled the need for representation from all business areas.

Finally, the project attached a person from the maintenance group (software improvement) with skills in project leadership and testing and a consultant from Delta (Talent@IT partner) whose main purpose was to secure that the coming testing process adhered to IEEE standards and that is was state of the art.

### **4 The test project**

The purpose of the project was twofold: First, to develop a new common process description and second, to select and implement a proper tool for managing the test process.

#### **4.1 Project initiation**

Two test management tools had been analyzed and installed in two different business projects prior to the improvement project. No evaluation reports were made but both projects gave positive feedback related to the use of these test management tools. Without any further analysis the test project manager chose one of the tools and obtained the management's approval. The only reason why this tool was chosen was due to the project members' prior experience with it. This was not the most structured and valid way of selecting a tool but gave new enthusiasm and facilitated the implementation and adoption of the test tool.

A structured pilot test was initiated in a live business project to secure the implementation of the chosen test management tool. Three members of the SPI test project were also members of the business project and this overlap gave the possibility of implementing the test tool in conjunction with

the development of the test process and templates.

A series of workshops was introduced to develop the new, enhanced test process. All project members took part in these workshops. It was soon realised that test concepts and definitions were essential to consensus. Without a proper understanding and mutual agreement on this, an agreed test management process was not an option.

In the workshops much time was spent on test concepts i.e. what is a test script, a test strategy, what do these concepts involve, how can we map them against IEEE definitions, how does the different understandings in the three business areas match these concepts. A data model for test concepts with objects (concepts) and relations was defined and developed. The data model was matched to the original process model, to the different practises in the business areas and finally to the international standards. With this thorough debate and analysis the discussion of concepts, which had lasted for many years among the participants, finally came to an end. Consensus was imminent.

Even though the project members agreed on the test concepts and the fact that they came from different departments in the organisation did not give any assurance that consensus would reign throughout the whole community of testers and project leaders. To guarantee this broader consensus and to secure a proper adoption of the future results the SPI project initiated the formation of a test forum at PBS. With a representative from management in charge a test forum ("Test – now and in the future") was established. The test forum included all developers who took an interest in testing. Twenty people participated. The test forum now became the decisive test for the SPI project. If consensus could be achieved in the test forum one imperative factor was obtained to secure the success of the SPI project. The test concepts and definitions were introduced to the test forum and with minor improvements the concepts were agreed upon.

### 4.2 Project development

With this agreement the SPI project initiated the development of the test process and its different activities. Three project members (the project leader, the representative from life cycle maintenance and the Delta consultant) acted as the kernel group that was assigned to structure and describe the new test process. On a continuous basis, propositions were put forward to the SPI project. The propositions were debated, enhanced and approved in the SPI project prior to the final evaluation and agreement in the test forum.

The discussion and hearing process secured a thorough treatment of the propositions and a mutually accepted test process description. Fruitless discussions and misinterpretations in projects and among projects in the same business area concerning test concepts and test processes were history. Scarce project time could now be used on test issues directly connected to the business demands and customers due to a mutual consensus on how the test process should be completed.

Parallel to the development and approval of the test process description the first experiences came from the use of the test management tool. The experiences showed that the tool had the demanded functionality and usability and matched the new test procedure well. The pilot test of the test management tool was expanded to two other business projects from different business areas. The new test concepts and test processes were matched to the tool and controlled adjustments were made in both tool use and process descriptions. The human "overlap" between the development of the test process and the tool adaptation made this process possible and created only a few minor easily solvable challenges.

The new test process describes:

- how to develop a test strategy,
- how to make a test agreement with the customers and subcontractors,
- how to arrange the test planning process (test specifications and scripts, establishment of test environments, creation of basic data and input data and test preparation)
- how to accomplish the test

- error handling and finally
- how to write the concluding test report

These processes that were formerly handled and documented in word templates are now managed in the test management tool i.e. planning, accomplishment and error handling processes are now managed and for the greater part documented by using the tool (the test strategy is still documented in MS Word).

The common process can now be recognized from project to project and from business area to business area. Progress can be measured immediately by the number of test cases, test scripts, tests, errors found, errors solved etc.

The fact that a (test) process is managed and documented through a well-defined, usable and structured tool assures that the process is repeated from project to project. Figuratively speaking: How many people insist on breaking the syntax rules in a programming language for a longer time period? Former use of different test concepts and processes is precluded and measurement and use of testers in different projects improved.

### 4.3 Deployment

With generally accepted concepts and processes and a test management tool to handle the processes the deployment phase could be initiated. An implementation plan was developed with marketing, information meetings and educational and training activities. Responsibilities and roles were appointed.

Former experiences in PBS speak for an education lead by instructors from PBS. The instructors were chosen among the testers in the organisation (testers that had been part of the project and / or deeply involved in the test forum) supplemented by the person involved from the maintenance group (software improvement). It was a deliberate choice to use participants with test skills from the business development project as instructors as this also guaranteed diffusion and adoption.

In the deployment plan it was supposed that the adoption could be improved by combining tool education with process handling, hands-on experience and a minor degree of competition among the participants. The educational plan consisted of a two day workshop where test management tools and test process adoption were combined to secure the cohesion of the two parts of the test process. The hands-on experience was not a challenge either: The process was adopted through use and documentation of business related examples in the test management tool.

The competition part, however, was a greater challenge. The DELTA consultant attached to the test project had good experience with the use of a board game for educational purposes. By adjusting this game to the testing process, the element of competition was introduced in the workshop. Lessons, hand-on experience and a little competition were mingled throughout the two workshops. The feedback from the participants showed that this combination was an absolute success.

The workshop is now an established part of the education given to new employees in the development department. And all current staff has participated with great success.

In the same period the maintenance of the test management tool environment has been improved. A small group of employees has been assigned to maintain the product (implementation of new versions, user roles, identification and authorization, backup, surveillance etc.). The use of the tool is now continuously updated to fulfil the needs of the business projects and the tool maintenance group is constantly challenging the provider to introduce new enhancements.

Another succeeding activity has been to investigate the possibilities of using the tool for regression test purposes in the different business areas. A test template has been build for each business area to secure the use of predefined requirements and test cases. Especially in the card processing area improvements have been made and test scripts are being used for regression test in existing and future projects.



### 4.4 Results and status

Status for the deployment of the new test process and test management tool is that all new projects use the test process description and templates including the management tool. The use of the test management tool is mandatory for all new projects. Dispensation for use can be given by management only.

All developers have participated in the basic educational program for the test process and tool.

No measures have been made to evaluate the developer's anticipation of the new process and tool but the general impression is a positive feedback from the users. The only problem we have experienced so far is the lack of a user friendly report facility in the test management tool. The tool maintenance group has addressed this problem to the provider.

To sum up the factors that seem to have contributed to a successful diffusion and adoption of the new test processes and test management tool are the following:

1. **Accept and support from management.** The management (the Head of the Department, the line managers and the project leaders) prioritized test as a major concern for improvement. Continuous accept and support from management in the form of resources (time and money) and participation (management in charge of the test forum, mitigation through departmental meetings and assemblies) is vital.
2. **All participants were subordinated one departmental head** (project leaders, project members, the SPI maintenance group, testers etc. all referred to the departmental head). The SPI project was a project organised and contained within the same limits as ordinary business projects. The project management process was well known to all participants and interest groups. Similarly, the participants and the interest groups were part of the same organisational grouping – disagreement between different departmental heads concerning project prioritisation and goals would never be an option.
3. **User need decisive for project initiation.** The developers and testers expressed a need for improvement in the test area. Lack of consensus and ever lasting debates in projects on how to accomplish the testing process was a good starting point for improving the process and prioritising the project.
4. **The improvement area was familiar to the users.** The activities and organisational aspects of the test process were known and used, and the changes to the area were not fundamental.
5. **Trend setting users were part of the development process.** The fact that users with a good reputation in testing were involved from the beginning and had critical influence on concepts, process and tool acquisition was a major reason for the overall acceptance. User involvement in project definition and development contributed to a successful deployment.
6. **Involvement of the user community through meetings.** The established test forum had a similar effect on the universal acceptance. Although many users did not have direct influence on the specification of processes and tool selection they had the opportunity to address the project related issues to the test forum on a continuous basis.
7. **State of the art.** The fact that consultants were involved gave confidence to the user community and management that the activities to be deployed would be state of the art and comply with international standards.
8. **The crucial role of the user in the deployment activities.** The user community was educated by their own group participants. Alignment and trustworthiness are enhanced through the use of super users. Proven success in testing practises among the instructors along with a recommendation of a specific way of doing things was bound to be an example to be followed.
9. **Educational and training activities close to practise.** The use of practitioners from the user community in the educational process and the use of illustrations from the different business areas assured that the learning process was as close to practise as possible.
10. **Tool support for process implementation.** New processes and new ways of performing known activities will be maintained through the mandatory use of the tool. The learning process is

improved through tools which help users institutionalise new practises. The learning process is facilitated by predefined templates, established forms, and automated processes. Comfort in relation to new skills is improved.

11. **Tool securing the repeated process.** A tool that demands established processes and sequences safeguards the actual accomplishment and ensures that the process is repeated from project to project and from business area to business area. Economical and valid measurements are possible and improvements can be made and accomplished.
12. **Process ownership.** Participation and influence in defining the process is one part of ownership, another crucial factor is the influence that project members have on the actual implementation in their own business projects. The process description is general; the actual implementation in projects has been defined as a project activity. The project leader, testers and developers obtain the ownership of the process by using and interpreting the process.
13. **Keeping up momentum.** An improvement process has, like any other project, a start and a finish. When the project is completed it is important that the organisation keeps track of the process and keeps renewing the accomplishments. Like a business project that changes, business processes will deliver its software programmes to maintenance for continuous improvement (functions and production). An improvement project needs a maintenance group to improve the new process when experience shows that the original idea needs new enhancements (error handling, continuous education, problem handling, minor changes and propositions of improvement).

A generalisation from these subjective findings is dubious. To state that findings in one organisation promote diffusion and deployment does not indicate that these findings will work in another organisation. Parameters like company culture and degree of maturity (capability maturity) might influence the recommendations concerning activities and actions that facilitate deployment.

In relation to capability maturity models (CMMI, Bootstrap) PBS is on level 2 i.e. between the repeatable and the defined level. The processes are described and carried out in every project, but not to a level that variations in time and money are insignificant. The project leader is in absolute control, he or she is still vital in relation to the utilisation of processes in projects. General measurements are rare and process improvement efforts not totally centralized.

Processes are planned and carried out with the project leaders as the main contributor and decision maker. This indicates that improvement projects that fulfil project needs (the project leader) have a greater probability of success than improvement project that fulfil organisational needs only. Parameters for success should be close to the needs of the project members and leaders when we are dealing with companies at this maturity level.

The cultural factor may also have an impact on the success of different ways of improvement. In a culture that de-emphasizes the differences between citizens' power and wealth and where individualism reigns implementing common values has a greater probability of success than rules and sanctions imposed by management. Investments in participation and involvement seem to be crucial to the fulfilment of company goals.

### **5 A parallel project**

In the same period another improvement project was initiated and developed. This projects goal was to improve the requirement process. At this point the requirement process affected two different departments: The development area and the business area.

The decision process, the project initiation and the project development were practically identical to the processes carried out in the test improvement project. The need for improvement was expressed by the Head of Development, the project leaders and the Head of Business. The developers and the employees in the business area were only partly involved in the decision making process.

Mutual workshops took place. The two department heads, managers from development and business, the software process maintenance group, a Delta consultant and a SPI project leader were present at these workshops. The workshops described the goals, the requirements and the structure of the future

requirement process and the improvement project. After the workshops, a small group was established to fulfil structure and the specification of the improved requirement process.

To assure a unique borderline between business and development the workshop group prepared a specification of the requirement process including levels of requirements (business, user and technical requirements). A number of requirement types should be defined. These requirement types should act as a checklist for development projects and, finally, a specification of the requirement process and matching templates should be developed and described.

The process description, checklists, templates and techniques were elaborated and reviewed. Education and other deployment activities were prepared and the new requirement process was presented to the users in an educational program similar to the test process.

In spite of this almost identical course of action in terms of project initiation, development and deployment, the requirement process was only a partial success in relation to diffusion and adoption. Insecurity is still present in relation to the requirement process and the adoption is only partial in the business areas and projects. Whereas the degree of adoption and diffusion is high in the test process the degree concerning the requirement process is considerably lower.

### **6 Two process improvement projects – why different success rates?**

To deduce the factors that improve the success in software process improvement projects in an organisation like PBS (size, maturity and culture) it seems important to describe the differences that characterise the two projects.

The below factors describe the differences and they are considered to be significant to the success rate in software process improvement projects in PBS. Differences in staffing, accomplishment of project activities, review processes, number of involved users etc. are not taken into account. This may be considered problematic but has not been part of this research.

- **Different goals.** The organisational context was more complex in the requirement project. There were two different areas within the same organisation with different overall roles and goals – one acting as a customer (business) the other as a provider (development). The overall goals in these areas may not be the same neither in management nor among the employees.
- **Organisational changes.** During the development of the requirement project a major organisational change took place in the business area. The business area was divided into three different departments (card processing, payment and payroll systems and new business areas). The project failed to fully integrate the management of all the new business areas. The two new lines of business / departmental heads had not previously been part of the past prioritisation and did not express the same need.
- **Two different divisions involved.** Unlike the test process the requirement process specifically involves more than one organisational unit. The need for improvement of the requirement process had its origin in a disagreement between the business area and the development area. Different views on the other part's competence and the scope of each area's role in the process were widespread. In spite of the consensus between the two areas with regard to their role in the requirement process that seemed to have been achieved in the course of the workshop, it became evident that the original disagreements were still present in the organisation. A software improvement project needs unity among sponsors and a high degree of consensus concerning goals and roles.
- **No consensus but new unknown practise.** In the test improvement process a shared experience was the basis for the definition and structure of the process. The project assignment was to extract and concretise experiences. In the requirement improvement project the assignment was a completely new process where the two divisions had different opinions concerning project goals, role definitions, division of labour etc.

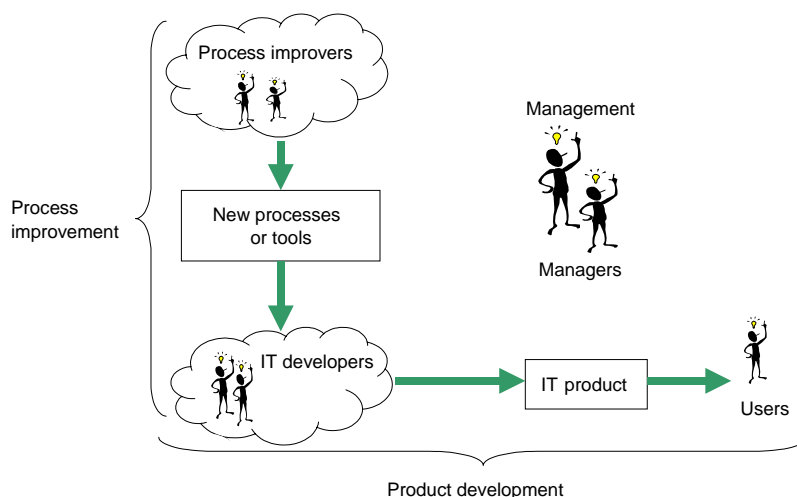
- **No tool support.** One of the goals of the test project was to implement an automated test tool. This type of automation was not part of the requirement project. No tool support for the adoption of the new process was present or intended. The processes were manual and the only support was through MS Word templates (at present tool support is being evaluated to improve the adoption).
- **No imminent user need.** Initiation of the requirement improvement process was basically a need within management. A need justified by the lack of a defined and agreed borderline between the user community and development. Basically, the need of the employee is not pronounced due to the fact that cross-organisational processes are not in focus among individual employees. Success among individuals is rather due to accomplishments within the organisational unit and its goals.
- **Revolution not transformation.** The requirement improvement project consisted of a new context, new borderlines, new roles, new competences, new processes, new comprehensions etc. The test improvement project was based on building consensus through already used practises.

### 7 The ImprovAbility™ organization assessment at PBS

As part of the Talent@IT project, PBS volunteered to participate in an *ImprovAbility*™ pilot study in the Spring of 2005. The purpose of the study was to evaluate the ability to improve both product development and process improvement.

#### 7.1 Selecting the interviewees for the assessment

The actors in an *ImprovAbility*™ assessment are shown in Fig. 6.1. The IT developers develop IT products for the end users and the process improvers develop new processes, methods and tools to be used by the IT developers. The management initiates, prioritises, supports, follows-up and sometimes terminates projects.



**Figure 1: The actors in ImprovAbility™**

In an organisation assessment 3 different groups will be interviewed

- The management
- The projects
- The users of the products that are created in the projects

The projects can be either IT development or process improvement projects or both depending on the focus of the assessment. The selection of projects and users depends on the focus. If the focus is on IT development, the projects must be IT development projects and the users will be the end users of the type of products that the IT developers produce. If the focus is on process improvement, the projects selected for the assessment must be process improvement projects such as an improved requirement process, a new test process, or the selection and deployment of a new test tool. In this case the users will be the IT developers. For a single focus (IT development or process improvement) in a medium sized company, the number of interviews will typically be:

- 1 management interview
- 2 – 4 project interviews
- 1 – 2 user group interviews

Designing the management interview is fairly straightforward. As vision and strategy is on the agenda, the top management must be included. IT management and other managers with influence on and interest in the projects being assessed must also be present. The level of management can span from the level higher than the project managers to the very top.

It is important to cover the diversity of projects in the organisation as much as possible. Having perhaps only two project interviews at your disposal, it is important to include three or more projects in the two interviews. For an interview to run smoothly and provide information of sufficient diversity, the number of interviewees must be from 3 to 7. Normally, it is optimal to include the entire project team in the interview, again for the sake of diversity but also to respect all team members' contributions. For small projects it is hence possible to include two or more projects in a single interview. For larger projects a full interview with focus on only that one project may be needed. The number of project interviews is decided and designed in accordance with these fundamental principles: 3 – 7 participants, maximum number of projects covered (at least 3), preferably include the entire project team and look for an opportunity to interview two similar projects simultaneously.

Designing the user interviews follows a similar pattern. The users must be users of the type of products that the interviewed projects are producing. It is not necessary that they are exactly the same products as long as they are of a similar type. Users are individuals and it is therefore easier to design a group of users covering a variety of products in a single interview. To get a balanced picture, however, it is necessary to interview at least two users of any product limiting the number of products to be covered in a single interview to three and in practice often only two.

In the PBS case 6 interviews were designed

- Management - CEO, Development department manager, SPI manager, staff manager, business relationship manager
- Software process improvement project (process of putting forward proposals) – project leader and project participants
- IT development project (Payment card) – project leader and project participants
- IT development project (Payment services) – project leader and project participants
- One user group covering the software process improvement project – managers and developers
- One user group covering Payment card projects – managers and professionals

## 7.2 The assessment setup

The assessment was the second pilot test of the model and the first with a model resembling the final

model described in this book.

The assessment was scheduled over a two-week period with the opening presentation at PBS in January 2005. The assessor team conducted all interviews in the following week. The assessor team was the scientific staff of the Talent@IT project i.e. the project manager, the scientific manager, two senior consultants and two Ph.D. students. The project manager and scientific manager are very experienced maturity model (e.g. BOOTSTRAP, CMM and CMMI) assessors where as the senior consultants are experienced SPI consultants for Danish companies.

All interviews were conducted by two persons and mainly for training and evaluation purposes the interview team consisted of one experienced assessor and a lesser experienced co-assessor (observer, trainee).

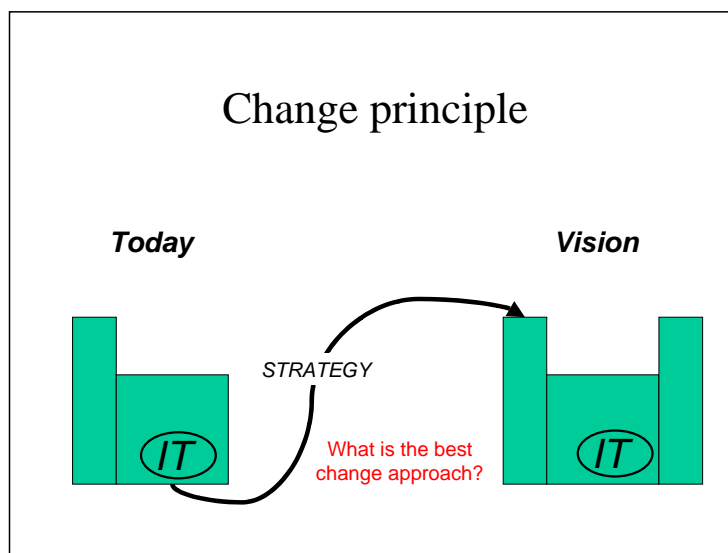
### 7.3 The findings in the assessment

Two sets of findings are accomplished in an ImprovAbility™ assessment: How to improve (the change principles) and what to improve (the parameters).

#### 7.3.1 The main change principles found in PBS

All assessments are based on the recognition that a change is needed. Fig. 6.2 shows a picture of the current state – the company as it is today. All companies have a vision or at least an idea of how the future ought to be - the vision of the company of tomorrow. The million dollar question is now “How do I go from today to the vision of tomorrow?”

Literature studies in the Talent@IT project revealed 10 different change approaches ranging from Business Process Reengineering (BPR) to “Optionality”. All 10 change approaches have their strengths and weaknesses – in some cases they work in other cases they do not. The task is to find the approach that is best suited for the specific company. Therefore, the ImprovAbility™ assessment includes a questionnaire that reveals the change approach that the assessed company should use in their process of improving their business.



**Figure 2: The change principle**

The primary change approaches found at PBS were **socializing** and **learning**. This means that in order for a change to take place and be rooted at PBS, the employees will have to learn it from their colleagues by socialising about it. Change should be driven by a focus on organizational learning, individual learning and what creates new attitudes and behaviour. Diffusion of innovation happens

through personal contacts rather than through plans and dictates. PBS are definitely not an organisation where BPR would be appropriate – the company are by no means in financial problems and do not need to throw away what has already been accomplished. But PBS do need to change in the very competitive market – not by revolution as suggested by BPR but just by training everybody to be a little better the whole time as the employees are in fact open to change.

### 7.3.2 *The parameters to be improved*

Parallel to the recommendation concerning change parameters an assessment of parameters was also part of the interviews at PBS. The recommendation related to two “In use” parameters: “Deployment means” and “Product quality” and one project parameter: “Involvement of others”.

**Deployment means** i.e. to what extent is the optimal mix of information, communication, education and training, plus marketing of new processes and products applied? The advice was to communicate often, early and adequately. The advice was to activate the users early in the process and to use socialization as the main mechanism. Information, involvement and training throughout the whole project and the implementation were the main issues. Adequate resources must be associated.

**Product quality** i.e. to what extent do new processes and products that are deployed comply with the following: high quality, few defects, user friendly, low complexity, compatible and efficient? Timeliness, functionality and non functional demands concerning security and operation are important and vital issues at PBS and were not at stake. The recommendations were more concerned with the involvement of the end users in PBS. They argued that their involvement through representatives were inadequate. They needed more information, more education and a system that was closer to their daily needs and working habits.

**Involvement of others** i.e. to what extent are other stakeholders (than the team and management) involved? This could for example include early user involvement at the right time and in the right way. A parameter in the project sphere but nonetheless very much alike the findings concerning product quality and deployment means. The recommendations dealt with workshops with the users during the early project phases and information, information and information...

## 8 *Comparison of the findings in the SPI evaluation and the ImprovAbility™ assessment*

How do the findings in the two SPI projects match the *ImprovAbility™* assessment? There seems to be a good correlation between the two sets of findings i.e. the evaluation reports from the projects and the *ImprovAbility™* interviews.

In the SPI requirement project the major obstacles of project success were:

- The project organization and the users in the business area had different goals. Involvement of the users (not just management) was lacking. More workshops and more information were needed.
- The fact that the user organization changed during the project was not taken into consideration. Lack of involvement of the management of the new user divisions was problematic. The expectation management process was absent.
- Two divisions involved. Again, a lack of involvement of others seems to be the problem and a mutual vision was absent (Vision and strategy).
- No consensus but new unknown practises. Too few means of deployment were present to implement a change that had a huge influence on daily life. The socializing processes in the user community were too weak for this major change. The users did not feel that the new requirement process and the associated means met their needs i.e. product quality.
- No tool support. Another crucial factor concerning deployment especially when the other deployment means were few and inadequate.

- No imminent user need. The expectation management process was partly ignored and not sufficiently supported by the involvement of users and others.
- Revolution not transformation. The change process was too extensive compared to the deployment process.

In the SPI test project the evaluation process found mutual factors that contributed to the successful diffusion and adoption of the new test process. A majority of these factors are equivalent to the recommended parameters found in the *ImprovAbility*<sup>TM</sup> assessment.

The contributing factors:

- “user need decisive for project initiation”
- “the improvement area was familiar to the users”
- “trend setting users were part of the development process”
- “involvement of the user community through meetings”
- “the crucial role of the user in the deployment activities”
- “educational and training activities close to practise”
- “tool support for process implementation and process ownership”

can all be interpreted as part of the *ImprovAbility*<sup>TM</sup> parameters: “Deployment means” and “Involvement of others”. Success parameters in the SPI test project show good relations to the findings in the *ImprovAbility*<sup>TM</sup> assessment.

### **9 Evaluation and conclusion**

The experiences from the two SPI projects have good correlation to the *ImprovAbility*<sup>TM</sup> findings and recommendations. The *ImprovAbility*<sup>TM</sup> model seems to make sense and actually gives valuable recommendations to a company like PBS.

### **10 Literature**

[www.delta.dk](http://www.delta.dk) (*ImprovAbility*<sup>TM</sup>)



### **11 Author CVs**

#### **Arne Staal Ibsen**

Has an academic degree in political science and economics. Has been involved in IT since 1980. Main focus includes IT and business development, user behavior, life cycle methodologies, development methods, IT strategies and Software Process Improvement. During the last couple of years IT security and architecture has been the main focus.

# Mining best practices of project management as patterns in distributed software development

*Antti Välimäki, Metso Automation Inc, Tampere, Finland  
Kai Koskimies, Institute of Software Systems, Tampere  
University of Technology*

## **Abstract**

Nowadays software products are even larger and more complicated than before. Sometimes it is not any longer possible to develop products in one site. In that case, there is a need to develop software in co-operation with other companies, which is one reason to change some projects from a centralized model to a distributed model. However, working with distributed development model is more difficult when compared to centralized development. Besides, distributed development methods are still quite unfamiliar. This paper describes the process of mining project management organizational patterns from distributed development projects in different cases in Metso Automation Inc, and presents the results of that work. The results were generalized in the form of a pattern, which makes it easier for other companies to reflect on and to apply the results to their own cases.

Organizational Pattern, Anti-pattern, Distributed development, Mining of patterns

### 1 Introduction

The need for distributed development is increased when software products need to be developed faster, cheaper and with better quality. One assumption might be that the development work can be made faster when tasks are divided in more than one site. Another assumption might be that it is cheaper to develop software in abroad because e.g. salary costs are lower. The third assumption might be that the quality is better because you can choose the best professionals to do certain work. Those assumptions can be true but there is one big risk that the distributed development work is more difficult than the centralized work. Other risks can be found in the distributed development researches. The distance, for example, will decrease communication, make it difficult to create a common vision with project members in different sites, decrease the speed of change and decrease the visibility of a project [2].

There exists a lot of research about distributed development in a VSC (Virtual Software Corporation) environment, e.g. [5]. A VSC consists of companies who are making co-operation in different sites [6]. Distributed development can naturally also be made between sites inside a company, as is the case with some evaluated projects in this paper.

Nevertheless, the distributed development is not a very common way of working and there is a need to find the best practices to avoid the pitfalls when choosing the methods to implement and organize a project. The organizational patterns seem to be one good method to describe the problem and the solution. There are different kinds of forms to describe patterns. For example, the Alexandrian form [1] is used when organizational patterns are described in the book [4]. In the Alexandrian form, the body starts with the statement of the context which includes the problem. The next part is the solution for the described problem. The more complicated organizational pattern form can be found in another piece of research [3] that presents a framework of agile patterns. There are three types of agile patterns which are practice patterns, concepts, and principles. Practice patterns describe actions, e.g. the creation of a functional specification, while concepts describe the attributes of an item, e.g. a functional specification. The third type, principles, consists of guidelines for development activities.

Which development process is a good target to start the mining of organizational patterns? At least a good project management is a key process to succeed in a distributed development project. There are two important phases in a project's life cycle in distributed development. The first phase is the start of the project and another important phase is the ongoing phase. A more specific description of project management phases can be found e.g. the Spice standard (ISO 15504). Some questions covered in the project management process section in the Spice standard are useful for the mining of project patterns.

The issues mentioned above were considered when the mining of best practices of project management as patterns in distributed software development was started. The goal of this software process improvement (later SPI) project was to find the best practices to improve distributed software process in Metso Automation Inc. The SPI project was implemented by the mining of pattern which was made with questionnaires and interviews with persons who were or had been involved in distributed projects for one year to several years. Thus, there were a lot of experiences from the good and not so good practices in distributed development projects.

### 2 Mining method and industrial context

The pattern mining was carried out by using a questionnaire and interviews. The interviewees were project managers or project members in distributed projects in Metso Automation. There were eleven interviewees in nine projects. Different distributed projects are or were implemented between Metso Automation sites in Finland, e.g. in Tampere, Helsinki, Kajaani, Jyväskylä etc., or in Canada, or between Metso Automation and other companies in other countries. The interviewed project managers and some other project members are especially familiar with the challenges in distributed development and they gave valuable information in questionnaires and through discussions in personal interviews.

The method in pattern mining was to send a questionnaire to each participant and then interview

him/her to clarify and research the answers about project management in distributed development more carefully.

The questionnaire was started with the motivation to answer to questions. It was followed by questions about the background, e.g. project name, how the work was divided between sites, what was the schedule of the project etc.

The other questions were gathered from the following issues: considering the distributed development risks, which problems, best practices, or ideas were involved with the project management methods both in the start-up and in the middle of the project. As stated earlier in the introduction chapter, the risk issues chosen for this research were the communication between project members (Communication), the understanding of the common goal and vision (Common Vision) in the whole project team, the speed of change between sites (Speed of Change), and the visibility of the project status (Project Status Visibility) to all project members.

When there was a best practice or an idea, it was also asked to describe it in more details if possible. When there was a problem, it was also asked to try to find a solution in the context of one's own project or in a project in which there were no constraints to implement them to free the imagination.

### 3 Suggested solution and results

The answers received from questionnaires and interviews were interesting. The most important comments were that the methods used in a distributed development must be re-evaluated from the best practices of centralized development. It is also important to understand and to find the changes which are needed because of the different requirements and constraints in a distributed development. For example, all data should be stored in common databases and everybody should be able to see at least the data which is needed in his/her work even though the sites are in different places. The visibility of the data is a security issue which should be considered carefully. Other comments were about communication and the visibility of the project and the related problems. The important methods in this area were suggested to be in all sites, e.g. a very good kick-off meeting, weekly meetings by the use of net meeting tools, the use of other communication tools, task-related communication mainly in written form, the use of a common Target & Task database, R&D documents database, source files database etc. Other important methods included task allocation according to the product architecture, the use of certain roles to improve communication etc.

The answers were analyzed after which practice patterns were created (later a pattern) based on gathered information, the experience of the author and related research materials. Some of the important patterns from the viewpoint of a distributed development challenges and project management are described below.

The patterns are grouped according to the form of the pattern below.

#### ***The form of a practice pattern***

- **Name and Id:** A short name of the problem (Number).
- **Problem:** Detailed description of the problem.
- **Problem area:** Is the pattern related to Communication or Common Vision or Speed of Change or Project Status Visibility?
- **Group category:** The group category to which the pattern belongs.
- **Solution:** Activities that will solve the problem. Activities can include other patterns. The underlined pattern means that there is another pattern which describes this more specifically.
- **Guidelines:** These are the guidelines on how to use this pattern.
- **Consequences:** The results and trade-offs when the pattern is applied.

There are different groups for different patterns and they are the same as in many process models. The group categories in this paper are:

## Session 6: SPI and Knowledge Management

- 1.x for project management
- 2.x for development
- 3.x for organization
- 4.x for support processes

Here are some patterns from the beginning of a project.

**Name and Id:** Plan the project – **(1.1)** (1.1 is the first pattern of a group 1.x which is for project management.)

**Problem:** The information about the goal of a project (including, for example, features, budget and time schedule) in a Business Plan or other document is too general. There is a need to make a more specific plan and to clarify who will participate in the project, what are the responsibilities of the involved personnel in different sites etc, so that the project group has all the required information to discuss the plan and to commit itself to the project.

**Problem area:** Communication and Common Vision

**Group category:** Project management

**Solution:**

- Make a project plan with the information from the business plan or other sources.
- Plan the product architecture and Allocate the development responsibilities between sites (3.2). (3.2 is the second pattern of group 3.x which is for organization. This pattern is not presented in this paper.)
- Choose your project members and make a decision about members' roles in development sites (3.3). (3.3 is the third pattern of a group 3.x which is for organization. This pattern is not presented in this paper.)

**Guidelines:** The business plan contains the main goals, features, and budget. The next step is to refine the project plan to implement the goal. If there are more than one site, there is a need to make a decision on how to allocate project targets and responsibilities between different sites. There are some alternatives to make this allocation depending on your members' knowledge and experience. The product architecture should be reflected in the allocation criteria because the architecture also has an effect on the communication relationships between members. More information in pattern **(3.2)**.

There are some important roles with distributed development which are needed in every site. The roles are a team leader, a change accelerator, a specifications interpreter, a domain expert, an architect, an IT administrator, and a developer. One person can naturally have many roles. More information in pattern **(3.3)**.

**Consequences:** If the plan is clear and project members know who is responsible for which part of the product, it is much easier to ask questions about tasks etc. If it is difficult to allocate the product architecture between sites, it will also be difficult to have good communication environment because the main flow of communication happens inside the modules of the product architecture.

Clear role responsibilities are important in distributed development because it makes it easier to solve who to ask a question. The roles are also needed to ensure that there are no breaks in information flow.

**Name and Id:** Start the project / Organize the first project meeting – **(1.2)** (1.2 is the second pattern of a group 1.x which is for project management etc.)

**Problem:** A lot of projects are started without a good kick-off meeting. If there is a kick-off meeting, all project members from different sites may not be able to participate in the same time.

**Problem area:** Communication and Common Vision

**Group category:** Project management

**Solution:**

- Invite all project members from all sites in one place.
- Announce the goal and the time schedule.
- Present project members and their roles and responsibilities.
- Organize some social group tasks for the whole group.

**Guidelines:** Have a good kick-off meeting and invite all project members to learn the presentations and maybe have a short group work to acquire the goal and the time schedule of the project. Assign a group work so that everybody can introduce oneself. Make a presentation about project members with their roles and responsibilities to clarify the work division in the project. This information is also needed later so take photographs, write down presented information and publish this information for the project group.

**Consequences:** It is important that all participants meet each other because it is much easier to talk with a person you have met than with a person you have communicated with only by mail or phone. It is obviously quite expensive to get together but the costs will be recouped by more efficient working.

Communication is more efficient if participants know whom they are working with. That is why it is very important to present persons, roles, and responsibilities.

This pattern is used to increase the visibility of a project.

**Name and Id:** Visibility of the project – (1.3)

**Problem:** In centralized development, it is easy to ask the status of tasks from project members or have a weekly meeting to improve the visibility of a project. A lot of information is shared by discussions in coffee tables or ad-hoc meetings. In distributed development it is not easy to share discussed information or arrange ad-hoc meetings very quickly. Also the distance makes it more difficult to start communication with project members from other sites.

**Problem area:** Communication and Project Status Visibility

**Group category:** Project management

**Solution:**

- Divide your high-level project plan to targets (1.4) and allocate them to participants.
- Manage targets and more detailed tasks in database (1.5) from which all data will be synchronized between sites in hours. Put a communication of a task in writing.
- Have a weekly meeting (1.6) by a phone or by a net meeting where the information is presented by the use of a computer and a video projector in every site.
- Use common document and source code tools (4.3) between sites and make the synchronizing of data happen in hours. *(4.3 is the third pattern of a group 4.x which is for support processes. This pattern is not presented in this paper.)*

**Guidelines:** Divide the high-level project plan to targets. Allocate targets to different project members who schedule their tasks, and add them in the common Target & Task database where data is visible to every project member in every site. It is possible to receive reports from the Target & Task database and to examine the status of the project and to make an analysis about the current progress. Put an important communication of a task in writing to assure that the risk of misunderstanding is smaller and the history of the tasks can be read later when needed. Add links from tasks to related tasks or documents or faults etc to make it easier to find out the related information later.

Also hold weekly meetings to assure that communication between project members is working. In the meeting, it is useful to shortly check the last week's results and the next week's task plan. It is natu-

rally important to check possible problems or difficulties which might distract the progress of the project.

The status of the project documents and source codes is also an important indicator of the current status of the project files. Prepare some reports from databases to make the status easily visible.

**Consequences:** It is important that the project manager concentrates mainly on targets and not on every task because the large amount of tasks makes it difficult to see the status of the whole project. It is also important that each project member will make his/her own schedule so that their commitment is much better than if the project manager had made all schedules. This is based on the concept that one's own idea is always the best idea.

If tasks are in the Target & Task database, it is possible to share the knowledge of tasks between project members and make a status report for the project manager and project members. Otherwise, it is very difficult to find out what the real status is - especially when working with many sites.

If there are no weekly meetings, there is no opportunity to point out the problems with development. As a result, the problems will emerge later when they will be more difficult to fix.

The patterns above are examples on how powerful method a pattern presentation is because, for example, the consequences show reasons why these solutions should be used thus motivating the reader to re-use these patterns in his/her own project.

The group categories in this paper are:

- 1.x for project management
- 2.x for development
- 3.x for organization
- 4.x for support processes

The list of patterns in this paper. The patterns which are underlined are not described in detail in this paper.

- Plan the project – **(1.1)**.
- Start the project / Organize the first project meeting – **(1.2)**
- Visibility of the project – **(1.3)**
- Divide your high-level project plan to targets **(1.4)**
- Manage targets and more detailed tasks in database **(1.5)**
- Have a weekly meeting **(1.6)**
- Plan the product architecture and Allocate the development responsibilities between sites **(3.2)**
- Choose your project members and make a decision about members' roles in development sites **(3.3)**.
- Use common document and source code tools **(4.3)**

### ***4 Exploiting the patterns in software development***

Similarly to other types of patterns, our distribution patterns are useful only as solutions to certain problems: an existing process is not necessarily improved by these patterns, unless the problems pointed out by the patterns are recognized. Thus, the deployment of the distribution patterns should be preceded by an analysis of the existing processes. To support such analysis, questionnaires and in-

Interviews of project managers and project members can be used to find out information about the current processes and tools.

An indication of the fact that the distribution patterns can improve the processes is the recognition of so-called anti-patterns [7]. Anti-patterns are typical bad solutions or practices that have well-known unfavourable consequences. The main benefit of the anti-patterns is that they document the situations where better solutions or practices would be needed, often given as actual patterns. Thus, anti-patterns can be used to recognize "bad smell" in existing distributed processes, and consequently the potential need to apply our patterns in those processes. Anti-patterns are typically created when a conventional centralized process is turned into a distributed one without carefully thinking about the real challenges.

In this research, we found some anti-patterns of distributed development processes. An example is an anti-pattern that could be called "God manager", analogously to the God class anti-pattern in [7]. A God manager tries to manage all sites alone without any site leaders in his/her project. One consequence was that the throughput of changes was long because the needed changes were not understood correctly and it took a long time to clarify those misunderstandings. Another anti-pattern concerned the use of different tools in different sites for the same work. This practice created a lot of extra work when information was manually transferred from one tool to another and vice versa. In this case, there was more than one way to solve this anti-pattern, for example, to automate the information transfer between different tools, or to take the same tools in use, or to divide the tasks in a new way. In this case, the same tools were chosen because it was considered to be the best solution in the long run.

When a decision to apply certain distribution patterns has been made, it is important to gather them in a place where they can be easily accessed, e.g. in a web-interfaced database. It is also important to train patterns and analyze their usage with project managers to improve the understanding of patterns, and try to find out which patterns are useful in which project. It is also possible that some patterns have to be customized according to the company context.

## **5 Conclusions and lessons learned**

Patterns seem to be an interesting way to describe solutions for specific problems in distributed or centralized development. It is possible to emphasize the important activities and consequences of a pattern which will improve the efficiency of a distributed project. For the pattern mining, the use of a questionnaire with personal interviews is also a good method to gather patterns and their contents. The patterns which have been found in this research are useful but obviously they need to be adapted according to the restrictions and possibilities of one's development project.

Distributed development is more difficult, and if a distributed project is run using the same methods and tools as a centralized project, the efficiency of the development project will decrease. This is the reason why it is important to find out a more efficient way of working. In any case, the distributed development model will be more common in the future than today because there is a need for sites providing special knowledge in Finland or another country, or there is a need for more flexible use of project members from different companies etc. The knowledge, methods, and tools to implement distributed development projects will be improved in the future and one way to improve methods is to find the best practices by the means of the pattern mining.

The future SPI and research directions will be the analysis of experiences with the current patterns in future development projects, the improvement of the patterns, and the creation of new patterns according to the feedback gained from projects.



### **Literature**

- [1] Alexander, C., Ishikawa, S., Silverstein, M., Jacobson, M., Fiksdahl-King, I., Angel, S., A Pattern Language: Towns, Buildings, Construction, Oxford University Press, New York, 1977.
- [2] Atkins, D., Boyer, D., Handel, M., Herbsleb, J., Mockus, A., Wills, G., Achieving speed in globally distributed project work, Human Computer Interaction Consortium, Frazer, Colorado, February 2001.
- [3] Bozheva, T., Elisa Gallo, M., Framework of agile patterns, European Software Process Improvement and Innovation Conference (EuroSPI), November, 2005.
- [4] Coplien, J. O., Harrison, N. B., Organizational Patterns of Agile Software Development, Pearson Prentice Hall, 2005.
- [5] Rahikkala, T., Towards virtual software configuration management, Doctoral thesis, VTT Publications 409, 2000.
- [6] Rahikkala, T., Taramaa, J., Välimäki, A., Industrial experiences from SCM current state analysis. In: Magnusson, B. (ed.), System Configuration Management, ECOOP'98 SCM-8 Symposium, Brussels, Belgium, July 1998, LNCS 1439, Springer, pp.13-25.
- [7] Brown, W., Malveau, R., McCormick, H., Mowbray, T. AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis, Wiley, 1998.

### ***Author CVs***

#### **Antti Välimäki**

Mr Antti Välimäki has worked in R&D in Metso Automation (ex-Valmet) since 1985 in many positions. He was graduated with a Master of Science degree from the department of Electrical Engineering (Software Systems and Automation) in Tampere University of Technology in 1986. He has also continued his postgraduate studies by studying software development at Tampere University of Technology. He has made research about software development in either centralized or distributed development. Since 1995 he has been working as a development manager, and his main responsibilities are software development methods, IT and Quality issues in R&D department and in PAS business area in Metso Automation in Finland.



# A Lightweight Model for the Assessment of Software Processes

*Francisco J. Pino<sup>1,2</sup>, Félix García<sup>2</sup>, Francisco Ruiz<sup>2</sup>, and Mario Piattini<sup>2</sup>*

*1 IDIS Research Group  
Electronic and Telecommunications Engineering Faculty  
University of the Cauca  
Street 5 # 4 – 70, Popayán, Colombia.  
fjpino@unicauca.edu.co*

*2 ALARCOS Research Group  
Information Systems and Technologies Department  
UCLM-Soluziona Research and Development Institute  
University of Castilla-La Mancha  
Paseo de la Universidad, 4 – 13071 Ciudad Real, Spain  
{Felix.Garcia, Francisco.RuizG, Mario.Piattini}@uclm.es*

## **Abstract**

Improvement in software development processes gives companies guaranteed high levels of maturity in their processes and increases their competitiveness in international terms. There are improvement, assessment and capability models which enjoy world-wide recognition but which must be adapted to the particular characteristics of the specific countries where those models are applied. These models can not easily be applied in the majority of organizations in many Latin American countries due to the large investment in money, time and resources that the models require. There is also the factor of the complexity of the recommendations they give, and the fact that the return on the investment is a long term prospect. This paper's main goal is to present MECPDS, a lightweight model for the assessment of the capability of software development processes and maturity of the organization. This model is based on the ISO/IEC 12207 e ISO/IEC 15504 standards and it is applicable to very small software enterprises. The model fulfils its function in a simple, economical way, using only a small amount of resources and in a short period of time.

## **Keywords**

Software process improvement, Software process assessment, Measurement Framework, Process capability, Process fulfillment, ISO/IEC15504:2004, Small and medium enterprises, SMEs.

### 1 Introduction

The software industry is a highly important economic activity in every country in the world. It provides a significant window of opportunity for developing countries, as is the case of the majority of Latin American nations. It must be said, however, that the software industry in the above mentioned countries is in its infancy and as such still immature [3]. This of course leads to an inability to compete, which in turn hinders growth.

So in Colombia, for example, the software development companies are not yet ready to compete in the international market. The computer sector faces a number of problems, such as the country's technological dependence. Alongside this, we can observe a lack of awareness of the importance that the development process has on the overall quality of the product. This is related to the fact that the software is manufactured like a craft and what is produced by the majority of companies is therefore of low quality. In addition, the time taken in development is not acceptable, costs are uncompetitive and activities in the operation and maintenance of the software are complicated. Final customers and users are therefore manifestly dissatisfied.

It is our conviction, therefore, that some strategies must be worked out to deal with all these problems. Such strategies will aim to set these countries on the same path as those nations which are already highly developed in terms of their IT industry. This will be done by setting up programmes for software process improvement. In every improvement process it is particularly important to be able to dispose of a suitable process assessment model. Such a model should identify the aspects which the organization really needs to improve. For that reason an improvement program depends in vast part on the acceptance of those aspects that the company must improve.

Increasing the quality of software products by improving processes is a measure which organizations should take when responding to two areas of special concern. The first of these has to do with their image, if they are going to be able to export software, and hence enter the global marketplace and maintain their position in it. These companies have could potentially be very competitive in trade of this kind, taking into account their low labour costs. The other area meriting special consideration is a patent need that these companies have, which is to be able to make their administrative project units efficient and affective.

One of the main characteristics of the Latin American software industry is that it is made up of very small software enterprises (VSEs). The term "VSEs" refers to small software enterprises having between 1 and 9 employees. As we all know, this type of company shows serious problems in the maturity of its development processes. In many cases there is no software development process known to the company. This leads to chaotic models of operation and these in turn affect the whole organization [5] and also, naturally, the software product. Although many of these organizations do set an organizational goal of ensuring the quality of products by taking on board the models of quality established in the SEI or ISO [3], it has to be said that these processes are really structured in such a way as to be applied in large companies, preferably. They cannot easily be applied in small organizations given the fact that an improvement project involves a large investment in terms of money, time and resources. There is also the great complexity of the recommendations, as well as the fact that the return on the investment undertaken has to be seen from a long-term perspective [7, 9, 13].

A project has been set up in Colombia with the aim to lessen the negative effects of the aspects we have just outlined. Its name is "Sistema Integral para el Mejoramiento de los Procesos de Desarrollo de Software SIMEP\_SW" (An Integrated System for the Improvement of Software Development Processes). This project seeks to provide organizations in the IT sector with the tools needed to help them to improve their software development processes. The goal is to increase the quality of the products developed, at the same time making it easier for the organizations to take up a competitive position in national and international markets.

As a result of this project an improvement system which integrates elements from improvement, capability, process and assessment models has been developed. These models are internationally recognized but they are tailored in such a way as to fit in with the specific characteristics of Colombia. Another feature is that this system can be copied by industries of a similar type at a national or interna-

tional level. What is sought is for improvement projects to follow a national model that is consistent with the country's particular idiosyncrasies and which is therefore adapted to the specific socio-economic context. [8].

The main result of the SIMEP\_SW Project is Agile SPI (Software Process Agile Improvement) [8], whose basic premise is that the models used should be lightweight and based on international standards. The architecture for Agile SPI is displayed in Figure 1

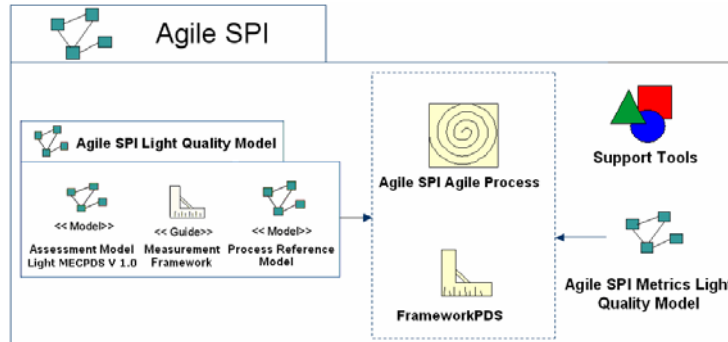


Figure 1: Architecture of Agile SPI.

The components of the system's architecture are as follows: (i) Agile SPI Agile Process: An agile process which guides the process improvement programme; (ii) Agile SPI Light Quality Model: A lightweight capability and assessment model of productive process; (iii) Framework PDS: conceptual and technical, to support processes and (iv) Agile SPI Metrics Light Quality Model: A lightweight model of measurements for the productive process.

In this article we introduce the definition of a lightweight model for the assessment of software development processes known as MECPDS, which is based on the ISO/IEC 15504:2004 [2] and ISO/IEC 12207:2004 [1] standards. This model provides a lightweight framework for the assessment of capability and the fulfilling of the process, along with a process reference model.

The paper proceeds as follows. In section 2 we give an overview of related work. Section 3 presents the model itself. In section 4 presents the utilization of MECPDS in an improvement programme. Finally, the conclusions and future work are outlined.

## 2 Related Work

Some Latin American countries have become concerned in recent years about quality of software development processes in their own industry, seeing it as a fundamental element in increasing product quality. Proof of this is seen in the "MoProSoft" model from Mexico and the "MR mps" from Brazil, amongst others that could be mentioned.

In the case of Mexico, the MoProSoft model has been developed - "Modelo de Procesos para la Industria de Software" [10] (Model of Processes for the Software Industry). This model is based on ISO 9001:2000, ISO/IEC 15504-2:1998 y CMM. MoProSoft aims to provide the software industry in Mexico with a model based on the best international practices. This model is at the same time easy to understand, simple to apply and economical to adopt. It seeks to assist organizations in standardizing their practices, in the assessment of their effectiveness and in the integration of ongoing improvement.

MoProSoft defines three process categories: High direction, Management and Operation. For each one of the processes it specifies three parts: a general definition of the system, practices and a guide for adjustments.

The basis tenet of its improvement strategy is that the organization should establish its own strategy for the setting up of the processes defined by the model. The processes should evolve in line with the suggestions for improvement. The objectives of the organization's strategic plan will be reached with increasingly ambitious goals being set all the time. In this way the company can reach maturity progressively, by this ongoing and continual improvement in its processes.

In Brazil, the mps Br project [14] has been developed. Its basis lie in ISO/IEC 12207:2002, CMMI e ISO/IEC 15504:2003. The mps Br project came up with two models: a Reference Model for the software improvement process– MR mps along with a Business Model for the software improvement process– MN mps. MN mps defines the elements and interactions involved in the certification of the organization by implementing MR mps in two ways: a personalized one for an organization or a group of organizations together (thus managing to make it more affordable for small and medium-sized enterprises). MR mps is made up of maturity levels, along with an assessment model. The maturity level is organized in two dimensions: capability and process. The process maturity is classified into seven levels: Optimized, Managed Quantitatively, Defined, Almost Completely Defined, Partially Defined, Managed and Partially Managed. Process areas are attributed to each maturity level based on the levels of CMMI. This is so as to ensure a gradual and fully appropriate implementation in Brazilian SMEs (Small and Medium Enterprises). The level of implementation of the practices associated with a process area is evaluated by means of indicators.

Other works related to the tailoring of the assessment process are RAPID (Rapid Assessment for Process Improvement for Software Developed) [12] based on ISO/IEC 15504:1998 and MARES (Método de Avaliação de Processo de Software) [4] based on ISO/IEC 15504:2003.

In previous models no improvement strategy guided by an improvement process has ever been set out explicitly. SIMEP\_SW bases its improvement strategy on providing the organization with an agile process which will set the basis for a programme addressed at the improvement of the processes. Thus it is absolutely vital to have a lightweight assessment available, since if we are to be able to promote improvement in software processes, there is something important for us to do beforehand. This is to establish an assessment framework which will let us know the strong and weak points of the assessed processes.

MECPDS is based on ISO/IEC 15504:2004. It defines the measurement framework for assessment in the capability dimension of the process as well as in the fulfilling dimension of the process. In the capability dimension, there are only three levels of maturity, making the model lighter, so that it can be applied to VSEs. With the fulfillment dimension the VSEs can have a better knowledge of the carrying out of its processes, which is the first step towards software process improvement.

### 3 The MECPDS Lightweight Assessment Model

MECPDS is made up of a measurement framework, together with a process reference model, both of which should be applied during the assessment of an organization's software processes (see figure 2).

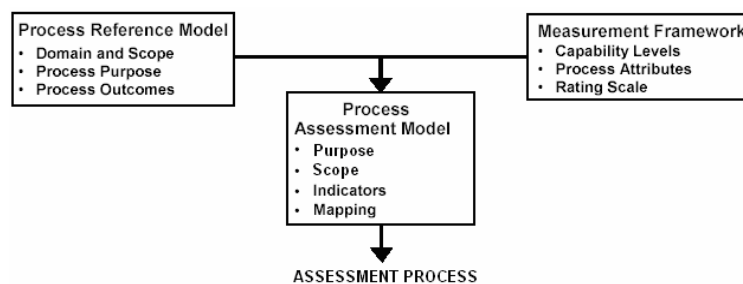


Figure 2: Structure of MECPDS

The purposes of MECPDS are:

- To set out the elements needed to evaluate the maturity and the fulfillment of an organization's processes in relation to a process reference model.
- To contribute a lightweight assessment model, this can be easily and economically applied to VSEs (with a small investment in money and time).
- To foster assessment on the part of the software VSEs in Colombia, so that they might find out their strong and weak points. That would be their basic guide when seeking to improve software process development in the organization.

- To form part of the Agile SPI Light Quality Model component in Agile SPI.

The scope MECPDS is made up of software life-cycle processes defined in the international standard ISO/IEC 12207:2004. However, MECPDS can use any reference process model, where each of its processes is described in terms of its purposes and its results. An element which also allows us to make the assessment more lightweight is the possibility of choosing the relevant applicable processes that are to be assessed in the organization.

To make the assessment model lighter, and bearing in mind the type of organizations in which it is to be applied, the assessment model is based on the international ISO/IEC 15504:2004 standard but only considering up to the level three of the capability model. Furthermore in section 3.2 a set of processes which are typically used by the Colombian VSEs is proposed.

The first step towards the improvement in VSEs is to define, manage and implement the processes, and these aspects are assessed by means of process attributes of the first and second capability levels (performed and managed) of the 15504 standard.

In addition, to make the assessment model lighter, three levels have been set out with three process attributes (from the nine attributes given by the standard). This means that a company which wants to assess the state of its processes is able to make the assessment significantly more lightweight in form.

MECPDS should form part of a software improvement programme set up by the organization, which takes their business and improvement goals as the starting point. From the set of processes described in the process reference model selected, the processes that are relevant and appropriate for assessment should be chosen. First of all, an assessment of the fulfillment process dimension has to be carried out in order to determine the degree of compliance with which a process is performed in relation to the selected process reference model. Then, once the process is explicitly defined to facilitate its carrying out, the capability process dimension is assessed.

MECPDS is based on a set of indicators which direct the purposes and results of all the processes within the process assessment model. These demonstrate what the attributes of the process have achieved in the realm of the capability level of the assessment model. These indicators are as follows:

- For the process capability dimension: the management practices that are associated with obtaining the results of the attributes of the process.
- For the process fulfillment dimension: the base practices associated with obtaining the results of the processes defined in the process reference model.

The degree of implementation of the practices is also evaluated by means of the indicators. These should be recognized by the organization for each practice involved, and may be of any of the following three sorts:

- Direct: these are the products which are the result of an activity.
- Indirect: in general, these are documents which indicate that an activity has been carried out.
- Comments: these are opinions given by those people who are involved in the process that is being evaluated.

### 3.1 Measurement Framework

The MECPDS measurement framework is based ISO/IEC 15504:2004 and embraces both the process capability dimension and the process fulfillment dimension.

**The process capability dimension** is defined by a hierarchical scale of three levels, which represent an increase in the capabilities of the software development processes: (i) Level 0: Incomplete Process, (ii) Level 1: Performed process, (iii) Level 2: Managed process. Reaching a level is shown on this dimension by the fulfilling of the process attributes. The process attributes are those elements which allow us to find out the capacities and abilities of a process. The process attributes are made up of



management practices.

A management practice is a process management activity which shows the capacity to carry out a process. A management practice supports the implementation or management of a process and can be applied to whatever process. Management practices allow individualized measurement. Thus we can determine the scope reached by the attribute to which it pertains, along with the specification of the particular level that the process under study is in. Each one of these attributes on its own allows us to measure a specific aspect of the capabilities and abilities in a process.

Both of the components described above (the management practices and process attributes), should have a specific scale of measurement. Thus for management practices and process attributes values are seen on a discrete scale made up of the following elements: (i) F: Fully achieved, (ii) L: Largely achieved, (iii) P: Partially achieved, (iv) N: Not achieved.

Each level demands a degree of fulfillment and/or a greater number of process attributes to reach it. In tables 1, 2 and 3 process attributes are specified, along with the management practices associated with it.

<b>Id. Attribute</b>	<b>Description attribute: Process performance</b>		<b>Scale</b>
PA 1.1	The process performance attribute is a measure of the extent to which the process purpose is achieved.		N, P, L, F
<b>Level</b>	<b>Id. Practice</b>	<b>Description of the management practice</b>	<b>Scale</b>
1. Performed	MP 1.1.1	The process achieves its defined outcomes	N, P, L, F

**Table 1: Process performance attribute**

<b>Id. Attribute</b>	<b>Description attribute: Performance management</b>		<b>Scale</b>
PA 2.1	The performance management attribute is a measure of the extent to which the performance of the process is managed.		N, P, L, F
<b>Level</b>	<b>Id. Practice</b>	<b>Description of the management practice</b>	<b>Scale</b>
2. Managed	MP 2.1.1	Objectives for the performance of the process are identified.	N, P, L, F
	MP 2.1.2	Performance of the process is planned and monitored.	
	MP 2.1.3	Performance of the process is adjusted to meet plans.	
	MP 2.1.4	Responsibilities and authorities for performing the process are defined, assigned and communicated.	
	MP 2.1.5	Resources and information necessary for performing the process are identified, made available, allocated and used;	
	MP 2.1.6	Interfaces between the involved parties are managed to ensure both effective communication and also clear assignment of responsibility.	

**Table 2: Performance management attribute**

<b>Id. Attribute</b>	<b>Description attribute: Work product management</b>		<b>Scale</b>
			N, P, L, F
<b>Level</b>	<b>Id. Practice</b>	<b>Description of the management practice</b>	<b>Scale</b>
2. Managed	MP 2.2.1	Requirements for the work products of the process are defined.	N, P, L, F
	MP 2.2.2	Requirements for documentation and control of the work products are defined.	
	MP 2.2.3	Work products are appropriately identified, documented, and controlled.	
	MP 2.2.4	Work products are reviewed in accordance with planned arrangements and adjusted as necessary to meet requirements.	

**Table 3: Work product management attribute**

The value of a process attribute is obtained by finding the average of the percentage values of the management practices. It should be noted that each management practice has the same weight in a process attribute.

Table 4 defines the level of capability associated with a process, which allows the measuring of the degree of quality in a software product it has created. There is a relationship between the levels of capability and the degree to which the attributes of the process being evaluated have been fulfilled.

<b>Capability Level</b>	<b>Attributes of the process</b>	<b>Rating</b>
Level 1. Performed	Process performance	L or F
Level 2. Managed	Process performance	F
	Performance management	L or F
	Work product management	L or F

**Table 4: Fulfillment of Capability Levels**

A capability level is defined for each one of the processes evaluated and defined by the process reference model. It is important, however, to give an overview of the state of the maturity of the organization. To evaluate the “overall level of maturity” of the organization, the assessment results of the processes associated and defined by the process reference model of MECPDS (see Table 5) are taken into account.

Level of General Maturity of the Organization	Criterion to reach the level
Level 1. Performed	If all the processes applicable to the organization in the attribute of process PA 1.1, have a degree fulfillment L or F <b>then</b> the level is reached about the organization, <b>else</b> the level is not reached about the organization.
Level 2. Managed	If all the processes applicable to the organization in the attribute of process PA 1.1, PA 2.1 and PA 2.2, have a degree fulfillment L or F <b>then</b> the level is reached about the organization, <b>else</b> the level is not reached about the organization.

**Table 5: Determination of the maturity level of the organization**

**The process fulfillment dimension** is characterized by its focus on the characteristics and purposes of a specific process that has been established and defined by a process reference model. The processes are made up of base practices, which are software engineering activities which directly guide the purpose of a particular, contributing to the generation of its outputs.

The goal of this dimension is that VSEs can assess their processes in order to identify their strong and weak points. This dimension, therefore, provides the basis for the improvement of the VSEs processes, since it requires the definition and determination about how their processes are carried out. At least, these small organizations must achieve the fulfillment level L in their processes in order to guarantee a minimum knowledge of the processes to assess.

On this dimension, reaching a process is demonstrated by the fulfilling of the base practices associated with the process which is being assessed. The base practices can be individually measured and it allows VSEs to find out the value of fulfillment to which the process under study has been achieved.

In order to assign an implementation value to the base practices and to the processes, a specific scale is needed for that measurement. These values are on a discrete scale made up of the following elements F, L, P or N. The process fulfillment value is obtained by finding the average of the percentage values of the base practices, expressed in the values defined beforehand. It should be noted that each base practice has the same weight in a given process.

A fulfillment value is defined for each one of the processes which are evaluated and defined by the reference process model. It is important, however, to give an overview of the state of fulfillment of the organization’s processes. First of all, the value of the fulfillment of each one of the process categories should be obtained (primary, supporting and organizational) defined in the process reference model. This value is obtained by finding the average of the percentage values of the corresponding processes, with this average expressed in terms of F, L, P or N. Once more it should be observed that each process is of equal weight.

To work out the “overall state of process fulfillment” in the organization, the fulfillment value of each one of the process categories should be taken into account. The value of the overall state of the process fulfillment in the organization is obtained by finding the average of the percentage values of its process categories, expressed in terms of F, L, P or N. Each process category should have the same weight.

### 3.2 The MECPDS Process Reference Model

MECPDS can use any process reference model, where each of its processes is described in terms of its purposes and its results (or a set of goals), for instance ISO/IEC 12207 or CMMI. Nevertheless, in order to give directions to the VSEs on which processes to establish when they initiate an improvement programme, a set of processes which are typically used by the Colombian VSEs is proposed. These processes are based on the ISO/IEC 12207 standard. The processes proposed in the table 6 have been obtained from different research works, as it is described in the following lines.

<b>PRIMARY</b> Software Life Cycle Processes	<b>PRI 3</b>	Development	<b>PRI 3.1</b>	Requirements elicitation
			<b>PRI 3.2</b>	System Requirements Analysis
			<b>PRI 3.3</b>	System Architectural Design
			<b>PRI 3.4</b>	Software Requirements Analysis
			<b>PRI 3.5</b>	Software Design
			<b>PRI 3.6</b>	Software Construction
<b>SUPPORTING</b> Software Life Cycle Processes	<b>SUP 1</b>	Documentation		
	<b>SUP 2</b>	Configuration Management		
	<b>SUP 3</b>	Quality Assurance		
	<b>SUP 11</b>	Change request management		
<b>ORGANIZATIONAL</b> Software Life Cycle Processes	<b>ORG 1</b>	Management	<b>ORG 1.3</b>	Projects Management
			<b>ORG 1.6</b>	Measurement
	<b>ORG 3</b>	Improvement	<b>ORG 3.1</b>	Process Establishment

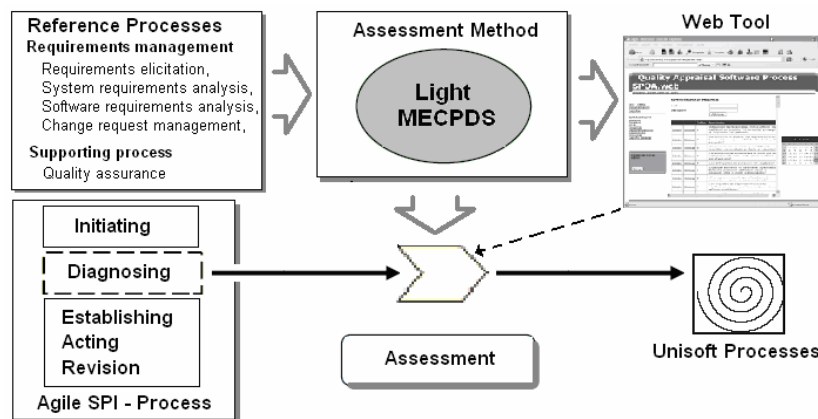
**Table 6: Reference Processes for MECPDS**

- A systematic review of the literature about the SPI efforts carried out in small and medium software enterprises –SMSEs- presented in [11]. Its research question is ¿What approaches concerning SPI have been focused on SMSEs and also present a real case study? The objective is to know what has been carried out and achieved about software process improvement in this type of companies. According to this systematic review the frequency of the improvement efforts aim to improve processes like project management, documentation, change requirement management, processes establishment, configuration management and requirements elicitation.
- The technical report about the state of the practice of software development processes in the Colombian southwestern region, presented in [8]. In the report an overview of software development practices, typically used to construct software products in several representative companies of the region, is shown. This study involved twenty software companies of three cities different from the region. These companies were visited by two people of SIMEP\_SW project to carry out an interview and a survey. The 70% of the visited companies have less than 10 employees, 20% have between 11 and 20 employees and 10% have between 21 and 50 employees. In accordance with this technical report the practices focus mainly on the engineering processes (requirements, analysis and design, construction and testing), project management and quality assurance.
- The measurement process has been proposed because is a key responsibility in the software process management and improvement. Measurements are the basis for detecting deviations from acceptable performance of process, furthermore they are also the basis for identifying opportunities for process improvement [6].

### 4 Utilization of MECPDS in an improvement programme

Currently MECPDS is being applied in an improvement programme carried out by Unisoft Colombia Ltda., a small software organization from Cauca (Colombia) with 5 years experience in the regional market. The company has four employees, two people are dedicated to the administrative area of the organization and other two are dedicated to the software product development, operation and maintenance. Unisoft has two main products: “Academic”, which is a software for academic management and a software product for payroll management.

The SPI programme began in December of last year with the support of the SPI Group part of the IDIS Research Group. The improvement programme is guided by Agile SPI Process, a tailoring for small software organizations of the IDEAL model. The installation phase has been already carried out. Currently, the improvement programme is being applied on the diagnosing phase in which assessment activities to estimate the general state of processes in the company are being performed. The process assessment method used is Light SPI Quality Model which uses MECPDS. For the first iteration, the processes related to requirements management such as requirements elicitation, system and software requirements analysis, change request management, have been chosen. Also the quality assurance which is related to supporting process has been chosen. A web tool [8] developed by the SPI Group is being used to support the assessment activity. (See figure 3)



**Figure 3: Unisoft SPI programme and Light MECPDS**

Although at this moment the assessment process is running, from the first application of Light MECPDS in the enterprises all participating considered that the assessment experience was satisfactory. Furthermore, people of the company who are participating in the assessment agreed that:

- The questions of the interview and survey (from assessment instrument) allow them to visualize easier the changes they are adopting, to keep in mind the activities they should do but they do not actually do, and identify their problems and weaknesses.
- The assessment contributed to a better understanding of their software processes. The person in charge of the improvement in the company can have a general vision of the states of the processes and this information is useful to manage the software process improvement programme.
- The assessment process consumed little time and few resources because the assessment instruments have few questions.

Our experiences acquired in this first study case, indicate the suitable applicability Light MECPDS in VSEs. The model's principal strength is its simplicity, especially because use only three levels of standard ISO/IEC 15504 for assessment of the capability dimension. This allows to focus the improvement effort of very small software companies in establishing and managing its processes. Furthermore is easy to use and useful for rapid assessment, allows a fast feedback to improvement programme.

## 5 Conclusions

In this paper a lightweight model for the assessment of software process quality has been presented. Its basic components are: the measurement framework and the process reference model. The measurement framework gives a guide for working out the capability and level of fulfillment of a process. In the process capability dimension it defines how to assess the general capability level, the specific levels, process attributes and management practices. In the process fulfillment dimension it establishes how to assess the overall state of fulfillment, categories, processes and base practices.

One important contribution of this work is the definition of the process fulfillment dimension which has to be assessed before the capability process dimension. It provides VSEs with the necessary reference to identify and define their processes, which is an essential requirement for software process improvement programmes to be successful.

Furthermore, in this paper the set of processes which are typically used by the Colombian VSEs have been presented. This set of processes has been obtained from a systematic review of the literature on the SPI efforts carried out in small and medium software enterprises and a study of the state of the practice of software development processes in the Colombian southwestern region. The proposal considers also the measurement process as a key activity in the software process management and improvement, in order to measure process capability and to lead the improvement of the organiza-

tional processes maturity.

A tool prototype called SPQA\_web has been produced for the MECPDS lightweight assessment model. It supports the MECPDS information gathering instrument and, since it is a Web tool, it contributes characteristics which make its application easier and help to fulfill certain objectives which open the way towards creating a culture of quality in small software organizations.

Currently, the proposed model is being applied in the Unisoft Company and we intend to apply it to other organizations in order to evaluate, fine-tune and validate the MECPDS model and support tool.

### **Acknowledgments**

This work has been possible thanks to Project COMPETISOFT financed by CYTED, to the project SIMEP\_SW financed by Colciencias and University of Cauca, to the company Unisoft Colombia Ltda. and to the research group in software processes improvement - SPI Group.

### **Literature**

1. *ISO/IEC 12207:2002/FDAM 2. Information technology - Software life cycle processes*. 2004, International Organization for Standardization: Geneva.
2. *ISO/IEC 15504-2:2003/Cor.1:2004(E). Information technology - Process assessment - Part 2: Performing an assessment*. 2004, International Organization for Standardization: Geneva.
3. *Panorama de la Industria del Software en Latinoamérica*. 2004, Mayer & Bunge Informática LTDA: Brasil. p. 97.
4. Anacleto, A., C.G.v. Wangenheim, C.F. Salviano, and R. Savi. *A Method for Process Assessment in Small Software Companies*. in *4th International SPICE Conference on Process Assessment and Improvement (SPICE 04)*. 2004. Portugal. p. 69-76
5. Batista, J. and A. Figueiredo, *SPI in a very small team: a case with CMM*. *Software Process: Improvement and Practice*, 2000. **5**(4): p. 243-250.
6. Florac, W.A., R.E. Park, and A.D. Carleton, *Practical Software Measurement: Measuring for Process Management and Improvement*. 1997, Pittsburgh: Software Engineering Institute, Carnegie Mellon University. 1-12.
7. Hareton, L. and Y. Terence, *A process framework for small projects*. *Software Process: Improvement and Practice*, 2001. **6**(2): p. 67-83.
8. Hurtado, J. and otros, *Agile SPI : Sistema integral para el mejoramiento de procesos de desarrollo de software en Colombia. Proyecto SIMEP-SW*. 2006, Universidad del Cauca y Colciencias.: Popayán, Colombia.
9. Maller, P., C. Ochoa, and J. Silva, *Lightening the Software Production Process in a CMM Level 5 Framework*. *IEEE Latin America Transactions*, 2005. **3**(1): p. 14-21.
10. Oktaba, H. *MoProSoft®: A Software Process Model for Small Enterprises*. in *Proceedings of the First International Research Workshop for Process Improvement in Small Settings*. 2006. Pittsburgh: Carnegie Mellon University. p. 93-101
11. Pino, F., *Revisión sistemática de mejora de procesos software en micro, pequeñas y medianas empresas. RT Competisoft-001/UCLM*. 2006, Grupo IDIS, Universidad del Cauca y Grupo Alarcos, Universidad Castilla - La Mancha: Ciudad Real, España.
12. Rout, T., A. Tuffley, B. Cahill, and B. Hodgen. *The Rapid Assessment of Software Process Capability*. in *SPICE 2000*. 2000. Limerick. p. 47-55
13. Saiedian, H. and N. Carr *Characterizing a software process maturity model for small organizations*. *ACM SIGICE Bulletin*, 1997. **23**(1): p. 2-11.

14. Weber, K., E. Araújo, A. Rocha, Machado, D. Scalet, and C. Salviano, *Brazilian Software Process Reference Model and Assessment Method*, in *Computer and Information Sciences*. 2005, Springer Berlin / Heidelberg. p. 402-411.

## Author CVs

### Francisco J. Pino

Is student of PhD in Computer Science in the University of Castilla-La Mancha (UCLM). Specialist in Telematics Networks and Services from University of Cauca. Assistant Professor at the Electronic and Telecommunications Engineering Faculty at the University of Cauca in Popayán (Colombia) and member of IDIS Research Group. His research interest is software processes improvement in small companies.

### Félix García

Félix García is MsC and PhD in Computer Science by the University of Castilla-La Mancha (UCLM). He is assistant Professor at the Department of Computer Science at UCLM in Ciudad Real (Spain) and member of Alarcos Research Group. Author of several papers and book chapters on software processes management, from the point of view of their modeling, measurement and technology. His research interests include business process management, software processes, software measurement and agile methods.

### Francisco Ruiz

Is a PhD in Computer Science from the University of Castilla-La Mancha (UCLM) in 2003, and MSc in Chemistry-Physics from the University Complutense of Madrid in 1982. He is a full time associate professor of the Department of Computer Science at UCLM in Ciudad Real (Spain) and member of Alarcos Research Group. He was the Dean of the Faculty of Computer Science between 1993 and 2000. Previously, he was Computer Services Director in the above-mentioned university (1985–1989) and he has also worked in private companies as an analystprogrammer and project manager. Author of several books, book chapters and papers in national and international journals, congresses and conferences. His current research interests include: software process technology and modelling, software maintenance, and methodologies for software projects planning and managing.

### Mario Piattini

Is a MSc and PhD in Computer Science from the Politechnical University of Madrid. MSc in Psychology from the UNED. Certified Information System Auditor and Certified Information Security Manager from ISACA (Information System Audit and Control Association). Full Professor at the Department of Computer Science at the University of Castilla-La Mancha, in Ciudad Real, Spain. Author of several books and papers on databases, software engineering and information systems. He leads the ALARCOS research group specialized in information system quality. His research interests are: software quality, advanced database design, metrics, software maintenance, information system audit and security.



# Adept – A Software Process Appraisal Method for Small to Medium-sized Software Development Organisations

Fergal Mc Caffery<sup>1</sup>, Ita Richardson<sup>1</sup>, Gerry Coleman<sup>2</sup>

<sup>1</sup>Lero- the Irish Software Engineering Research Centre,  
University of Limerick,  
Ireland.

[Fergal.mcCaffery@dkit.ie](mailto:Fergal.mcCaffery@dkit.ie), [ita.richardson@ul.ie](mailto:ita.richardson@ul.ie)

<sup>2</sup>Dundalk Institute of Technology,  
Dundalk,  
Ireland.

[gerry.coleman@dkit.ie](mailto:gerry.coleman@dkit.ie)

## Abstract

In this paper we describe the appraisal method that was developed by Lero (the Irish Software Engineering Research Centre) and Dundalk Institute of Technology (DkIT) to assess software processes within small to medium-sized Irish software organisations that have little or no experience of software process improvement (SPI) programmes. We developed a method, called Adept, through investigating the key factors that contribute to an effective software assessment method, reflecting the desires of Irish SMEs in terms of providing a “manageable” approach to SPI and through expanding upon the Express Process Appraisal (EPA) method [1] to fulfil these objectives.

## Keywords

Assessment method, Small to Medium-sized Enterprises (SMEs), Software Process Improvement (SPI)

## 1 Introduction

This Adept assessment method was developed as a result of a meeting between researchers from Dundalk Institute of Technology (DkIT), Lero – the Irish Software Engineering Research Centre, and Enterprise Ireland (the economic development agency for indigenous Irish companies) in relation to how a culture of SPI could be instilled into small to medium sized Irish software development organisations. Among other initiatives, it was decided that for individual companies who were interested in SPI, it would be important to initially understand the current state of software practice within these companies. Therefore, an assessment method had to be developed that could diagnose any weaknesses in a company’s software process. This diagnosis would then form the starting point from which to base a path for continuous SPI that will make the company more effective in fulfilling their business goals.

The remainder of the paper is structured as follows. Section 2 investigates the important characteris-



tics of effective lightweight assessment methods, and section 3 lists the requirements for the Adept method. Section 4 describes the Adept method in detail, while section 5 discusses the benefits of the Adept method. Finally, section 6 contains our conclusion.

## 2 Investigation into existing lightweight software process assessment methods

In order to provide guidance to our development of an assessment method that would be suitable for the Irish software industry we looked at some preliminary results from the Irish market [2] and performed a literature review of software process assessment methods used within other regions. We discovered that such assessment methods are generally based upon one of two process models: (i) ISO/IEC 15504 [3] and (ii) CMMI<sup>®</sup> [4]. There have been many attempts to develop light-weight process assessment methods for each of these models. For example, RAPID [5], SPINI [6], FAME [7], TOPS [8], and MARES [9] are all based on the ISO/15504/5 reference model. With regard to the CMMI<sup>®</sup>, EPA [1] is an example of an ARC class-C compliant method.

Anacleto et al. [10] have considered process assessment methods for small software companies. They provide criteria for comparing assessment methods and have compared the ISO/IEC 15504 based methods mentioned above. In this paper, we compare the CMMI<sup>®</sup> class-C compliant EPA method against the other assessment methods using criteria from the Anacleto et al. [10] comparison framework, discussing particular features which are applicable for Adept.

The criteria Anacleto et al. [10] uses to compare assessment methods is as follows:

1. *Low cost* - The EPA, RAPID, SPINI, TOPS and MARES assessment methods all try to minimise the cost of performing an assessment and these are consistent with the preliminary findings in [2]. In order to encourage adoption within the Irish software SMEs it will be important to minimise the cost of performing the Adept method.
2. *Detailed description of the assessment process* - The EPA, SPINI, TOPS and MARES assessment methods all provide a detailed description of the assessment process. Similarly, the Adept method will provide a description of the assessment process.
3. *Guidance for process selection* - The RAPID assessment method does not provide any information in relation to process selection as it pre-defines eight mandatory process areas. The MARES assessment method plans to provide guidance for process selection in the future. However, both the SPINI and EPA assessment methods currently provide guidance for process assessments. The Adept method will enable companies to select assessment within software process areas deemed to provide the most benefit to their business goals. Therefore the Adept method will contain optional process areas so guidance will be provided in relation to process selection.
4. *Detailed definition of the assessment model* - The EPA, RAPID, SPINI, TOPS and MARES methods each provide a detailed definition of the assessment model. The Adept method will provide definitions of both the CMMI<sup>®</sup> and ISO/IEC: 15504 models.
5. *Support for identification of risks and improvement suggestions* - Most of the assessment methods (EPA, SPINI, FAME and TOPS) provide support for the identification of risks and improvement suggestions, with this feature currently being developed in the MARES method. This will be provided within the Adept method.
6. *Support for high-level process modelling* - Only one of the assessment methods (FAME) provides this feature with another (MARES) currently developing this feature. This will not be provided within the Adept method.
7. *Conformity with ISO/IEC 15504* - The assessment methods (RAPID, SPINI, FAME, TOPS and MARES) are conformant with ISO/IEC 15504 whereas the EPA method is compliant with the ARC 1.1 for a CMMI<sup>®</sup> class-C method. The Adept method will conform with both the ISO/IEC 15504 and CMMI<sup>®</sup> models.
8. *No specific software engineering knowledge required from companies' representatives* - Unlike FAME and TOPS, four of the assessment methods we investigated (EPA, RAPID, SPINI and

MARES) do not require assessment participants to have specific knowledge of the assessment model. In order to encourage participation (particularly in SMEs) within the Irish software industry it will be vital to minimise the overhead on the company engaging in the appraisal. Therefore, the Adept method may be performed within an organisation without the assessment participants from the company having to undergo training on the assessment model.

9. *Tool support* - Tool support for the assessment methods we investigated appears to vary with the SPINI and FAME having tools to assist with data collection, analysis and rating. Tool support is currently being developed for the MARES method. However, the RAPID and TOPS assessment methods do not include tool support and therefore depend upon paper forms. The EPA method provides tool support but also relies upon paper forms so that both assessors may compare data when preparing the findings report. The Adept method will provide tool support similar to that adopted within the EPA method.
10. *Public availability* - Only two of the assessment methods (TOPS and MARES) are publicly available. There is no current requirement for the Adept method to be made public.

### **3 Requirements for the Adept assessment method**

From investigating other assessment methods and based upon the experiences gained by one of the researchers when involved both in the development and usage of the EPA method within SMEs in Northern Ireland, the Adept method was designed to adhere to 8 of the 10 criteria outlined by Anacleto et al. [10], the exceptions being, 6. *Support for high-level process modelling* and 10. *Public availability*.

However, Enterprise Ireland (representing the Irish SMEs), requested that Adept take the following factors into account:

- Improvement is more important than certification and a rating is not required;
- The amount of preparation time required by the company for the assessment should be minimal;
- The assessment should be performed over a short period of time;
- The assessment method should enable companies to select assessment in process areas that are most relevant to their business goals;
- Whilst the assessment will be based upon both the CMMI<sup>®</sup> and ISO/IEC 15504 models the SPI models should be invisible to the SMEs that are being assessed.

### **4 The Adept method**

The Adept method development was based largely on the structure of the EPA method that was adopted in Northern Ireland [1], but with the following differences:

- Adept does not highlight either CMMI<sup>®</sup> or the ISO/IEC:15504 SPI models. It refers to general SPI, thus focusing on improvement rather than certification.
- Adept enables the development of a SPI path, based upon a company's business goals, from the findings report that is produced as a result of the assessment.
- Adept includes a stage that involves revisiting the company after a period of 3 months and re-assesses the company's SPI path.
- Stage 1 (Develop Appraisal Schedule) and stage 3 (Conduct site Briefing) of the EPA method are both covered in stage 1 of the Adept method but require only half the time to perform.

### 4.1 Selecting an SPI model

The main aim of the Adept method is to encourage SPI improvement based upon the generic SPI principles that are shared by both CMMI<sup>®</sup> and ISO/IEC:15504. To progress the Adept method we had to decide whether to:

- Develop a completely new model that would contain new process areas based upon input from both the CMMI<sup>®</sup> and the ISO/IEC:15504 models;
- Base the Adept method upon relevant process areas from the CMMI<sup>®</sup> model and include input from the ISO/IEC:15504 model
- Base the Adept method upon relevant process areas from the ISO/IEC:15504 model and include input from the CMMI<sup>®</sup> model

As the development team had more experience of the CMMI<sup>®</sup> model, option (b) was preferred to option (c). Additionally, option (a) was ruled out due to the effort that would be involved in developing both a new assessment model and an assessment method. Therefore the Adept method consists of an assessment component for each CMMI<sup>®</sup> process area that is deemed applicable for Irish SMEs. However, even though each assessment component adopts a CMMI<sup>®</sup> process area name, it will provide equal coverage of both the CMMI<sup>®</sup> and ISO/IEC 15504 models by containing questions that relate to ISO/IEC 15504 in addition to questions based upon the CMMI<sup>®</sup> model.

### 4.2 What Process Areas should the Adept Method Assess?

The next important key decision in the development of the Adept method was to decide what process areas are most applicable to the Irish software SMEs. The process areas included in Adept are based on the previous involvement in the EPA method development and usage, and upon the results of previous research performed by DkIT and Lero in relation to software processes with Irish SMEs [2,12,13]. Based upon this information we then sequentially (by taking a CMMI<sup>®</sup> staged model maturity level grouping of process areas at a time) investigated the potential benefits to software SMEs of each of process areas within the CMMI<sup>®</sup> model.

#### 4.2.1 *What CMMI maturity level 2 processes should be included?*

Upon investigation, six of the seven process areas associated with maturity level 2 of the CMMI<sup>®</sup> model were selected as they constitute the engineering management basis of an organisation and the foundation upon which an efficient software company is based. We omitted the seventh CMMI<sup>®</sup> process area at maturity level 2 (Supplier Agreement Management) as previous research indicated that it would not be as beneficial to SMEs in Ireland as other process areas [11]. Furthermore, omitting one process area ensures that we avoid situations where a company might attempt to claim some form of maturity level 2 compliance in the event that no issues or weaknesses were discovered in the process areas appraised. As stated previously, the Adept method will not attempt to provide any form of rating. The Adept method includes assessment components for each of the following process areas at CMMI<sup>®</sup> maturity level 2: Requirements Management, Configuration Management, Project Planning, Project Monitoring & Control, Measurement & Analysis and Process & Product Quality Assurance.

#### 4.2.2 *What CMMI maturity level 3 processes should be included?*

Upon investigation of the CMMI<sup>®</sup> maturity level 3 process areas, six of the fourteen process areas were deemed applicable for Irish software SMEs. Therefore, an assessment component was included in the Adept method for each of these process areas. These CMMI<sup>®</sup> maturity level 3 process areas are: Risk Management, Technical Solution, Verification, Validation, Requirements Development and Product Integration.

#### 4.2.3 Should any CMMI maturity level 4 & 5 processes be included?

The process areas listed at CMMI<sup>®</sup> maturity levels 4 and 5 would be of less benefit to companies that have little or no experience in SPI and therefore an assessment component was not required within the Adept method for any of the process areas at these levels.

In summary, the Adept method will enable an assessment to be performed in 12 process areas.

### 4.3 Should assessment in certain process areas be given priority?

While all 12 process areas may be assessed using the Adept method, four will be mandatory - Requirements Management, Configuration Management, Project Planning, Project Monitoring & Control. These process areas are critical to the success of any software development company. The choice of mandatory process areas was based upon the overlap of three factors. Firstly, priority was given to the process areas that are deemed to be the foundation of the CMMI<sup>®</sup> model. Secondly, priority was assigned to the process areas in which SMEs would gain most benefit [11]. Thirdly, research in Ireland has shown that specific processes are seen as important by software SME managers [2, 12, 13].

### 4.4 How many process areas should be assessed within a single Adept appraisal?

In an attempt to reduce the cost and time associated with the assessment, on-site interviewing should be restricted to one day. It was therefore decided that an Adept assessment would be limited to six process areas as this is as many as can reasonably be covered within one day [11]. Therefore, in addition to being assessed in the four mandatory process areas, companies will also be able to choose two of the other process areas discussed in Section 4.2. Based upon previous research into the applicability of process areas to software SMEs [11], companies will be advised against selecting either the Measurement & Analysis or Process & Product Quality Assurance process areas unless these process areas are directly linked to their business goals.

### 4.5 The Stages of the Adept Method

The Adept method is divided into eight stages. The appraisal team consists of two assessors who conduct the appraisal between them.

Stage 1 (*Develop Appraisal Schedule and Receive Site Briefing*) is a preliminary meeting between the appraisal team and the software company wishing to undergo an SPI assessment. This stage consists of two parts. The first part involves establishing the logistics, selecting the most applicable process areas and determining the schedule of the appraisal. The second part is used by the appraised organisation to explain elements of the company structures to the appraisal team, who learn a little about the company's history, the company's business objectives and about the types of ongoing projects, along with the lifecycle stage that each project has reached. This meeting involves 2 assessors and at least one representative from the company. This meeting lasts approximately two hours. Therefore 4 person-hours of assessor time and at least 2 person-hours of company time are normally required for this stage.

During stage 2 (*Conduct Overview Briefing*) the lead assessor provides an overview of the method for members of the appraised organisation who will be involved in subsequent stages. This session is used to remove any concerns that individuals may have and to establish codes of conduct and confidentiality. This overview session involves 2 assessors and on average 7 company staff (the number of company staff involved depends upon the size of the company). The overview normally lasts 1 hour. Therefore 2 person-hours of assessor time and 7 person-hours of company time are normally required for this stage.

Stage 3 (*Analyse Software Documentation*) provides a brief insight into project documentation. Normally the following documents will be requested: a typical project plan, a typical project progress report, a typical approved requirements statement and any documentation relating to the company policy on configuration management. The primary source of data for the Adept method is through a series of process area interviews conducted during stage 4. The brief consideration of some sample documents during stage 3 is used mainly to craft further questions for stage 4. This stage will involve 2 assessors and usually 1 member of personnel from the appraised organisation. Typically, this stage will involve the company member dedicating 1 hour to retrieving the requested documents. The 2 assessors performing the appraisal will then each analyse this data for approximately 3 hours. Therefore 6 person-hours of assessor time and 1 person-hour of company time are normally required for this stage.

The main part of the Adept method is stage 4 (*Conduct Process Area Interviews*). In this stage key staff members from the appraised organisation are interviewed. There are 6 interviews. Each interview is scheduled to last approximately 1 hour. However based upon the experiences with the EPA in Northern Ireland, interviews for the process areas of project planning and project monitoring and control typically require 1.5 hours, therefore 7 hours is required to complete the 6 process area interviews. Each interview involves two assessors, and at least one representative from the company (on average 3 staff are involved) is present for each process area interview. Therefore 14 person-hours of assessor time and on average 21 person-hours of company time are normally required for this stage. The schedule of the process area interviews should be carefully designed to follow the natural sequence of the software development lifecycle. This will assist the assessors in painting an accurate profile of the software development practices adopted by a company and responses from process areas' interviewees earlier in the day may be cross-referenced in later interviews (without specifying names or project details as all interview data is treated with the strictest confidence). During each of the process area interviews, one of the assessors invokes responses from the interviewees using a combination of pre-defined and follow-up questions while the other assessor makes notes. The lead assessor will use an Excel based tool which enables them to make an initial judgement about the responses by judging them against a discrete set of values – Red (not practiced), Amber (partially practiced), Yellow (largely practiced) and Green (fully practiced). In this way, the opinions of the questioner and not just the note-taker will also be recorded for subsequent review.

Stage 5 (*Generate Appraisal Results and Create the Findings Report*) is very much a collaborative exercise between the two assessors. The findings report will consist of a list of strengths, issues and suggested actions for each of the process areas evaluated. Global observations covering all process areas are also covered and the initial judgements recorded in the Excel tool are revised. The findings report is then developed through a review of the interview notes and the scores produced by the Adept tool for each of the 6 assessed process areas. The appraisers produce the findings report by sequentially reviewing the data obtained for each of the process areas and documenting an agreed set of strengths, issues and suggested actions. The findings report takes the format of a Microsoft PowerPoint presentation that lists a set of strengths, issues and suggested actions for each of the process areas, plus a list of global observations that affected all process areas. This stage involves 2 assessors collaborating together for six hours. Therefore a total 12 person-hours of assessor time is required for both these stages.

Stage 6 (*Deliver the Findings Report*) involves presenting the findings report to the staff in the appraised organisation who participated in the interviews. This presentation involves the assessors and typically 7 company staff (this depends upon the number of the appraisal participants). The briefing normally lasts 1 hour. Therefore 2 person-hours of assessor time and 7 person-hours of company time are required for this stage.

Stage 7 (*Develop a SPI Path with the Company*) involves collaborating with staff from the appraised company to develop a roadmap that will provide guidance to the appraised company in relation to practices that will provide the greatest benefit in terms of the company's business goals. The basis for this roadmap is the Software Process Matrix [14] developed for use in small software development companies. The Software Process Matrix enables companies to base their software process improvement roadmap on: (a) their business goals; (b) maximising their return on investment; and (c) providing 'quick implementations'. This should encourage management and staff to institutionalise the SPI effort. This stage involves 2 assessors and one member of the appraised organisation working together for 4 hours, requiring 8 person-hours of assessor time and 4 person-hours of company time.

Stage 8 (*Re-assess the SPI Path and Produce a Final Report*) involves revisiting the appraised company approximately 3 months after the completion of stage 7 and reviewing progress against the SPI path that was developed in stage 7. The outcome of this stage will be an updated SPI path and a final report detailing the progress that has been accomplished along with additional recommendations. This stage will involve the 2 assessors and one member of the appraised organisation working together for 3 hours. Additionally the 2 assessors will dedicate a further 2 hours to producing a final report in relation to the assessment. Therefore 10 person-hours of assessor time and 6 person-hours of company time will normally be required for this stage. This stage is important as it provides feedback and assistance to the appraised company after a period of time. This stage also assists in compiling research material in terms of SPI experiences.

Overall, the Adept method requires approximately 58 person-hours of assessor time and 48 person-hours of the appraised organisation's time. Ideally stages 1 to 7 of the appraisal process are completed over two elapsed weeks, with stage 8 happening approximately 3 months later.

## **5 Benefits of the Adept method**

The Irish software industry contains many software SMEs driven by entrepreneurs and often lacking a quality culture. Research [2,15] has shown that software companies are often not aware of SPI models or initiatives. In such an environment it is very difficult for software organisations to appreciate the global importance of having effective software processes. Part of the problem is one of education where software development managers fail to understand how to improve their business and further fail to appreciate their company's technical performance with regard to international standards. To combat this requires an appropriate approach that will facilitate education and begin to engage software managers in a quality agenda. The application of the Adept method will help raise the level of SPI education within the appraised organisations. Also, the high-level findings report and the detailed SPI path will provide a road map for SPI within each appraised organisation. Additionally, as the Adept method requires only 6 person-days of internal staff time, this should prove attractive to SMEs from the standpoint of costs and impact upon staff working time. This is important as the primary objective of the Adept method is to stimulate process improvement programmes.

## **6 Conclusions**

The Adept method has been developed to assess software processes within Irish software SMEs based upon information that has been obtained from four different sources: (a) by reflecting upon the effectiveness of the EPA method to assess software processes in SMEs within Northern Ireland; (b) through investigating the characteristics of other lightweight assessment methods; (c) from the outcome of a meeting between researchers from DkIT, Lero and Enterprise Ireland that discussed how a culture of SPI could be instilled into Irish software SMEs; and (d) through research that has been performed by Lero and DkIT in relation to Irish software SMEs. The Adept method is designed as a lightweight assessment model to be used within organisations that have very little experience of SPI. The method relies heavily on information obtained from interviewing company personnel and performs limited cross-referencing checks (due to the limited time available for data collection and analysis). As a result, this approach depends on the willingness of the company to engage in SPI. It is therefore important that senior management within the company encourage their employees to answer interview questions in a truthful and helpful manner so that the resultant findings report will provide an accurate reflection of the company's strengths and weaknesses within each of the appraised process areas. The findings report will contain a list of recommendations. Based on these, each company must then collaborate with the assessors to prioritise these recommendations into an action plan based upon their business goals and aspirations.

We are currently planning to perform a series of software process assessments in Irish SMEs organisations using the Adept method.

### 7 Acknowledgements

This research is supported by the Science Foundation Ireland funded project, Global Software Development in Small to Medium Sized Enterprises (GSD for SMEs) as part of Lero - the Irish Software Engineering Research Centre (<http://www.lero.ie>).

### 8 Literature

- [1] F.G. Wilkie, D. McFall & F. Mc Caffery. "The Express Process Appraisal Method" ,5<sup>th</sup> International SPICE Conference on Software Process Assessment and Improvement (SPICE 2005), Klagenfurt, Austria, 27th-30th April 2005., in "Proceedings of the 5th International SPICE Conference on Process Assessment and Improvement" (Edited by Alec Dorling, Henrich C. Mayr, Terry Rout), Austrian Computer Society - books@ocg.at (Klagenfurt, Austria), ISBN 3-485403-190-4, Pages 27-36
- [2] Coleman, G., 'An Empirical Study of Software Process in Practice', in Proceedings of the 38th Annual Hawaiian International Conference on System Sciences, - Big Island, HI, p. 315c. January 2005.
- [3] ISO/IEC 15504: Information Technology – Process Assessment – Part 5: An exemplar Process Assessment Model, ISO/IEC JTC1/SC7, October 2003.
- [4] Chrissis, MB, Konrad, M, Shrum, S. "CMMI: Guidelines for Process Integration and Product Improvement", Addison Wesley, (2003), ISBN 0-321-15496-7.
- [5] Rout, T.P, Tuffley, A, Cahill, B, Hodgen, B. "The RAPID Assessment of Software Process Capability", in Software Process Improvement, ed. R.B. Hunter and R. Thayer, IEEE Computer Society Press, Los Alamitos, California, 2001.
- [6] Makinen, T, Varkoi, T, Lepasaar, M. "A Detailed Process Assessment Method for SMEs", EuroSPI 2000.
- [7] FAME (Fraunhofer Assessment MEthod) – "A Business-Focused Method for Process Assessment", Fraunhofer Institute Experimentelles Software Engineering. <http://www.iese.fraunhofer.de/fame>
- [8] Cignoni, GA. "Rapid software process assessment to promote innovation in SME's.", Proceedings of Euromicro 99, Italy, 1999.
- [9] Anacleto, A, von Wangenheim, CG, Salviano, CF, Savi, R. "A method for Process Assessment in small software companies", 4th International SPICE Conference on Process Assessment and Improvement, Lisbon, Portugal, pp.69-76 (April 2004).
- [10] Anacleto, A, von Wangenheim, CG, Salviano, CF, Savi, R. "Experiences gained from applying ISO/IEC 15504 to small software companies in Brazil", 4th International SPICE Conference on Process Assessment and Improvement, Lisbon, Portugal, pp.33-37 (April 2004).
- [11] F.G. Wilkie, D. McFall & F. Mc Caffery, "An Evaluation of CMMI Process Areas for Small to Medium Sized Software Development Organisations" Software Process Improvement and Practice Journal: Issue 10: 2, June 2005, Pages 189-201 - Wiley Publishers.
- [12] Meehan, Bridget and Richardson, Ita, "Identification of Software Process Knowledge Management", Software Process: Improvement and Practice, Volume 7, Issue 2, June 2002, pp 47-56.
- [13] Blowers, Rosario and Ita Richardson, The Capability Maturity Model (SW and Integrated) Tailored in Small Indigenous Software Industries, International Research Workshop for Process Improvement in Small Settings, Software Engineering Institute, Pittsburgh, Pennsylvania, U.S.A., October 19-20, 2005. <http://www.sei.cmu.edu/publications/documents/06.reports/06sr001.html>.
- [14] Richardson, Ita, "Software Process Matrix: A Small Company SPI Model", Software Process: Improvement and Practice, Volume 6, No. 3, September, 2001
- [15] Keane, Brendan and Ita Richardson, "Quality: Attitudes and Experience Within the Irish Software Industry", European Software Process Improvement Conference, EuroSPI05, pp 49-58, Budapest, Hungary, 9-11 November, 2005.

## 9 Author CVs

### Dr Fergal Mc Caffery

Dr. Fergal Mc Caffery is a senior research fellow with Lero - the Irish Software Engineering Research Centre. He has both an industrial and academic background. His current research interests include the development of a software development framework for the medical device industry, software process improvement frameworks and assessments, and global software development. He is a member of the BioMedIreland and MediTen steering groups in relation to software quality. He also is a member of the program committee for various International Conferences including the International Conference on Software and Data Technologies (ICSOFT 2006), the International Conference on Software Development (SWDC-REK-2005) and the International IEEE "Software Technology and Engineering Practice" (STEP 2004) Conference. He also is a regular reviewer for IEEE Software. He is also registered to participate in CMMI SCAMPI appraisals.

### Dr. Ita Richardson

Dr. Ita Richardson is a senior lecturer in the Department of Computer Science and Information Systems at the University of Limerick where she lectures to undergraduate and postgraduate students. Her main research interests are in Software Process Improvement with a specific focus on small to medium sized enterprises and on Global Software Development. She is a project leader on the GSD for SMEs project, which is funded by Science Foundation Ireland and operates within Lero – the Irish Software Engineering Research Centre. Her research involves qualitative research with companies, and some of her post-graduate students are in full-time employment with software development companies. She is also a researcher on the B4-STEP Principal Investigator project.

### Mr. Gerry Coleman

Mr. Gerry Coleman is a member of the Computing and Maths Department at Dundalk Institute of Technology (DKIT) where he specialises in teaching Software Engineering and Project Management. He is also Director of the Software Technology Research Centre (SToRC) at DKIT and a member of the Irish Software Engineering Research Centre. His research involves examining the development and evolution of software processes in small software development companies. He is a researcher on the GSD for SMEs project, which is funded by Science Foundation Ireland and affiliated to Lero – the Irish Software Engineering Research Centre. He is also researching the use and deployment of agile development methodologies in software SMEs as part of the SPACE project, funded by Co-operation Ireland.





# Lighthouse: An Experimental Hyperframe for Multi-Model Software Process Improvement

*Semih Cetin, Ozgur Tufekci, Erman Karakoc, Burcu Buyukkagnici*  
*Middle East Technical University Teknopark, Ankara - Turkey*  
{*semih.cetin, ozgur.tufekci, erman.karakoc, burcu.buyukkagnici*}@cs.com.tr  
<http://www.cybersoft.com.tr/english/index.html>

## Abstract

Unifying the process conformity with every emerging standard and improving the business processes accordingly are not trivial for organizations. Either partly supporting the enterprise or forming the whole enterprise itself, software processes are not exceptions in this respect. Being comparatively at infant stage, software processes need an intensive care in that sense since software industry is exposed to the vastly emerging standards for process improvement. This paper reveals the complexity of managing such a software development infrastructure conforming to multi-standards, and proposes a practical multi-model outlook combined with a task-oriented approach to overcome these difficulties. Besides, it also positions a conceptual hyperframe to abstract the core details of every standard from the actual software processes. Finally, the paper exemplifies the benefits of employing such a model assisted by the practical tool so-called "Lighthouse".

## Keywords

Hyperframe, Process Framework, Process Framework Compliance, Process Improvement

### 1 Introduction

Information Technology (IT) and especially the software as an essential part of it have a noteworthy impact on the quality of services the enterprises should deliver all over the world. Constituting one of the largest economies in the world, European business enterprises have duly witnessed the same so far and been employing strategies for improving the quality of both business and underlying software processes accordingly. Employing more than a million dedicated specialists in 2004, complete picture of IT and software-related services is a significant driving factor behind the European economy in its own right. Besides, by taking the added value of IT in other businesses like finance, e-Government, automotive, and logistics into account, the real share of software and IT services in European GDP is around 5% to 6% [32].

Realizing this parallelism between quality of business processes and underlying software processes clarifies that software processes have both direct and indirect impacts on all other industries and thus improving the software processes and practices will bring never-ending business opportunities to the enterprises. Nevertheless, being in an immature state and exposed to emerging set of standards for process improvement, software is still one of the most fragile parts of businesses today.

Software industry now faces several process improvement techniques ranging from methodical and formalist ones to more practical and agility-based others. From the other hand, software acquisition departments are urging further standards as well for software process unification and conformity with other business processes. So-called ISO 9001 Quality Management Systems Requirements, ISO/IEC 12207 Software Life Cycle Processes, ISO/IEC 15504, SW-CMM (Software Capability Maturity Model) and/or CMMI (Capability Maturity Model Integration), ITIL (Information Technology Infrastructure Library), RUP (Rational Unified Process), AQAP (Allied Quality Assurance Publications) and even ISPL (Information Services Procurement Library) that has its roots in Euromethod initiative are surrounding the quality management and development departments of software organizations [8]. Coping with the management of only a few of these models may annually cost from 49K to 1.202M US Dollars to a software organization [17]. Therefore, managing more than the half may even cost in geometrically increased magnitudes if it is not practically impossible to achieve.

In order to remain competitive, software organizations are expected to comply with process models, quality standards, and even contractor evaluation criteria, but the list is unfortunately growing rapidly and then turning the multitude of these standards into a sort of "quagmire" [28]. As a result, Software Engineering Process Groups in every organization usually have to focus on "yet another standard to meet" rather than the improvement areas through which organizations may get real benefits. Each organization has been unfortunately left alone in this quagmire and expected to get out by its own ways, which is a kind of "mission impossible".

The authors strongly believe that software process improvement models are more than needed for any software organization, however the addition of new standards into the list of compliance every day complicates the development, quality, and managerial activities that should be achieved by a minimal effort and without defeating any other part of the software construction. Besides, we consider that the multitude of standards should be abstracted as much as possible from the viewpoints of stakeholders, particularly the developers. Software organizations are expected to provide "single stop shopping" environments to development and quality management departments by concealing the tasks and even simple details of process improvement models being in wide variety.

This paper proposes such a multi-model hyperframe outlook combined with a task-oriented approach to get out of this multi-model quagmire. Following the identification of existing approaches, the needs for a new approach are discussed in detail. Then comes the idea of a conceptual hyperframe to veil the core details of emerging standards that software processes should comply with. Eventually, the paper exemplifies the benefits of using such a model over a practical tool so-called "Lighthouse" and ends with the conclusion.

## 2 Existing Approaches for Software Process Improvement

As software products become large and more complicated, the models and tools for Software Process Improvement (SPI) attracted more attention. Supportively, many software process assessment models have been proposed just to create stable infrastructures for benchmarking and prepare organizations for SPI. Such a vision has stimulated the use of tools for controlling software processes, which was elevated to paramount in 1987 by Osterweil's slogan: "software processes are software, too". This mainly encouraged the approach of "process programming" to define software development processes [24]. Meanwhile, enterprises like Boeing, Paramax, Bell Canada; IT companies such as HP and IBM; and even government institutions across the Europe and United States started generating their own process assessment/improvement models alone or collaboratively [3, 21]. This happened to be the early signals of boom period for the disciplinary use of SPI models everywhere.

### 2.1 Software Process Improvement Models

SEI published a report in 1987 to reveal the state of software industry. In 1993, the first version of SW-CMM was released whereas ISO started the studies for process assessment officially. Subsequently, draft versions of SPICE and ISO/IEC TR 15504 were released in 1995 and 1998, respectively. Then, CMMI version 1.0 [31] was live in 2000 and followed by the release of ISO /IEC 15504 in 2005 [27].

The parallel initiatives have spawned several activities for multitude of standards that were ended up with a sort of quagmire pointed as the "Frameworks Quagmire" by Sheard in 1997 [28]. As updated by Systems and Software Consortium [30], the picture draws existing models and interaction among them to reveal the chaos in SPI frameworks as shown in Figure 1. Besides, Sheard strictly suggested that new framework initiatives should be aware of the existing ones and mostly fit their propositions into one of them, which may at least help slow down the growth of this mess.

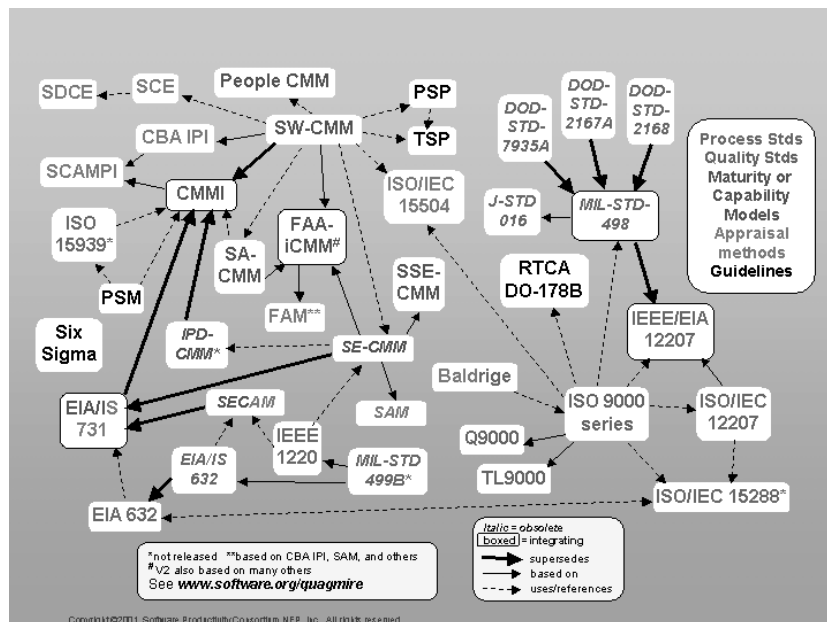


Figure 1: The Frameworks Quagmire

Likewise, Dorling proposes a quite similar approach in his recent paper to use ISO 15504 as a uniform process assessment framework for diverse business areas such as automotive, space and medicine with some specific extensions [13]. This will help in two folds: first for keeping the number of process frameworks in a certain amount and second for generalizing the processes across multi-disciplinary business domains that will enable enterprises to learn reciprocally from each other's process stability.

The authors think in parallel such that ISO/IEC 15504 will allow the use of different process reference models like ISO 12207, ISO 15288 System Life Cycle Processes and others for assisting the SPI. Supportively, two recent joint-publications pointed out that exploiting the multi-disciplinary approaches help reshape the “knowledge engineering” [1] by isolating the business and software processes and then restructuring the cross sectioned “battlefield processes” between IT and business personnel [34].

Despite the existence of studies for unifying process assessment techniques and frameworks, the number of process assessment standards that an average software company has to comply today is not negligible. Understandingly, Table 1 gives an idea for the depth of Sheard’s quagmire by means of stating the values pertaining to major assessment frameworks with their process categorization, total number of processes, process elements, and finally the number of requirements for each element.

**Table 1. Software Process Assessment Methods**

Framework	Process Category	Number of Processes	Process Elements	Number of Requirements
CMMI-SW (staged)	Process Management Project Management Engineering Support	22	Generic Goal (Required) Generic Practice (Expected) Specific Goal (Required) Specific Practice (Expected) Sub Practice (Informative)	3 GG 13 GP 47 SG 161 SP
ISO 15288	Agreement processes Enterprise processes Project processes Technical processes	25	Activity	208 Activities
ISO/IEC TR 15504-5	Customer-Supplier Management Engineering Organization Support	36	Process Attribute Base Practices Management Practices	9 PA 233 BP 34 MP
ISO 12207	Primary Life Cycle Processes Supporting Life Cycle Processes Organizational Life Cycle Processes	17	Activity Task	74 Activities 228 Tasks

## 2.2 Software Process Management Tools

First stimulated by the slogan of Osterweil [24], the need for supportive automated tools in SPI has been realized by almost every organization for more disciplinary and faster implementations. There are extensive studies on software process automation in identifying the efforts to manage software processes through a central repository. Among them, there exist tools that only coordinate the task management or perform the task management according to life cycle models [19]. Additionally, Gray and Barnes investigated workflow and related database technologies in order to extend the process automation application areas by selecting Waterfall life cycle model [7]. Workflow automation is an essential part of task execution where tasks are executed through well-defined processes. In this way, scheduling of phases and processes, and managing their dependencies require workflow engines for binding them during runtime. Therefore, automation tools should be capable of interactively defining relevant processes. In addition to process definition, the tool should also guide the process analyst to derive the executable tasks from defined processes [2].

Osterweil suggests the same by stressing the requirements to enact processes and evaluate defined processes through the aid of these tools. He also emphasizes that user interfaces should be designed well in order to take manual inputs from users that are defined in processes. He refers to a rigorous process description language for matching developer tasks with process definitions [23].

There are many other studies for process-oriented software engineering tools as summarized in Table 2. Even though every tool listed in this table supports task execution, many of them do not have the

capability of defining a discrete process structure such that organizational processes based on these structures may be mapped to different standards like CMMI and ISO 12207 later on. Also, they usually do not clearly focus on a meta-model for both software process assessment and execution or on a full "Plan-Do-Control-Act (PDCA)" cycle hiding the intricacies of multi-model SPI. With this vision in mind, the authors introduced a new hyperframe approach that is explained in the next section.

**Table 2: Process-Oriented Software Engineering Tools**

	Business Process Modeling	Organizational Process Definition	Process Categorization	Task Execution and Life Cycle Management	Hierarchic Document Relations	Tailoring Support	CASE Tool Integration	CMMI Mapping	ISO 12207 mapping	ISO 9001Mapping	Software Process Assessment	Software Process Improvement Module
<b>Aris</b>	YES	NO	NO	YES	NO	NO	YES	NO	NO	YES	NO	NO
<b>Casewis Corporate Modeler</b>	YES	NO	YES	YES	NO	NO	YES	NO	NO	YES	NO	NO
<b>Fujitsu Interstage</b>	YES	NO	NO	YES	NO	NO	YES	NO	NO	YES	NO	NO
<b>Interneer Intellect</b>	NO	NO	NO	YES	NO	NO	NO	NO	NO	YES	NO	NO
<b>Lighthouse</b>	YES	YES	YES	YES	YES	YES	NO	YES	YES	YES	YES	YES
<b>Mega</b>	YES	NO	NO	YES	NO	NO	NO	NO	NO	YES	NO	NO
<b>Scientific Software</b>	YES	NO	NO	YES	YES	NO	NO	NO	NO	YES	NO	NO

### 3 The Need for a Hyperframe Approach in SPI

As noted by Humphrey, the quality of a system or product is highly influenced by the quality of the process used to develop and maintain it [18]. Knowing that software processes are essentially forming the baseline for management and execution of business processes, the need for improving software processes then becomes an irrefutable fact. Principally stimulated by the business drivers, software customers almost always demand better, faster, and more productive process cycles for high quality products and they certainly require improved processes that can be achieved by well-defined, tracked, measured and predictable approaches.

Sometimes, such a demanding approach may need the incorporation of new standards and/or further adjustment of existing ones. As very well known, once the processes have been established, it is quite hard to change since this will complicate the design, integration, testing, and deployment as well as training the people. Consecutively, a practical hyperframe is highly needed to abstract the details from the stakeholders of software processes. The rationale behind that is summarized as follows:

- Ease in integration of emerging models and sustained compliance:** Despite the consensus on unification and internalization efforts like ISO 12207, ISO/IEC 15504 and finally ISO/SPICE, it has become a fact that new standards and models will continue to emerge [15]. Existing hybrid models including Bootstrap and Trillium incorporate ISO 9001, ISO 9000-3, the Malcolm Baldrige National Quality Award criteria, and yet other standards do not meet the needs of organizations for capitalizing ISO 9001 and CMM [5]. Contrary to the studies to determine which models, or which elements from either model, offer the most value for a software organization, they usually have to strive for new certifications and levels of maturity not only for continuous improvement, but also complying with subcontractor requirements set by large enterprises or customers directly. As the number of these standards increases, it becomes tough and very costly to manage redundant and overlapping process definitions. New versions are nightmare, too. There is an extensive ongoing research for mapping the model contents primarily by European Software Institute and Software Engineering Institute [6, 14, 15, 25] to facilitate the practitioners, however the availability of these comparisons are usually late. In fact, when they are available, manual tracing between the actual processes of organizations and these standards is infeasible. Even worse than that, whenever a

change proposal is issued, software process teams have to go over every standard and model not to violate any for sustainable compliance. Organizations are in search of an effective mechanism to map commonalities and variations of the applied standards to their own processes. A meta-model including standard mappings under the supremacy of a hyperframe will certainly eliminate redundant process descriptions and provide a baseline to map the daily tasks of stakeholders to the requirements of process frameworks [20, 26].

- **Transition from procedures to day-to-day work:** Organizations usually develop static paper-based processes that are difficult to follow and improve. Several books and articles are available on specific process models but there is a significant gap in the literature existing between model and application, and between theory and practice. Developing mature processes and enacting them are not that much easy. In fact, the main challenge of software development teams today is providing creative solutions for a complex artifact: the software. All they need should be nothing but a roadmap that will guide them in their daily work without understanding neither the processes they should accomplish nor the compliance of process models they must fulfill. The same is true for project managers, team leaders and even the customers who wish to track actual performance of the project. The current picture of process definitions detains the project teams to focus on what they are actually responsible for.
- **Rapid process deployment and decreased adaptation period:** Organizations at higher maturity levels have to dedicate vast amount of training time for newcomers. These trainings must be repeated periodically and especially whenever a significant change occurs in the process. This puts the agility of software processes away; however restricting the training may jeopardize the overall performance on the other hand.
- **Separation of concerns:** The stakeholders involved in software development have varying perspectives and requirements for information to carry out their work. They should be able to reach the information they need without having to extract it from dizzying mass of data. Each stakeholder should be assigned to work items and fed by the relevant context based on the role s/he has to play in a task-oriented manner.
- **Adherence to life cycle models:** The process management should adhere to the definition of software life cycle models and allow modification of the existing models. The tasks and processes should be matched with the software life cycle phases, which further facilitates the integration of CASE tools that are used in different phases of the life cycle. Moreover, this matching relates tasks and work products to the ongoing work in the phase-specific CASE tools, too.
- **Cut down the cost of process assessment/capability determination:** The amount of resources required to assess software organizations is another considerable issue. Process assessments are mostly realized as "expensive, time consuming, exhaustive, and disruptive to the organization being assessed" [26]. Thus, there is a need for straight mapping of the step-by-step actual work to quite a lot of process compliance models. The assessors/auditors, regardless of being internal or external to the organization, should be armed with mechanisms to easily reach supportive task lists and artifacts for reducing the operational timeframes, hence the costs.
- **Ease of metric collection and analysis:** Every process enacted once needs to be monitored in real time to identify the areas of improvement while the process is in actual use. Software teams need to check return on investment for developing the software better, faster and more cost-effectively. So, the metric collection should be abstracted by a hyperframe such that automated tools will be principally responsible for it by hiding the process execution cycles from every stakeholder, particularly from the developers.

### **4 The Multi-Model and Task-Based Conceptual Hyperframe**

Reaping the profits of mature processes together with having sustainable improvement can be mainly achieved by process automation, but not in the classical sense. Organizations should dedicate all the power and latest technologies to provide means for implanting the procedures into day-to-day work by veiling the compliance issues wherever possible. Such a tool-oriented approach can hide many details from the stakeholders through the capabilities of an inherently skillful hyperframe.

Most of the life cycle descriptions represent extremely abstract models for software development without a clear guidance on how to integrate the several process steps that project teams should perform. Curtis states that one objective of the software process modeling is to decompose process descriptions into a sufficient detail so that they can provide more explicit guidance for executing the software projects. In that sense, existing process modeling perspectives are grouped into functional, behavioral, organizational and informational. He claims that combination of these perspectives will produce an integrated, consistent and complete model of the process analyzed. He also adds that this hypothesis needs verification with practical process modeling [10].

Likewise, the authors propose a multi-model and task-based conceptual hyperframe shown in Figure 2, which fulfills the aforementioned hypothesis as follows: standards help the market in acquisition management whereas market, in return, derives business goals of software development organizations – the “why”. Hyperframe enables the management of business goals and facilitates the process engineers to define process elements in terms of tasks, artifacts and business rules – the “how” - using predefined process properties and representation models - the “what”. By doing that, hyperframe accomplishes its essential mission of abstracting the sizzling details of multi-model SPI from the rest of the picture – combination of the “who” and “how”, too.

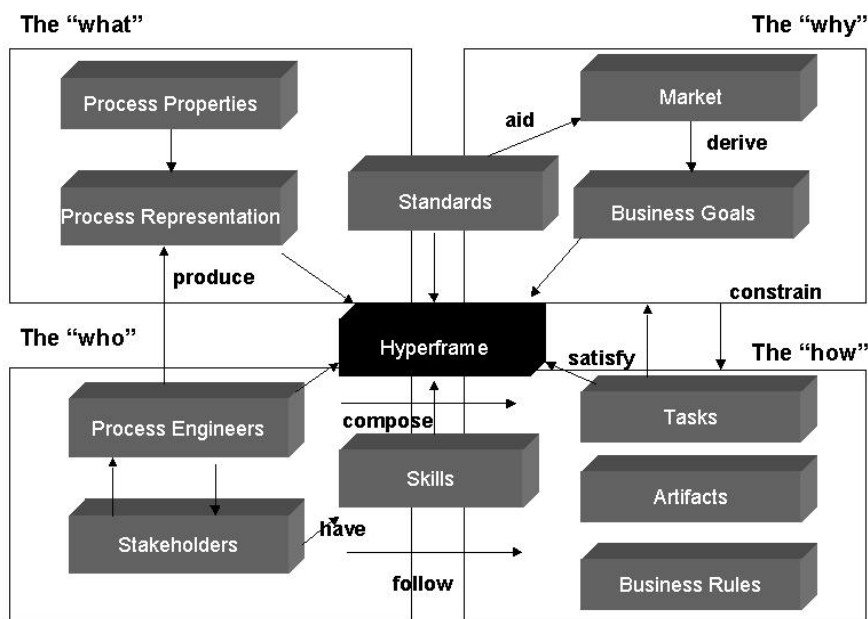


Figure 2: Multi-Model and Task-Based Conceptual Hyperframe

Stakeholders having the required skills carry out those process tasks assigned to them. They could be analysts, architects, developers, quality assurance personnel, project manager, project sponsor, and even the customer having different concerns. Eventually, hyperframe provides a different viewpoint of the environment to each one for easily accomplishing their tasks, plus the guidelines, templates as well as other mechanisms to organize, prioritize, estimate, control and improve their work.

## 5 Lighthouse: The Experimental Hyperframe

Cybersoft has established the internal process management system in 2002 including ISO 9001:2000, ISO 15504 (SPICE) and ISO/IEC 12207 standards. A year later, the company started using the tool Lighthouse as a collaboration and quality management system to improve its software processes [33]. It is the result of more than 60 man-months effort and version 2.0 is still in development/improvement.

Believing that process improvement should be continuously performed, Cybersoft staff has been trained for ISO 15504 international assessor certification. The documented method of assessments within the company is certified to be in conformance with the requirements of ISO/IEC TR 15504:1998. In June 2006, Cybersoft has attained NATO AQAP 160 certification and passed through a successful



CMMI Level 3 Class C appraisal. Lighthouse helped a lot to pass all these milestones quite smoothly.

Acting in all public, defense, Telco and finance sectors, the company faces a variety of procurement requirements including ISO 9001:2000, CMMI, SPICE and AQAP. With changing needs of the market and hence the company, Lighthouse has evolved as a practical hyperframe for multi-model software process improvement. In the mean time, Cybersoft has developed another framework, CSASSESS, to provide auditing and assessment facilities to the enterprise and/or a project of the enterprise for the quality standards like ISO/IEC 15504, CMMI, ISO 9001:2000, and plug-in support for new ones [12].

Synthesizing the expertise in enterprise-scale software development with the quality practices; Cybersoft determined the Lighthouse motto as “relieving the pain: (P)rocess re(A)lizat(I)on burde(N) of software development teams” to have them focus on their value added processes in a “single stop shopping” environment without giving up the quality targets. We believe that everybody in a software organization should only do whatever s/he is supposed to do for fulfilling the corporate mission as defined by the task assignments. Lighthouse facilitates that by supplying all the information needed whereas abstracting away the underlying complexity of the process descriptions that fulfill the standards compliance. Hence, it has an inherently capable architecture explained in the next section.

### 5.1 General Architecture of Lighthouse

Lighthouse has a 4-tier architecture depicted in Figure 3, which includes Process Composer, Process Manager, Rules Manager and Workflow Manager in principal.

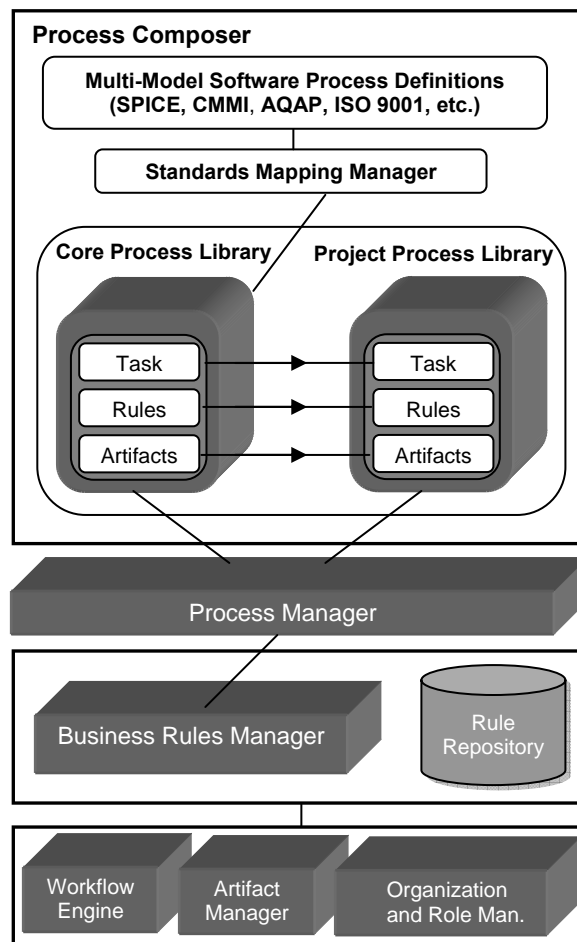


Figure 3: General Architecture of Lighthouse

The principal components of Lighthouse architecture accomplish the followings:

- **Process Composer** is the tool for structuring organizational processes. It is used to:
  - introduce the requirements of complying standards the organization selected to fulfill. The underlying ontology-based model, which will be explained more in Section 5.2, enables the definition of different process elements inherent in different standards like generic/specific practices of CMMI or activities/tasks of ISO 12207,
  - define the core processes of an organization as the combination of step-by-step tasks, business rules and artifacts. Rules and artifacts may be further associated with tasks or collaboration of tasks in Core Process Library,
  - map the requirements of complying standards to the tasks given in core process definitions, which is achieved by Standards Mapping Manager,
  - tailor the core processes to project needs according to the tailoring guidelines defined in Project Process Library.
- **Process Manager** principally manages the conversion of software processes and synchronization with workflows. It is used to:
  - associate the execution of processes with Business Rules Management module, Artifacts Management module and User Organization module,
  - enable the incorporation of a dedicated execution platform based on Service-Oriented Architecture. This is actually required since Cybersoft has an in-house Software Product Line together with an execution framework, so-called Aurora, used in almost every enterprise-scale application [11, 4]. This vision facilitates that “production” and “product” qualities can be achieved simultaneously.
- **Business Rules Manager** defines business rules within Rule Repository to assert business structure or decision logic to the behavior of task execution. Business Rules Manager is used to:
  - model the complex decision-making logic,
  - enable the separation of rules from the business logic and workflows [4],
  - encapsulate the reusable business logic for similar problem domains.
- **Artifact Manager** is responsible for managing products created or modified during the execution of a process. An artifact may be a form like corrective action request or a document such as System Requirements Specification or even a very simple data related to a task execution stored in the database for measurement purposes.
- **Workflow Engine** is used to execute processes, thus creation and termination of process and task instances, navigation between tasks, and triggering the external services. The workflow engine can be seen as a state-transition machine to manage process state changes such as resulted by the interaction of human and machine-based activities of the Lighthouse.
- **Organization and Role Manager** is responsible for defining the structure of an organization, and assignment of roles to its users. Also, it collaborates with Artifact Manager to introduce policies for processing the artifacts.

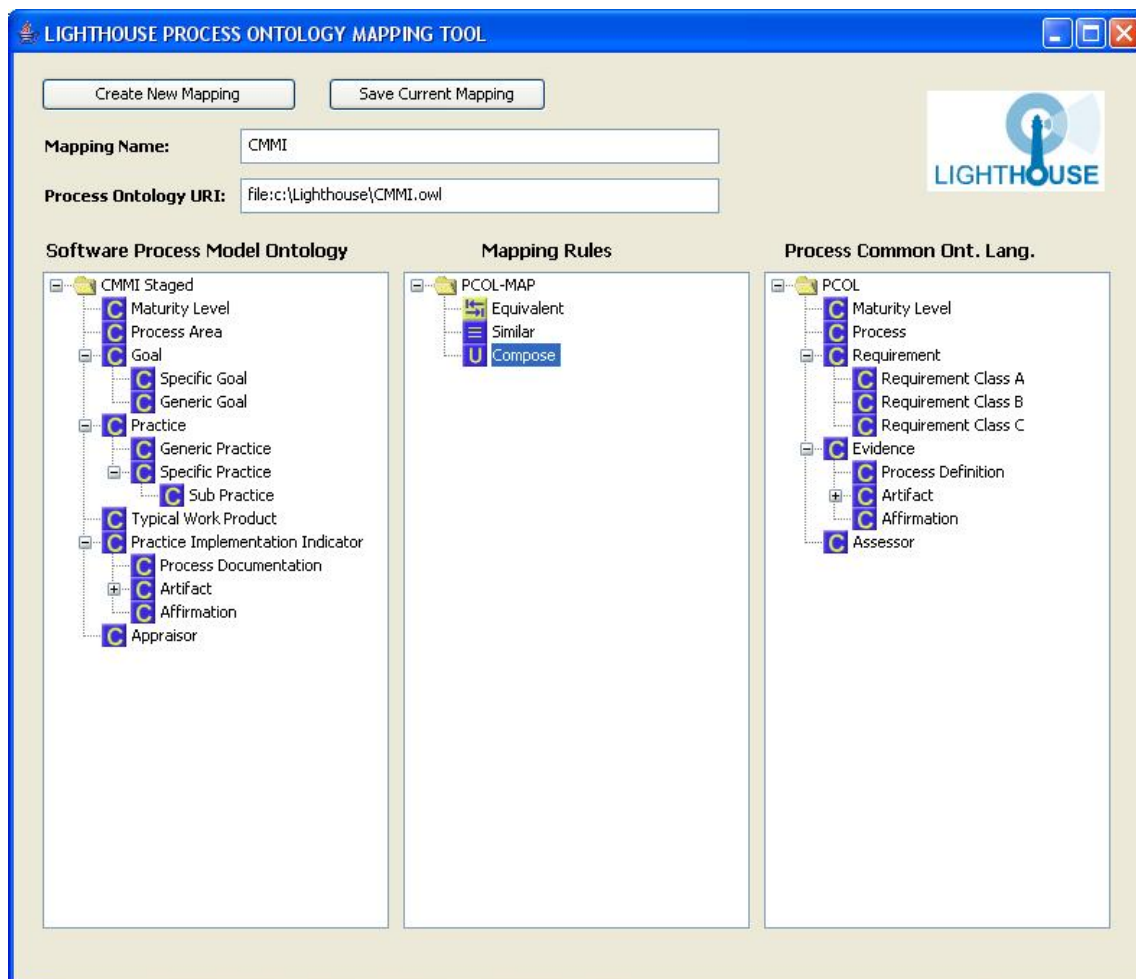
## 5.2 Standards and Use of Process Common Ontology Language

Each SPI standard has a unique model and structure having both commonalities and diversifications. Therefore, a common vocabulary to define the basic SPI model is needed to establish a relationship with the multi-model, task-oriented approach of Lighthouse. It achieves that with the ontology concept, which is known to be "a formal and explicit specification of a shared conceptualization" [15] to make each software process model pluggable. Towards that, a basic ontology so-called “Process Common Ontology Language (PCOL)” is developed to make the adoption of new process models achievable. The Web Ontology Language (OWL) is used to formalize ontologies of different process models and

## Session 7: SPI Assessment

PCOL. OWL helps to describe the relations and correspondences that allow semantic interoperability between different SPI ontologies and Process Common Ontology.

There are existing works on ontology mapping such as RDFT [22], Knowledge Management Tool [9], and MAFRA [29]. Like all these approaches, Lighthouse uses a mediator view to define the mapping relations between ontologies. Ontology mediation in Lighthouse consists of mappings of elements in different OWL ontologies. In practice, Lighthouse lets each SPI standard have a specific ontology model defined independently. An ontology mapping helper tool, PCOL-Mapping depicted in Figure 4, is then used to represent the knowledge mappings of SPI ontologies to PCOL.



**Figure 4: PCOL-Mapping Tool**

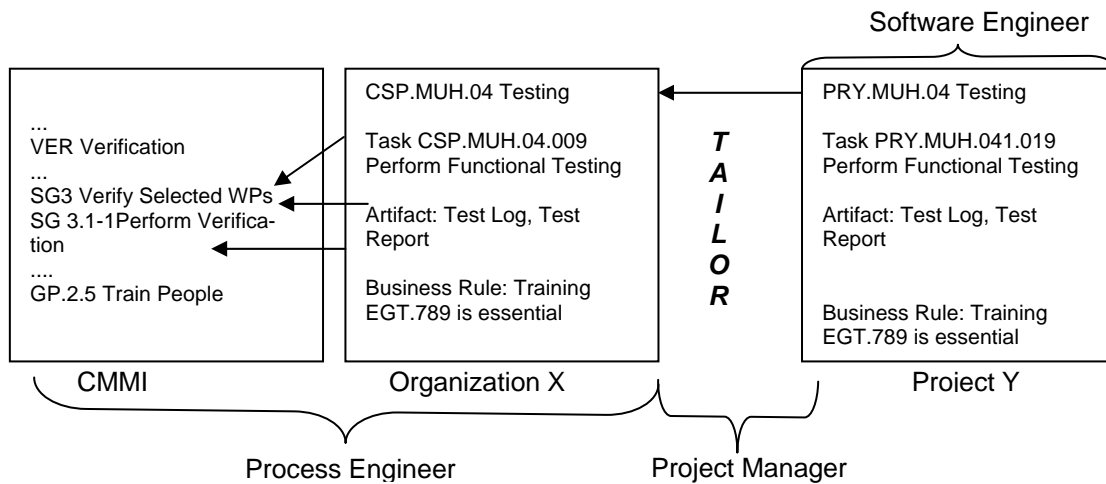
PCOL Mapping tool enables process engineers to use an externally defined SPI ontology and an in-house PCOL ontology to identify the following three available mapping types:

- **Equivalent:** An individual ontology from SPI and another from PCOL exactly define the same concept.
- **Similar:** Two ontologies have mostly common parts; hence similarities or differences can be defined in the mapping relation.
- **Compose:** One or more SPI ontologies can be associated with PCOL Ontologies. Compose mapping can also be defined as Equivalent or Similar.

Currently, mapping relations are based on <owl:Class> level in Lighthouse, however we are working on mappings based on <owl:Property> level as well.

### 5.3 Execution: How does it actually work?

Lighthouse allows organizations to define the processes in their own terms and map their process architecture to SPI models/standards. Figure 5 illustrates this capability on an example. Suppose that Organization X has defined tasks for functional testing of its software products. By using Lighthouse, process engineers may map these tasks to CMMI Process Area Verification Specific Practice 3.1-1 and ISO 12207-5.3.9 Software Qualification Testing. Organizational processes are then tailored to the projects by allowing the process engineers to shop from the set of organizational process assets: tasks, artifacts and rules. For example, imagine that Project Y of Organization X has taken the testing task and included it within the every iteration of Project Y lifecycle. Without being aware of fulfilling the SP.3.1-1 and ISO.5.3.9, testers are performing their daily activities by producing test logs and reports in this mapping. However, the artifacts they have generated will be automatically listed by Lighthouse in PIID (Practice Implementation Indicator Database) for SCAMPI (Standard CMMI Appraisal Method for Process Improvement) and sources of evidence list for NATO AQAP audit when requested.



**Figure 5: Mapping Between Standards, Organizational Processes & Project Terms**

Additionally, Lighthouse can also use existing industry-wide mappings such that when the organization process architecture has been mapped to a standard, automatic mappings are generated for these standards that are already in Lighthouse. Upcoming standards can also be added to Lighthouse with the model ontology by using PCOL and PCOL-Mapping.

Once organizational processes are defined and mapped to SPI standards by the process engineer, project managers can tailor them with respect to their specific needs. Figure 6 depicts sample screens used for task definition and workflow design steps performed by process engineers. Project manager is then expected to perform the tailoring by documenting the underlying rationale for the selected choices. After the definition of project tasks by the tailoring process, the basic utilization scenario for the software engineer is as follows: process execution is triggered by the user initiation. First task of the process instance is assigned to a user based on the type of assignment defined by process properties. Then, the assigned user would see this task in his/her awaiting task list. S/he is responsible for carrying out the required work identified by the task definition; but s/he can modify or create an artifact, or initiate an external service for completing the work as well. Subsequent to the completion of task, Process Manager triggers the Workflow Engine (in Figure 3) to manage the workflow by executing the process accordingly and creating new task instances correspondingly.

Completing the PDCA cycle, process engineers can easily identify bottlenecks and opportunities for improvement by using the metric database. Identified bottleneck points could be an input to organizational SPI plan. Moreover, since changing requirements in software development is a fact of life, the workflows of the project processes could be updated according to the information gathered from these previous phases of the software lifecycle. Lighthouse allows project managers to change and update their workflows seamlessly through integrated functionality of software iteration planning and business process modeling.

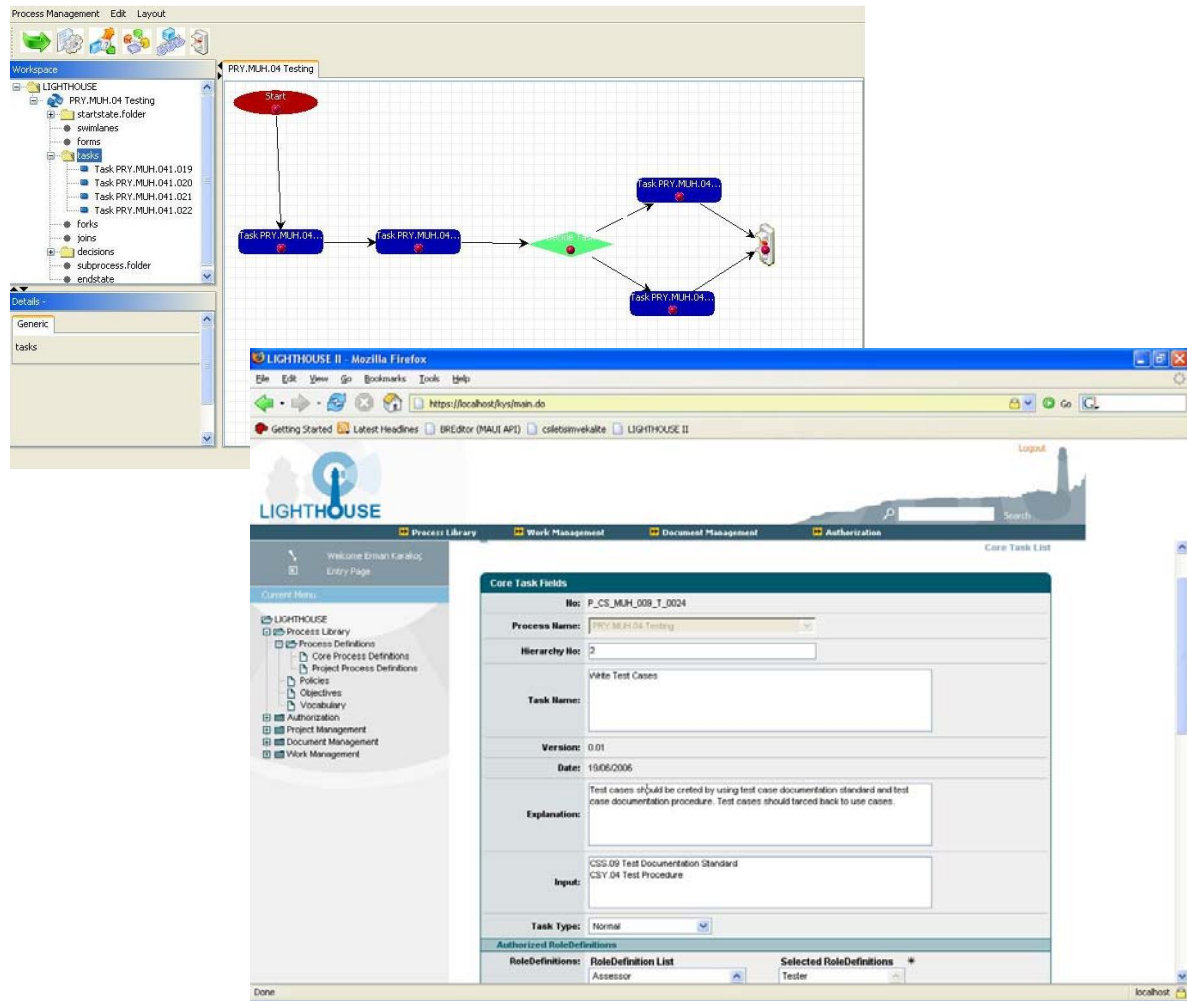


Figure 6: Sample Task Definition and Workflow Design Screens

## 6 Conclusion

In parallel to the advancements in SPI, there have been many initiatives to introduce new standards, guidelines or even frameworks, which turned the software process management into an undesired quagmire, and almost each software organization has been left alone for finding the right directions to get out. Even organizations can rarely find the right directions, they have been usually doing that using ad-hoc approaches, which inevitably complicates the management, inflates the budgets and even worse than ever tiring the development and quality management teams.

This paper discussed the complexity of managing such an environment conforming to multi-standards, and presented a new multi-model and task-oriented perspective to get out of this quagmire. Moreover, a conceptual hyperframe has been proposed to abstract the details of standards from every stakeholder. By putting this conceptual hyperframe into action with a practical tool so-called Lighthouse, the authors have witnessed that an organizationally accepted way of working tailored to the specific requirements of different projects happened to be achievable without tiresome compliance audits in traditional terms.

Furthermore, the assessment process has no longer questioned the compliance with what is actually defined and performed, not even the compliance of processes with the standards. The only requirement has turned out to be the correctness and completeness of the mapping defined in hyperframe.

Moreover, the compliance of actual works has been visible by reporting the tasks waiting in queues, which allowed auditors/assessors to focus more on the quality attributes of artifacts. Consequently, Lighthouse facilitated the organizational culture, short learning curves, domain know-how capitalization, mobility of engineering workforces and more acceptable quality budgets. Even, the appraisers have witnessed the same and shared this with us throughout ISO 9001, NATO AQAP-160 and even CMMI Level 3 Class C appraisal.

## 7 Literature

1. Aktas, Z. A., Cetin, S.: We Envisage The Next Big Thing, The Invited Keynote Speech, Integrated Design and Process Technology, IDPT-2006, <http://www.cs.com.tr/free/publications/WETNBT.pdf> (2006)
2. Allen, R.: Workflow: An Introduction, [http://www.wfmc.org/information/Workflow-An\\_Introduction.pdf](http://www.wfmc.org/information/Workflow-An_Introduction.pdf) (1999)
3. Allgood, B., Clough, A., Cunha, G., VanBuren, J., Godfrey, S.: Process Technologies Method and Tool Report, Volume I (1994)
4. Altintas, N. I., Cetin, S.: Integrating a Software Product Line with Rule-Based Business Process Modeling, Trends in Enterprise Application Architecture: VLDB Workshop, TEAA 2005, LNCS 3888 (2005) 15-28
5. Bamford, R. C., Deibler, W. J.: Crosstalk, <http://www.stsc.hill.af.mil/crosstalk/1998/09/bamford.asp> (1998)
6. Bamford, R. C., Deibler W. J.: Comparing, Contrasting ISO 9001 and the SEI Capability Maturity Model, COMPUTER, IEEE Computer Society (1993)
7. Barnes, A., Gray, J.: COTS, Workflow, and Software Process Management: an exploration of software engineering tool development, Australian Software Engineering Conference (2000)
8. Cetin, S.: An Information System Acquisition Management Model for Turkey, The Ph.D. Thesis Based on Euromethod, Middle East Technical University (1998)
9. Crubézy, M., Pincus, Z. and Musen, M.A.: Mediating Knowledge between Application Components. In, Proceedings of the Semantic Integration Workshop of the Second International Semantic Web Conference (ISWC-03), Sanibel Island, Florida (2003)
10. Curtis, B, Kellner M. I., Over J.: Process Modeling, Communications of the ACM, Volume 35, No.9 (1992)
11. Cybersoft: AURORA Software Product Line: <http://www.cs.com.tr/english/products/aurora.html> (2000 - 2006)
12. Cybersoft: CSASSESS, Web Based Plug-In Supported Process Assessment Tool Project, <http://www.cs.com.tr/english/r-d/projects-csassist.html> (2003)
13. Dorling, A.: Modern Approaches for Software Process Assessment: Standards, Models and Applications, The Foundation of System and Software Engineering, Moscow, To Appear in April (2006)
14. Feller P., Gallo, E.: SPICE-CMM Mapping Version 1.0, European Software Institute (1998)
15. Garcia, S.: Evolving Improvement Paradigms: Capability Maturity Models and ISO/IEC 15504 (PDTR) in Software Process Improvement and Practice, Volume 3, Issue 1, Wiley/Gauthier-Villars (1998)
16. Gruber, T.: A Translation Approach to Portable Ontology Specifications, Knowledge Acquisition 5 (1993)
17. Herbsleb J., Carleton A., Rozum J., Siegel J., Zubrow D.: Benefits of CMM-Based Software Process Improvement: Initial Results , CMU/SEI-94-TR-013 (1994)
18. Humphrey, W. S.: Managing the Software Process, Addison-Wesley, Reading, MA (1989)
19. Hutchens, K., Oudshoorn, M., Maciunas, K.: Web-Based Software Engineering Process Management, IEEE (1997) 676-685
20. ISO/IEC 15504-5:2006: Information technology -- Process Assessment -- Part 5 An exemplar Process Assessment Model (2006)
21. Kitson, D. H.: An Emerging International Standard for Software Process Assessment, <http://www.sei.cmu.edu/iso-15504/resources/PapersBriefings/ISESS97.PDF> (1997)
22. Omelayenko, B.: Integrating Vocabularies: Discovering and Representing Vocabulary Maps. In Proceedings of the First International Semantic Web Conference (ISWC-2002), Sardinia, Italy, June 9-12, 2002

## Session 7: SPI Assessment

23. Osterweil, L. J.: Automated Support for the Enactment of Rigorously Described Software Processes, ISPW 1988 (1998) 121-125
24. Osterweil, L. J.: Software Processes are Software Too: Revisited; An Invited Talk on the Most Influential Paper of ICSE 9, ACM Press (1997) 540-548
25. Paulk, M. C.: A Detailed Comparison of ISO 9001 and the Capability Maturity Model for Software, Software (1994)
26. Richardson, I.: Quality Function Deployment - A Software Process Tool, Third Annual International QFD Symposium (1997)
27. Rout, T., Dorling, A.: ISO/IEC 15504 and the SPICE Network, A Status Report (2004)
28. Sheard, S.A.: The Frameworks Quagmire, Crosstalk, Volume 10, Number 9 (1997) 17-22
29. Stumme, G and Maedche, A.: (FCA-Merge) Ontology Merging for Federated Ontologies on the Semantic Web, Proceedings of the International Workshop for Foundations of Models for Information Integration (FMII-2001), Viterbo, Italy, September (2001)
30. Systems and Software Consortium: The Frameworks Quagmire, <http://www.software.org/Quagmire/>
31. Software Engineering Institute: Capability Maturity Model Integration (CMMISM), Version 1.1 (2000)
32. The S3 Initiative: European Vision for Software, Services and Systems, Vision Document (2005)
33. Tufekci, O.: Lighthouse: A Practical Quality Management Information System, Workshop Proceedings, IMPROQ 2004, <http://trese.cs.utwente.nl/workshops/improq2004/Papers/workshopReport-Improq-2004.pdf> (2004)
34. Tufekci, O., Cetin, S., Altintas, N. I.: How to Process [Business] Processes, Accepted Paper, Integrated Design and Process Technology, IDPT-2006, <http://www.cs.com.tr/free/publications/H2PP.pdf> (2006)

## Author CVs

### Semih Çetin

He was born in 1968. Dr. Cetin received B.Sc., M.Sc. and Ph.D. degrees from the Department of Computer Engineering at Middle East Technical University. He has previously worked as a software engineer, consultant, R&D and project manager at the beginning of his career. In 1995, he has co-founded Cybersoft Information Technologies and still works in the company as a managing partner mainly responsible for improving the technology base. Cybersoft has been awarded by Computerworld Honors in 2003 for the successful implementation of Internet Tax Office project for Ministry of Finance.

His technical interests include information systems development methodologies, software process management, object-oriented design, aspect orientation, middleware technologies and software architectures. He is a part-time instructor teaching graduate courses in computer engineering departments at Middle East Technical University and Cankaya University.

### Özgür Tüfekçi

Mrs. Tufekci was born in 1973. She has graduated from Industrial Engineering Department of Middle East Technical University in 1995. She has received her M.Sc. degree in 1997 from the Industrial Engineering Department at Bilkent University while working as a research assistant in the same department. Later, she moved to industry and started working in R&D Department of ASELSAN Communications Division.

She has joined Cybersoft in 1998 and still works as Engineering Processes and Quality Group Manager in the company. Mrs. Tufekci is a Ph.D. candidate in Informatics Institute at Middle East Technical University.

### Erman Karakoç

Mr. Karakoç was born in 1982. He has graduated from Department of Computer Engineering at Middle East Technical University in 2004. He is M.Sc. candidate in the same department with the thesis entitled by "Web Service Composition Planning under Resource Allocation Constraints". Mr. Karakoç has been working as a software engineer in the Lighthouse project since 2004.

### Burcu Büyükkağnıcı

She was born in 1982. She has graduated from Industrial Engineering Department at Middle East Technical University in 2003. While being a M.Sc. candidate in Software Management program of Informatics Institute at Middle East Technical University, she has been working for Cybersoft as a quality and process engineer since 2004.





# Model-Based Test Design

*Dr. Anne Kramer*

## **Abstract**

The world is rapidly becoming more speedy and complex. What sounds like a commonplace is particularly true for integration and quality assurance of IT products. Various tools and methods have been established to support development. For functional testing, however, manual document-based tests are still standard.

Model-based test design is an interesting alternative to document-based tests that allows us to react fast and efficiently on changes. Model-based test design is mostly used in test phases close to development (unit and integration tests). It is also often associated with automatic test case generation.

In this paper we will explain how we use this method in all test phases including system test and what advantages we obtain from it. We will show some quantitative evidence of our – throughout positive – experiences in larger customer projects with and without automatic test case generation.

## **Keywords**

model based test design, MBT, validation and verification, quality assurance

### 1 Introduction

In the IT business, like everywhere else, time-to-market has become the most important factor for market introduction of a product. Increasingly complex and heterogeneous software products have to be delivered in less and less time and, preferably with fewer costs and reduced maintenance effort. Development and test under these conditions have become a real challenge. In reaction to this, iterative and agile processes are used. While these processes increase efficiency in development, they tend to produce high numbers of iterations which, in turn, require rapid and efficient regression tests.

There are processes, tools and methods to cope with this challenge. But even the processes themselves have to be flexible and must be adapted continuously to guarantee quality of the products and cost efficiency of the process itself. For SW development numerous approaches exist. Two examples are model-driven architecture (MDA) and reuse of modular object oriented components. For test, however, the number of applied methods is usually restricted. In the safety-relevant areas of IT development (e.g. for medical devices), two methods are still standard:

- automated unit tests and
- document-based manual system tests.

While these methods are well established, they require a high amount of maintenance effort, especially when dealing with short iteration cycles. As a result, it becomes difficult to maintain the level of quality and efficiency required.

In this paper we will present an alternative approach, i.e. “model-based test design”. The concept of model-based testing (also known under the acronym “MBT”) is not particularly new. Information can be found in reference [1] and related links. However, the use of model-based test design is still not that widespread as could be expected, especially in later test phases. In fact MBT is often associated with test case generation and re-use of existing design models. Several vendors propose tools for test case generation (see [2], [3]). In this paper, we will show how we use model-based test design with and without automatic test case generation and in various test phases. We will also present some of our experiences in larger customer projects and show quantitative evidence of the advantages and profit obtained with this method.

### 2 Establishing Test Design Models

First, we analyze the product requirements. New requirements can be change requests, results from risk or impact analysis, or simply known bugs. All requirements are associated to use cases, where they can be traced for completeness (requirements tracing). These test use cases focus on one particular test aspect, e.g. one specific workflow of the system.

Now, a model is drawn for every test use case. This can be done in different ways. Since unit and integration tests are very close to implementation, the models are best realized as state diagrams or activity charts. For every test use case a state and/or activity diagram is drawn, taking into account all alternatives and paths. This diagram contains the main workflow, completed by additional and exceptional paths, reflecting the tester’s mindset. If required, it is also possible to define pre- and post conditions of the use cases or to include test data into the model. We usually write our test design models using IBM Rational Rose (for reasons that will become clear soon), but the method itself is completely tool-independent.

If available, existing SW design models can be reused from the development department. However, it is important to realize that development models focus on other aspects than test models. Test cases derived from a development model focus on the verification of functionalities (“Does everything work as specified?”). Test cases obtained from a test model will rather be used for validation (“Does the product fulfil the expectations?”). Therefore, development models are best used for unit or integration tests.

In “later” test phases (e.g. system tests or customer acceptance tests) the test focus lies rather on the complex dynamical behaviour of integrated or deployed systems. For these test phases, specific test models should be written. This can be done without specific knowledge of the implementation of the system. It turned out to be helpful to use message sequence charts as test design models.

Once the test design model is completed (i.e. all state / activity diagrams or message charts are drawn) it is reviewed together with the stakeholders. At that occasion, one of the major advantages of this method becomes strikingly obvious: the test design model is – above all – a communication platform. Pictures are understood much faster and easier than text. The structured approach allows a higher level of abstraction. As a consequence, inconsistencies or incomplete requirements are discovered easier during review (and even before). It is also easier to prioritize test cases.

### 3 Test Case Implementation

The test cases are now derived directly from the test design model. Every test use case corresponds to a set of test cases that focus on one specific test goal: the combination of transition paths that cover a specific workflow. The single test case corresponds to a possible path that the system may take through the state / activity diagram or the message sequence chart for the specific use case.

Test cases can be obtained either manually or automatically. If done manually, the test case author just follows all possible paths through the diagram and writes down the corresponding actions and verifications points. Using the graphical approach, it is easy to prioritize the branches of the test model and, thus, to focus on specific parts (e.g. extended tests for high risk components, non-regression tests for standard functions).

As has been said in the introduction, various tools for automatic test case generation exist on the market. We have also developed our own product called *.getmore* ([3]) that is best adapted to the tools and environment of our customers. Using *.getmore* we obtain test cases from state and/or activity diagrams that are written with an IBM Rational Rose. We can choose between different strategies, i.e.

- full path coverage,
- full transition coverage,
- random or
- user guided paths.

The path length is configurable. If pre- and post conditions have been defined, they are automatically taken into account.

Once the test cases have been generated, they can be exported and imported as manual test cases into various standard testing tools via standard interfaces (\*.csv) or API. Possible export formats of *.getmore* are MS Excel (\*.csv), IBM Rational TestManager or Mercury TestDirector. For proprietary tools (e.g. *.faust* of sepp.med) it is possible to generate test scripts for test automation.

### 4 Advantages

One of the major advantages of MBT has already been mentioned above. Due to the graphical representation, a higher level of abstraction can be obtained. Since the entire diagram is reviewed rather than single test cases, it becomes easier to understand the relations and dependencies between requirements. Inconsistencies and incomplete specification are identified rapidly. It is also possible to determine the completeness of the test model and to prioritize paths. As a result, a higher quality of test design is obtained. Moreover, the degree of quality becomes measurable. A possible metric can be the path coverage during test execution.

The second main advantage concerns maintenance. Test cases that have been obtained using model-based test design can be maintained very efficiently, since redundancies are eliminated. If one

activity has to be modified, this change is made at one central location in the test design. If we use a test case generator, these changes are then introduced automatically in all test cases that have been derived from the modified use case. But even without automatic test case generation, the test cases that have to change can be identified straightforward. This procedure reduces the number of errors that might have occurred if no test model had been available.

Using test design models improves traceability. It is easy to retrace why test cases have changed and where the new requirement came from. Moreover, test design models are very helpful when you have to decide on the scope of partial tests. From the model it is obvious which test cases have been affected by changes and, therefore, have to be included in the regression tests. All these automatisms systematically help to avoid errors in test specifications and improve the overall quality.

Finally, we obtain a more homogeneous output from the model. Activities are described in the same way throughout all test cases. This makes it easier for the tester to understand the test steps, but it also helps to communicate with a larger number of stakeholders from different backgrounds.

## 5 Experiences

We have used model-based test design in different projects, both with and without the test case generator *.getmore*. Our experiences with this method are throughout positive. We obtain a high degree of efficiency and quality. The amount of errors is considerably reduced due to automatisms and the traceability is improved. The latter is of particular importance if dealing with safety-relevant processes (e.g. FDA standards). Moreover, using MBT we can react more flexibly, allowing us to handle complex products and processes. Just think of the development of entire product families and the related testing effort.

Obviously, it is difficult to give a quantitative estimate of saved time and money, since we have never used the model-based and the “classical”, document-based approach together in one project. However, we can clearly identify the tendencies. These depend on the test phase for which test design models have been used.

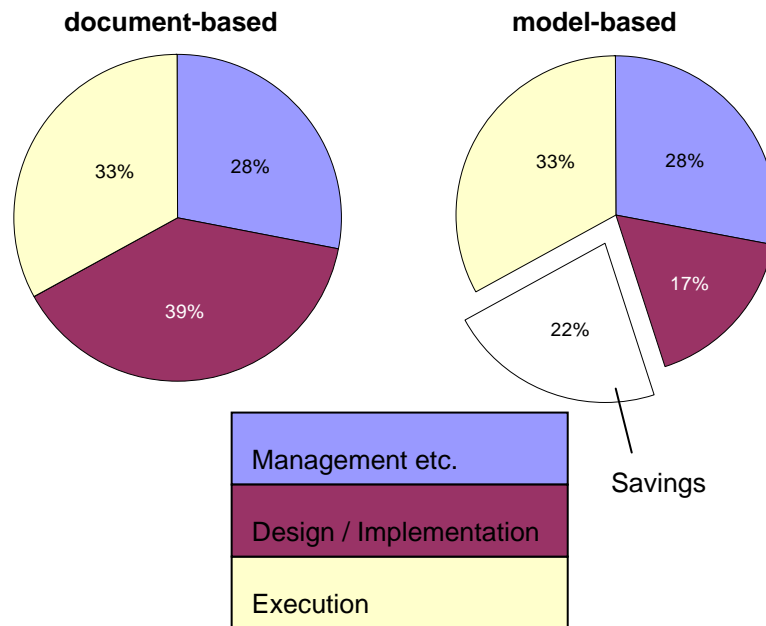
### 5.1 Unit and integration tests

The test design and implementation of a unit or integration test model, if done for the first time, requires more effort than the direct implementation of test cases. This is due to the fact that the test design for these test phases is described in a highly formal way. Every step is described in detail using an exact and dedicated semantical approach. Especially, if a test case generator shall be used, the test model has to be very exact, since test cases shall not be modified at all after generation. We plan about 25% additional effort. However, – and this is extremely difficult to quantify – the test model usually covers more aspects and we obtain a higher quality of the tests.

The main gain is obtained during maintenance of existing test cases. Changes are made at one single location in the test model and then reported into the derived test cases. Moreover, once the initial model has been agreed at, it is easier to communicate on changes. We obtained up to 75% effort reduction in the maintenance phase. Typically, we deal with long living projects where maintenance efforts sum up to 400% or more of the initial effort. For these projects the test design model becomes particularly interesting.

## 5.2 System test or customer approval

Later test phases such as system test or customer approval typically focus on a top down approach with requirements/workflow and deployment driven description less bound to semantical details. In other words, the test design model does not have to be perfect to start with. We use iterative procedures to elaborate the final test design model.



**Figure 1: document-based vs. model-based tests**

For these test phases, experience shows that the return on investment can already be attained in the first test design cycle. Figure 1 shows the comparison of two example projects we worked on with our customers. These projects have been chosen, because they are comparable with respect to overall effort, amount of new requirements, duration, environment and complexity. In one case, we used model-based test design, whereas the other project was conducted with document-based tests. As can be seen from the charts, we saved more than 50% effort during test design and implementation, and without automatic test case generation.

When introducing MBT to test case authors that are not yet familiar with the method, it takes some time for them to get used to the models. Once the meaning of the symbols is clear, reading test design models becomes somehow intuitive. The method is rapidly accepted by managers, because it is possible to get a rapid overview on the test design in short time.

Writing a good test design model, however, is an art where experience is important. Nevertheless, the increasing success of MBT proves that it is worth investing in it.

### 6 Summary

We cannot report any negative experiences with model-based test design, simply because we do not have them. In some cases, we would simply not be capable to manage the increasing complexity of the products under test without model-based test design.

Unlike most industrial references where MBT is used for test phases close to development (i.e. unit and integration tests), we also use MBT successfully for system tests in larger customer projects. We start writing system test models at an early stage and refine them iteratively. This procedure would not be efficient with a document-based test design.

Using test design models considerably improves the test quality, because the approach is understandable, structured and straightforward. The number of possible errors is reduced (especially if working with the test case generator *.getmore*). Traceability is improved, a fact that is of particular importance for products that have to fulfil standards of the American Food & Drug Administration (FDA).

We obtain a considerable increase in efficiency which gives our customers strategic advantages. Test design and implementation using test models require up to 50% less effort compared to the “classical” document-based test. The method is particularly interesting, if the maintenance effort is large compared to the initial effort, but it can even pay off immediately.

## 7 Literature

- [1] <http://model-based-testing.org/> (Link called on May 30, 2006)
- [2] <http://www.goldpractices.com/practices/mbt/index.php> (Link called on May 30, 2006)
- [3] <http://www.seppmed.de/index.php?id=65> (Link called on May 30, 2006)



### 8 Author CVs

#### **Dr. Anne Kramer**

1967 born in Bremen (Germany)

1986-1992 Studies of Physics at the University of Hamburg (Germany)

1992-1995 PhD at the University Joseph Fourier in Grenoble (France)

1996-2000 working for Schlumberger Systems in Paris (first as software developer for smart card tools, than as project manager for point of sales terminals)

Since 2001 technical project leader at sepp.med (near Erlangen, Germany)

The sepp.med GmbH has specialized on IT solutions with integrated quality assurance in complex, safety-relevant domains. A main aspect of the consultant activity lies in introduction and support of processes, methods and tool-based automatisms.



# Project Management based on Effective Practices: an Alternative Framework to SMEs<sup>1</sup>

Cuevas, Gonzalo., Calvo-Manzano, Jose A., Garcia, Ivan. & San Feliu, Tomas.

Languages and Informatics Systems and Software Engineering Department  
Faculty of Computer Science, Polytechnic University of Madrid. Spain  
{[qcuevas@fi.upm.es](mailto:qcuevas@fi.upm.es), [jacalvo@fi.upm.es](mailto:jacalvo@fi.upm.es),  
[igarcia@zipi.fi.upm.es](mailto:igarcia@zipi.fi.upm.es), [tsanfe@fi.upm.es](mailto:tsanfe@fi.upm.es)}

**ABSTRACT.** This paper aims to show an alternative metamodel to improve the project management process. This work is based on effective practices through the analysis and study of relevant models and standards related to project management. It is true that there are many models that improve the project management effectiveness, but most of them are focused only on two processes: Project Planning and Project Monitoring and Control, neither of which is affordable for small and medium enterprises. Our study proposes a new metamodel to achieve higher performance levels in these types of organizations.

**KEYWORDS:** Software project management, software process improvement, effective practices models, repository of assets, processes, small and medium enterprises.

## 1 Introduction

In recent years we have been witnessing the increasing demand for software to solve more and more complex tasks and greater added value [29]. Under these circumstances, we can ourselves the following question: Is the software industry prepared to deliver the software that is needed in the coming years, according to client demands? Unfortunately, according to the Prosoft Foundation (Program for Develop the Software Process) [30] and researchers as Oktaba [25], Brodman [2], and Carreira [4] the answer is no. The software development process is very far from being a mature process.

At the moment, there is a consensus in the software industry sector that a complex product like software must be developed with the help of the engineering and management process and metrics that enable us to effectively predict the risk levels of software products (in terms of costs, schedules, and defects primordially) [8]. In spite of the benefits that this area promises, the percentage of project success has been generally poor although it is improving [34].

But the truth is that the IT projects usually fail partially and, sometimes, completely [13]. The "software crisis" of 1969 has lasted up to the present, with the same old causes of project failure [18] [34]:

- The delivery of software is delayed,
- The software fails due the low quality,
- The maintenance of software absorbs the resources assigned, and
- Projects are abandoned or the costs are excessive.

The studies related to project failure causes are very interesting for our analysis. These studies have been published as technical reports or study cases in conferences and/or international specialized issues.

There are studies [33] [5] which have evaluated the causes of failure of software projects. The lack of management is confirmed in the CHAOS Reports where analysts of the Standish Group indicate that the main causes of project failures are the lack of efficient project management and the executive support. According to the 2004 Third Quarter Research Report [34], a large number of projects fail.

Around the world a million of projects are executed every year, Cairó [3] indicated that a third of these projects, exceed 125% in time and cost. But, why so many failures? The same study indicates that although many reasons exist, one of the most important is project management.

---

<sup>1</sup> This work is sponsored by Endesa and DMR Foundation Consulting through the "Research Group of Software Process Improvement in Latin America".

Jones [17] has identified three principal causes of failure and delays in software projects: *inaccurate estimate, poor communication of project status, and lack of historical information*. These are key issues in project planning and project monitoring and control areas. Jones also affirms that these causes can be eliminated through an adequate project management process.

A recent DoD report on this problem states that: “*After two decades of unfulfilled promises about productivity and quality gains from applying new software methodologies and technologies, industry and government organizations are realizing that their fundamental problem is the inability to manage the software process, the low quality in the risk management area specially*” [1].

To address these problems, almost all research and applications have been developing methodologies and tools like Primavera Enterprise 3.0 [27], OpenPlan Professional [26], Microsoft Project 2003 [24], PMLP [12] and EPM [7], which focused exclusively on technological aspect of projects. These initiatives include the following process:

- **Project Planning:** The establishment and maintenance of plans that define the project activities with its estimations and schedules.
- **Project Monitoring and Control:** The understanding of project progress to take adequate corrective actions when significant deviations from the plan are generated.
- **Management and Follow-up of Risk:** The early identification of potential problems that permit the planning and execution of solutions during the project life.

However, these initiatives have not considered aspects like:

- **Definition of Project Management Processes:** The establishment and maintenance of a valid set of organizational process assets: *standard project management process, description of life-cycle models, guidelines and criteria for tailoring the standard project management process, repository of organizational measures, and the repository of assets*.
- **Improvement of Project Management Processes:** The definitions of practices that help the organization to plan and implement the process improvements, taking into account the strengths and weakness of the organization’s project management processes asset library.
- **Quantitative Project Management:** The definition of quantitative models, which enable project managers to predict the future behavior of projects through the use of organizational historical information and provide improvement information.

So, it is necessary to take all these processes into account when defining and implementing the project management processes.

## 2 Motivations

From beginnings of the 90’s, industry and researchers interested in Software Engineering have been expressed a special interest in Software Process Improvement (SPI). An indicator of this is the increasing number of publications, and the development of international initiatives related to SPI, such as CMMI [32], ISO/IEC 15504:2004 [16], SPICE [15], and ISO/IEC 12207:2004 [14].

As a result of this increasing interest in SPI, many methods for implementing improvements such as SCAMPI [31], ISO/IEC 15504:2004 [16], IDEAL [21], CBA\_IPI [6] have been developed.

This interest in software improvement in large enterprises is now extending to small enterprises. However, the problem is the high implementation cost, independently of the size of the company [9] [10] [23].

As models have been developed for large enterprises, only a few SMEs know them. Even in the case where a small enterprise knows the models and recognizes the need to improve the processes, the principal disadvantage is its resources (financial and human) which are limited [6]. Kuvaja [20] and Kilpi [19] considered the size of investments to be the principal disadvantage that SMEs may face when trying to adopt a model [28]. Thus, it is necessary to adapt existing models to small and medium enterprises.

In Spain, in January 2003 there were 3 million SMEs that make up 99.87 % of all companies. Similarly, more than 99% of Mexican enterprises are micro, small, and medium; and they concentrate the 50% of all country entrance. With this information we could determine the importance of SMEs at macroeconomics level (see Table 1).

**Table 1.** Characteristics of the Spanish and Mexican enterprises (Source: DIRCE 2003, Secretary of Economics 2005)

Spain					
Micro-enterprises	Small (10-49)	Medium (50-249)	SMEs (0-249)	Big (250 and more)	Total
2.642.775	145.418	21.192	<b>2.809.385</b>	3.735	2.813.120
93,94%	5,17%	0,75%	<b>99,87%</b>	0,13%	100%
Mexico					
Micro-enterprises	Small (21-50)	Medium (50-249)	SMEs (0-249)	Big (250 and more)	Total
2.722.003	88.173	25.599	<b>2.835.775</b>	8.533	2.844.308
95,7%	3,1%	0,9%	<b>99,7%</b>	0,3%	100%

### 3 The PROMEP Framework

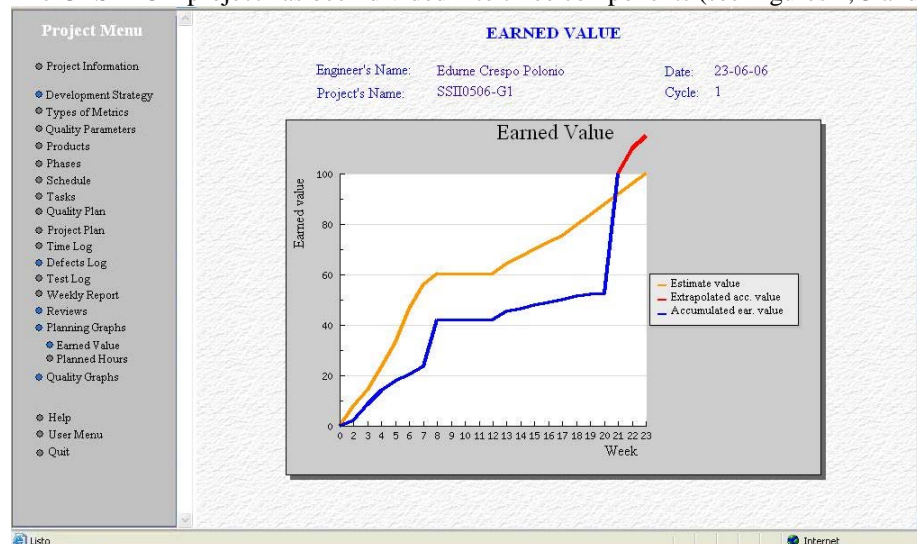
As an alternative solution to these problems, we are proposing the definition of a new PROject Management framework based on Effective Practices (PROMEP) that will be affordable for SMEs. The aim of the framework is to define, design and implement the Project Management Process conforming to CMMI capability level 4 (CL4). In order to achieve the CL4, it is necessary to define for each project management process the following components: *process definition including activities, process and product measures, quality aspects, techniques to be used, templates, checklists and tools to support the process.*

We have evaluated software project management approaches and we have concluded that TSP (Team Software Process) [11] may be a good solution for SMEs. In fact, TSP shows remarkable coverage with respect to most of the process area in the project management category [22].

The tool that supports TSP is an excel spreadsheet but there is not a valid tool for SMEs. Then, we focused our efforts on developing an alternative solution to improve and control the project management processes based on TSP effective practices. We have developed a web tool called GESPROY.

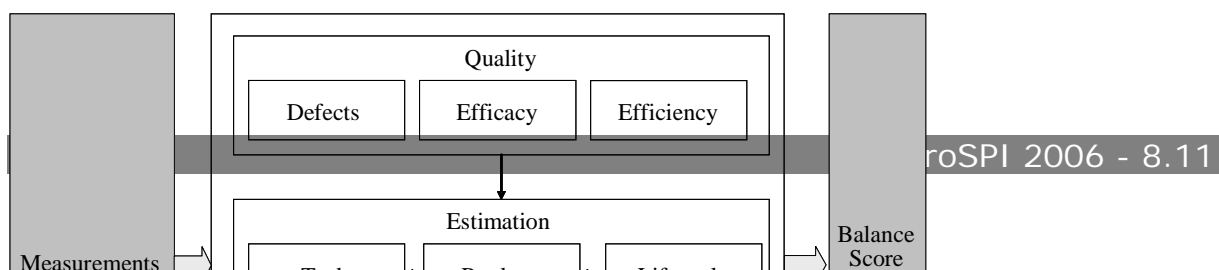
With this tool any manager can control his projects and thereby achieve the required productivity (see Figure 1).

The GESPROY project has been divided into three components (see Figures 2, 3 and 5).



**Figure 1.** The GESPROY tool

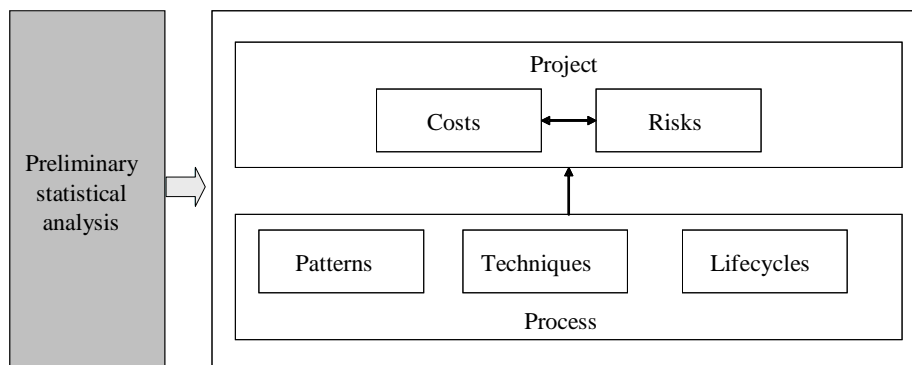
The first component is the *“planning, and monitoring and control” component*. This component will be supported by a tool that will allow project planning, and monitoring and control. This component will include the following modules:



**Figure 2.** Planning, and Monitoring and Control Component

- **Estimation module.** In this module, the product and task estimation will be defined. Also, the life cycle of the project should be selected.
- **Scheduling module.** This module will be in charge of defining the starting and ending dates for the project and the effort that will be dedicated each week.
- **Monitoring and control module.** This module will be in charge of recording the data collected during the project execution. It will be needed to define the indicators, monitoring parameters and thresholds. Also, updating the plan when deviations are significant is allowed.
- **Quality assurance module.** This module will calculate quality indexes in order to evaluate the process and product quality. The quality indexes will be calculated from the defect data collected through the life cycle. This module will analyze the quality activities efficiency and will provide the injected and removed defect rates in order to verify that the quality target will be achieved.

The second component is the “*financial and assets*” component. By using this tool, it will be possible to have the financial control on the project. In addition, the organization will be able to be managed the knowledge. This component will include the following modules:



**Figure 3.** Financial and Assets Component

- **Asset repository module.** This module will store the different life-cycle definitions including the project management processes. The life-cycle definitions will define the new projects. We use the UML language to represent the processes (see Figure 4).
- **Cost module.** This module will allow to have control on the cost of the projects, as well as to evaluate the economic performance obtained by the organization.
- **Preliminary statistical analysis module.** This module will allow a vision of the project. The decisions on the basis of the analyzed data will be taken.

- Risk module.** In this component the definition of different risk categories will be carried out. The risk module will provide information to the scheduling module in order to make the decisions based on the risk information.

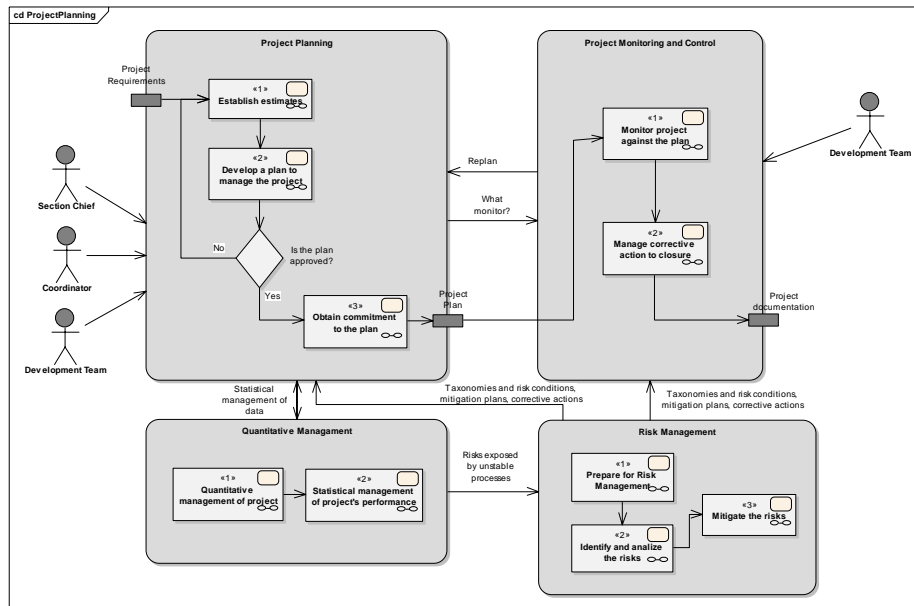


Figure 4. Process definition example

The third component is the *“balance score board” component*. With this component, we try to get the statistical control of the organization portfolio. This component will include the following modules:

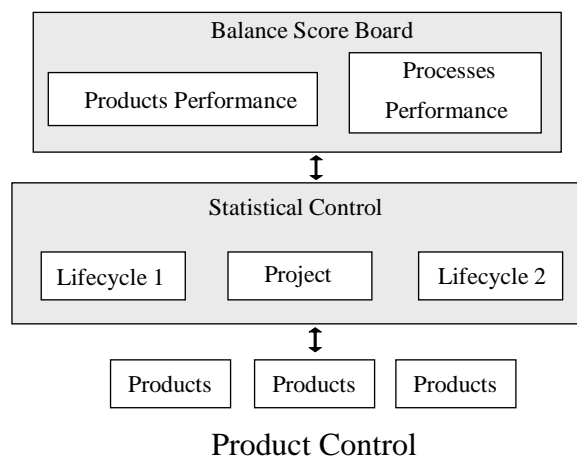


Figure 5. Balance Score Board Component

- Balance score board module.** This module will allow the integrated management of the organization portfolio. In this way, the results of the processes and products performance could be used as a reference for the new projects.

- **Statistical control module.** This module will allow the comparison of the results from the different processes of the life-cycles used. The statistical analysis will calculate the thresholds to be used for the management parameters.
- **Product control module.** This module will allow the comparison of the results from the different products of the life-cycles used.

### 4 Conclusions

One handicap for SMEs is to fill the current gap between CMMI and the operational deployment. The aim of PROMEP is to help SMEs to overcome this gap permitting SMEs to implement project best practices in their companies. A web tool supporting PROMEP will increase the facility to institutionalize the effective management project practices in SMEs. It is very important to highlight that this framework will introduce and institutionalize the effective project management practices in SMEs in an affordable way.

Our research work in this area is to define an interoperable process repository that can be used by SMEs to interchange information about best practices. Once it has been experienced in a large number of organizations, we will focus on obtaining best practices from the process repository.

### 5 References

1. Brock, S., Hendricks, D., Linnell, S. & Smith, D. "A Balanced Approach to IT Project Management". ACM Publications. Proceedings of SAICSIT. 2003.
2. Brodman, J. & Johnson, D. "Project Planning: Disaster Insurance for Small Software Projects". LOGOS International, Inc. Proceedings from SEPG 2000: Ways to Make Better Software. 20-23 March 1999. Seattle, Washington.
3. Cairó, O. Proyecto KAMET II. Instituto Tecnológico Autónomo de México. 2004.
4. Carreira, M. & Román, I. "Estimación del Coste de la Calidad del Software a través de la Simulación del Proceso de Desarrollo". Revista Colombiana de Computación, 2(1), 2002. pp. 75-87.
5. Dove, R. Value Propositioning - Book One - Perception and Misperception in Decision Making. Icen Books. 2004.
6. Dyba, T. "Factors of Software Process Improvement Success in Small and Large Organizations: An Empirical Study in the Scandinavian Context". Proceedings of the European Software Engineering Conference and ACM SIGSOFT Symposium on the Foundations of Software Engineering, 2003. pp. 148-157.
7. Bull España. "EPM como Garantía de Aplicación de una Metodología de Gestión de Proyectos en una Corporación". Soluciones Empresariales. Microsoft 2002.
8. Hadden, R. "Effective Planning and Tracking for Small Projects". Datatel, Inc. SEPG Conference. 2002.
9. Hersleb, J., Carleton, A., Rozum, J., Siegel, J. & Zubrow, D. "Benefits of CMM-Based Software Process Improvement: Initial Results". August 1994. Software Engineering Institute, CMU/SEI-94-TR-013.
10. Hersleb, J. & Goldenson, D. "After the Appraisal: A Systematic Survey of Process Improvement, Its Benefits, and Factors that Influence the Success". 1995. Software Engineering Institute, CMU/SEI-95-TR-009.
11. Humphrey, W. Introduction to the Team Software Process. SEI Series in Software Engineering. Addison Wesley, 2000.
12. Hussain, S. "PMLP - Project Management Using Temporal Logic Programming". IEEE Software. 2000.
13. IBM. "Gestión de Proyectos TIC en la Seguridad Social. Asociación Internacional de la Seguridad Social". Asociación Internacional de la Seguridad Social. Ginebra, 2004.
14. ISO/IEC 12207:2002/FDAM 2. Information Technology – Software Life Cycle Processes. 2004, International Organization for Standardization: Geneva.
15. ISO/IEC TR 15504:1998(E). Information Technology – Software Process Assessments. Parts 1-9. 1998, International Organization for Standardization: Geneva.
16. ISO/IEC 155504-2:2003/Cor.1:2004(E). Information Technology – Process Assessment – Part 2: Performing an Assessment. 2004, International Organization for Standardization: Geneva.
17. Jones, C. "Why Flawed Software Projects Are Not Cancelled in Time". Cutter IT Journal. 16 (12) pp. 12-17. 2003.
18. Jones, G. Software Engineering. John Wiley & Sons, Inc. New York, NY. 1990.
19. Kilpi, T. "Product Management Challenge to Software Change Process: Preliminary Results from Three SMEs Experiment". Software Process: Improvement and Practice. 3 (3) pp. 194-207. 1997.

20. Kuvaja, P. & Messnarz, R., “*BootStrap - A Modern Software Process Assessment and Improvement Methodology*”. Proceedings of the Fifth European Conference on Software Quality. 1996.
21. McFeeley, B. IDEALSM: A User’s Guide for Software Process Improvement. Handbook CMU/SEI-96-HB-001. Carnegie Mellon, University, Software Engineering Institute, February 1996.
22. McHale, J. & Wall, D. “Mapping TSP to CMMI”. Technical Report CMU/SEI-2004-TR-014. Carnegie Mellon University, Software Engineering Institute. April 2005.
23. Mondragón, O. “*Addressing Infrastructure Issues in Very Small Settings*”. Proceedings of the First International Research Workshop for Process Improvement in Small Settings. 2005. CMU/SEI-2006-SR-001. pp. 5-10.
24. MP, 2003. Microsoft Project 2003. Technical description available in: [www.microsoft.com/spain/office/products/project/default.mspx](http://www.microsoft.com/spain/office/products/project/default.mspx), 2006.
25. Oktaba, H. “MoProSoft: A Software Process Model for Small Enterprises”. Proceedings of the First International Research Workshop for Process Improvement in Small Settings. 2005. CMU/SEI-2006-SR-001. pp. 93-100.
26. OP, 2006. Open Plan Professional. Technical description available in: [www.welcom.com](http://www.welcom.com), 2006.
27. PE, 2006. Primavera Enterprise 3.0. Technical description available in [www.primavera.com/customer/products](http://www.primavera.com/customer/products), 2006.
28. Peirano, F. & Suárez, D. “*Las TICS mejoran el desempeño de las PYMES ¿Somos capaces de explicar cómo lo hacen?*”. Simposio sobre la Sociedad de la Información (SSI 2005). Rosario, Argentina. Septiembre, 2005.
29. Pressman, R. S. Software Engineering: A Practitioner's Approach. 5th Edition - European Adaptation. McGraw Hill, 2004.
30. Programa para el Desarrollo de la Industria Software (ProSoft). Avances al 1er semestre del 2004. Portal de la Industria del Software. Available in: [www.software.net.mx](http://www.software.net.mx)
31. Members of the Assessment Method Integrated Team. (2001) Standard CMMI® Appraisal Method for Process Improvement (SCAMPI), Version 1.1, (CMU/SEI-2001-HB-001). Pittsburgh, PA., Software Engineering Institute, Carnegie Mellon University. December, 2001. Technical description available in: [http://www.sei.cmu.edu/pub/documents/01\\_reports/pdf/01hb001.pdf](http://www.sei.cmu.edu/pub/documents/01_reports/pdf/01hb001.pdf)
32. Software Engineering Institute. CMMI for Systems Engineering, Software Engineering, Integrated Product and Process Development, and Supplier Sourcing (CMMI-SE/SW/IPPD/SS, V1.1). Continuous Representation. CMU/SEI-2002-TR-011, Software Engineering Institute, Carnegie Mellon University. 2002.
33. Standish Group International. “*Extreme Chaos*”. The Standish Group International, Inc. 2001.
34. Standish Group International. “*2004 Third Quarter Research Report*”. The Standish Group International, Inc. 2004.





# Towards Tool Support for Situation-Dependent Software Process Improvement

Gábor Bóka<sup>1,3</sup>, Dr. Katalin Balla<sup>1,3</sup>, Prof.dr. Rob J. Kusters<sup>2</sup>, Dr.ir. Jos J.M. Trienekens<sup>2</sup>

1. *Budapest University of Technology and Economics, Department of Control Engineering and Information Technology, Magyar tudósok körútja 2, 1117 Budapest, Hungary*  
{*bokusz, balla*}@iit.bme.hu  
<http://www.iit.bme.hu>
2. *Eindhoven University of Technology, Department of Technology Management, PO Box 513, 5600 MB Eindhoven, The Netherlands*  
{*r.j.kusters, J.J.M.Trienekens*}@tm.tue.nl  
<http://www.tue.nl>
3. *SQI Hungarian Software Quality Consulting Institute, József krt. 53, 1083. Budapest, Hungary*  
{*boka.gabor, balla.katalin*}@sqi.hu  
<http://www.sqi.hu>

## Abstract

The article focuses on a problem software companies often face: the need to choose a software quality model that ensures the connection of quality goals to the business goals of that company. As different models focus on different elements of software quality, a good solution seems to be to quit relying exclusively on one quality model and use more models in a synergic way instead: to do situation-dependent software process improvement. We explain a possibility to handle this problem by using a framework that helps positioning existing software quality approaches and standards according to the basic elements of software production they address, and gives guidance in the sequence of possible steps to do really efficient software process improvement in each concrete situation. We present QMIM, the theoretical framework of the suggested solution and the first results of an R&D project started to develop a methodology and a software tool usable by software companies operating in real business environment.

## Keywords

Software quality, Software process improvement, Situation-dependent software process improvement, CMMI

### **1 Introduction**

In the intense international competition software companies are more and more forced to think about proving their capability of delivering good products. One way of having such a proof is to produce an official certificate about usage of a certain standard or model. However, introducing an approach based on a standard or model, and institutionalizing it, so that the organization is able to pass an audit, requires a lot of investment from software companies, both in terms of money and effort – which a company would not like to waste. Therefore, really business-driven software companies will be willing to do only really efficient software process improvement.

In our opinion, efficient improvement programs are always based on real needs of companies, will always start from understanding the actual situation. Choosing the right approach, model or standard for the improvement program would be the next step. The difficult question is: which model to choose to best fit the company's needs in improving software quality? In which direction to move if the company wishes to reach higher software quality?

For answering these questions, we have to understand what software quality means in each particular situation - as there is no universally definable "good quality". Based on this understanding, the right quality profile has to be built – which means defining the most important objects and characteristics of software production in that particular situation.

In the beginning of the article (2.) we address the complex subject of software quality by describing the main objects we have to deal with and their basic attributes, as well as some of the most popular software quality models. We position these models against the objects of software production and we point out the need to quit exclusively relying on one model in favor of using more quality models in a synergic way. Companies usually face difficulties in choosing the right models, therefore some well structured guidance in would be useful to help connecting quality elements to existing standards and models.

In chapter 3. we develop further the ideas of important objects and attributes software quality is dealing with by describing QMIM, a theoretical framework that helps positioning the existing software quality approaches and standards, and gives guidance in the sequence of possible steps to do really efficient software process improvement. We present the elements of the theoretical framework, followed by the first results of an R&D project started to develop a methodology and a software tool based on QMIM, usable by software companies under market condition.

In the end of the article (4.) we show some characteristics of the first prototype of the software. We conclude by showing further directions of the work done up till now.

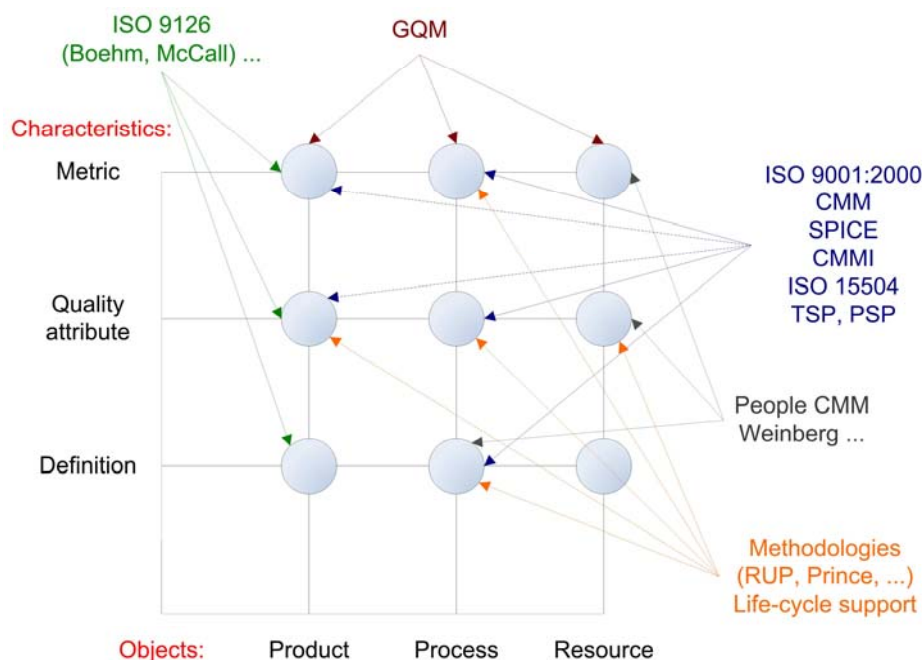
### **2 The complex subject of software quality**

Quality of software is a very complex subject, and, as such, it is extremely hard to define. If we wish to deal with software quality in its complexity, we have to think about the software products, the processes that produce the products and the resources that execute the processes (these elements appear also eg. in Fenton [1]). We have to define these objects, to choose the right quality attributes for them and verify the actual value of these attributes by the means of objective metrics.

The approaches, standards and models used in software industry are extremely various in their approach used. The first approaches used (Boehm [1] and McCall [2] models) were connected to software product quality. The fact that definition of objective quality attributes and metrics (emphasized by these models) turned out to be rather complicated, surely influenced the software community to look for approaches "easier" to apply and asses in business life. The idea that if the processes are "good", they will produce "good products", therefore we need to concentrate on organizing and managing the processes of software production, became very popular in the nineties. The processes of software production are being very emphasized by approaches like ISO 9001 (its actual version being ISO 9001:2000 [3]), CMM [4], SPICE [5], ISO 12207 [6], by PM and software development methodologies.

A big advantage of the process oriented approaches is that they are auditable, and the certificate of an audit is well usable in business. The problem with process oriented approaches is that, if used just "verbatim", they might leave software product quality out of consideration. Therefore, software professionals needed to combine process oriented approaches with approaches concentrating on software product characteristics and metrics. For this purpose ISO 9126 family [7] was developed, using the experience of earlier Boehm and McCall models. Answering the need to make software product more emphasized in process oriented approaches, new approaches and new versions of elder approaches appeared (like CMMI [8] or ISO 90003 [9]), that, besides the process-oriented characteristics, introduce elements of product quality as well. Nowadays approaches concentrating on different aspects of measurement (like GQM [10] or function point counting methods) are also known, and models emphasizing the importance of the human factor (e.g. P-CMM [11], PSP [12], TSP[13]) become more and more popular, also.

Summarizing the statements made before, we can represent the basic objects of software production and their important attributes in a framework. If we place the most popular quality approaches into this framework, we obtain a picture like the one shown in Figure 1. It is to the discretion of the reader to understand that if we wish to deal with software quality in its complexity, we have to complete the entire framework shown in Figure 1.



**Figure 1. Important objects of software quality, their characteristics and some popular quality-approaches**

From this picture we can draw 2 basic conclusions:

- The number of the approaches dealing with software quality is large, many of them overlapping in the coverage of some elements.
- There is no approach that would deal with all important objects of software quality.

The first conclusion tells us that it is extremely important to understand the elements covered by a certain approach.

The second conclusion leads to the idea that as no approach, model or standard covers all the aspects of software quality (although new versions of earlier models are definitely more broad in their scope, in the number of objects they are dealing with), companies will have to choose the right combination of approaches based on their business needs. This means to quit exclusively relying on one certain quality model - in favor of choosing among several approaches, using more approaches in a synergic way, according to the specific business needs of a certain organization. Such a combined

use of more quality models or approaches would be based, on the one hand, on a solid theoretical knowledge and understanding of the popular quality models and their interconnections, and, on the other hand, on a huge experience in the way the companies in case execute their development processes and do business.

For the moment, there is no guideline, aid or description in literature that would tell a company just starting to deal with software quality improvement what objects to choose, to what approach to connect them, what would be a “right combination of approaches” in their case and in which way to proceed if they want to do a real efficient software quality improvement.

We faced these problems in our day to day work, and we developed the framework presented in chapter 3 to help solving this problem.

### **3 QMIM: Quality through Managed Improvement and Measurement**

QMIM – Quality through Managed improvement and measurement (described in detail in [14]) is a framework that helps software developing companies to find their way among the many models, standards, methods connected to software quality, and to efficiently use the basic concepts of software quality. The framework facilitates understanding, consciously selecting and applying models, standards and methods connected to software quality, and combining them in a way that best fits own needs. QMIM is not a new approach or method towards software quality, but it acts like a “navigator” to clarify important objects of software quality and their interconnections, as well as to position any existing quality standard, approach or methodology against these objects.

#### **3.1 Some elements of the theoretical framework QMIM**

Basic elements of QMIM would be the objects and attributes shown in: product, process, resource on the “objects” dimension, and definition, quality attribute and metric on the “characteristics” dimension. If we agree on the fact that processes in software production can be divided into project management processes and technical processes, we will have the elements product, technical process (TP), project management process (PM) and resource on the “objects” dimension. As resources are dealt with usually within project management processes, we can regroup the objects to have product, TP and PM on the “objects” dimension.

The most important requirement at this moment of “getting started” with QMIM is to understand the important objects and attributes of software production and to define them unambiguously in the specific environment of the user. QMIM description ([14]) gives the basic elements of the framework, together with their suggested definition. When using the model in a concrete environment, one will have to analyze the actual situation and, maybe, redefine some of the elements.

While executing the activities suggested by the QMIM framework, one will generate different types of data. The quality manager will describe the project management and the technical processes, will identify different quality attributes and metrics to measure them, will record data regarding evolution of projects, results of measurements etc. Therefore, we consider it appropriate to have a static data model / view of QMIM framework, which will guide the company in storing the different data. This is the QMIM data model.

QMIM guidelines have been developed to help making the QMIM framework operational. They support the elements of the framework, and make them applicable in an organization. Having QMIM reference framework as a starting point, the guidelines provide aid in populating the quality framework and using it in the way that fits the company's goals in the best way. The guidelines, in fact, help build a company's own methodology to deal with software quality in a unified, balanced way. Using the guidelines, the company can build its own process model(s), using the experience gathered by existing standards, models, methodologies.

### 3.2 Towards tool support for QMIM

QMIM, as a theoretical framework, has been worked out based on the real needs of a Hungarian software company, IQSOFT Ltd. (see [14]). The basic need it satisfied was to give guidance for the company in understanding the main objects of software quality and to connect different popular models to these objects. While using the theoretical framework (see [18]), it turned out that the theoretical framework could be made really useful for software companies if a tool would be available to aid its usage. In December 2004 we started to develop such a tool, within the tender "TST-GVOP-2004-K+F-3.3.1" ("Development in Hungary of world-class services connected to software quality improvement and auditing"), sponsored by EU and the Hungarian Ministry of Trade. SQI coordinates the project, having TUE and TUB as partners. Authors of this article participate in requirements definition and development of the tool.

"QMIM tool" has as basic goal to help software companies understanding quality-related knowledge provided by different standard, methodologies and approaches, and choosing the software quality models and / or standards best fitting their needs. The tool will provide:

- structured and searchable information about popular software quality models, standards, best practices, reports,
- the possibility to store company-specific quality-related information in a structured way (also meaning customization of existing approaches),
- the possibility to store project-specific information in a structured way,
- the possibility to link company-specific and project-specific data to important to elements of software quality and the elements of the popular quality models.

In the requirements specification phase we looked at QMIM elements and to the needs of software development companies. After doing feasibility studies, we defined the basic structure of QMIM tool. This basic structure is presented in Figure 2.

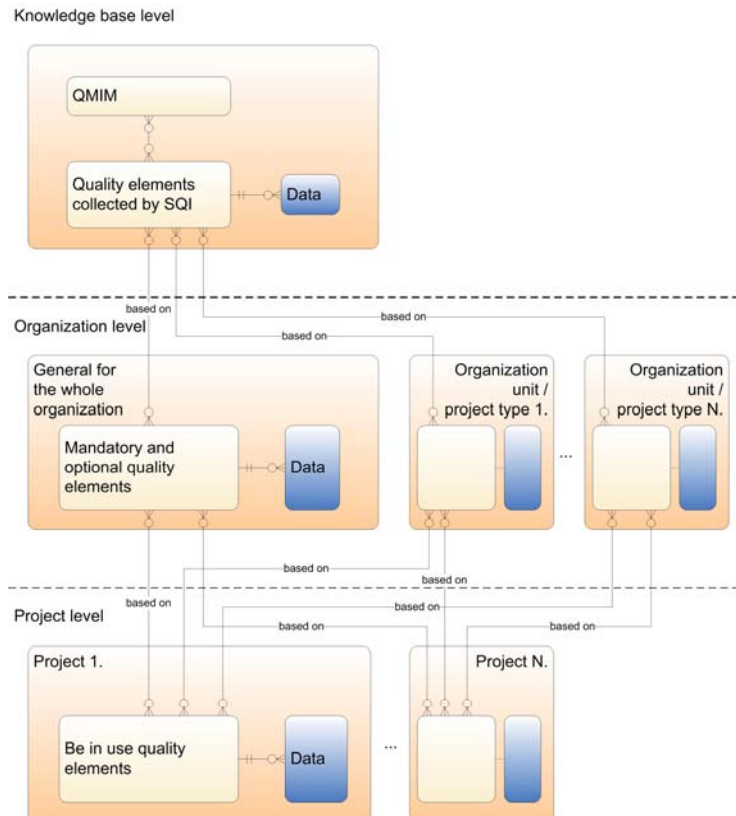


Figure 2: The three layer data model of the QMIM tool

We have defined three layers in the model: knowledge base level, organization level, project level. The theory level is maintained by the developer, the next two levels are maintained by the user organization. The knowledge base level contains the information regarding software quality models, standards, approaches, best practices etc. that a company will use as a primary source when deciding what quality elements will be addressed and what approach(es) will be implemented. At this level a description of the QMIM framework is given, and all approaches are positioned against QMIM. The quality elements on knowledge base level are stored by the developer of the tool in the development phase. This part of the software will be continuously updated with new models and standards, by the developer of the software.

The middle level is the organization level where the user organization can store quality elements valid for the whole organization. It will contain standard processes, tailoring guidelines, project types identified in the organization, life cycle models used by the company etc. – in fact all elements common within the organization.

The lowest level is the project level where the user organization stores quality elements of their own projects that are generated using the organization's standard processes and tailoring guidelines. Populating of each level is helped by the next higher level.

On all levels "data" are stored, resulting from the usage of a certain quality element. Eg., concrete data found in case studies regarding measurement (eg. ISO 9126 quality profiles, concrete results of a GQM approach...) will be stored at knowledge base level (as "models" possible to follow in the organization), while concrete data generated in the organization and in the projects will be stored at organizational and project level, respectively.

Due to the fact that we continuously update the knowledge base, the software has to be able to automatically notify the user if a change in a quality element - on which a user quality element is based - has happened. This enables the organization to maintain its quality system easier.

On every level of the previously presented data model there are quality elements. During the specification, we refined these elements. First, we collected all possible elements that could be connected to software quality, by looking to some popular models and standards. Next, we tried to group these elements and find some definitions for them. These trials have been documented and analyzed. For instance, in [17], GQM concepts are used to complete the list of quality elements. After more regroupings, 16 quality elements resulted (Case Studies, Templates, Tools, Standards, Quality Issues, etc.).

Refinement has continued by grouping similar elements and / or defining new elements for one group. Finally, 11 quality elements resulted. These have been defined in the following way:

- Guideline: a set of general and repeatedly applicable rules or attributes which give the optimal solution under given conditions.
- Case study: a practical application of a method through a real example.
- Best practice: defines the practices through which the best performance can be reached. It is usually elaborated from successful case studies.
- Tutorial: systematically constructed study aid which provides a step by step method for learning of something. In order to increase the learnability it usually applies special software tools.
- Lifecycle: a conceptual frame which contains such processes, activities and tasks which get roles in product development, use and maintenance, and which embrace the whole life of the system from the requirements specification to the end of its use.
- Template: a formatted document for a given task, which contains some variable parts for reusability.
- Definition: delimits or describes the meaning of a concept or term by stating the essential properties of the entities or objects denoted by that concept or term.
- Metric: describes what, why, when, how and what kind of metric scale has to be measured with
- Quality attribute: a set of properties through which the quality can be described and measured.

- Software tool: any instrument, technique, or device that provides a mechanical or mental advantage in accomplishing a certain task.
- Certificate description: describes what it certifies, what the process of the certification is, for how long the certificate will be valid, what organization did publish it, and which the well-known organizations who issue this certificate are.

The first six quality elements listed above have been grouped into the quality element called “document”, which will have “guideline”, “case study”, “best practice”, “tutorial”, “life cycle”, and “template” as subcategories.

The relationships between the finally resulting six quality elements are represented in Figure 3. Using E-R-diagram concepts and notations, some collectors have been introduced (“quality elements”, “definitions” and “metrics and attributes”) to facilitate representing all the existing connections between the elements.

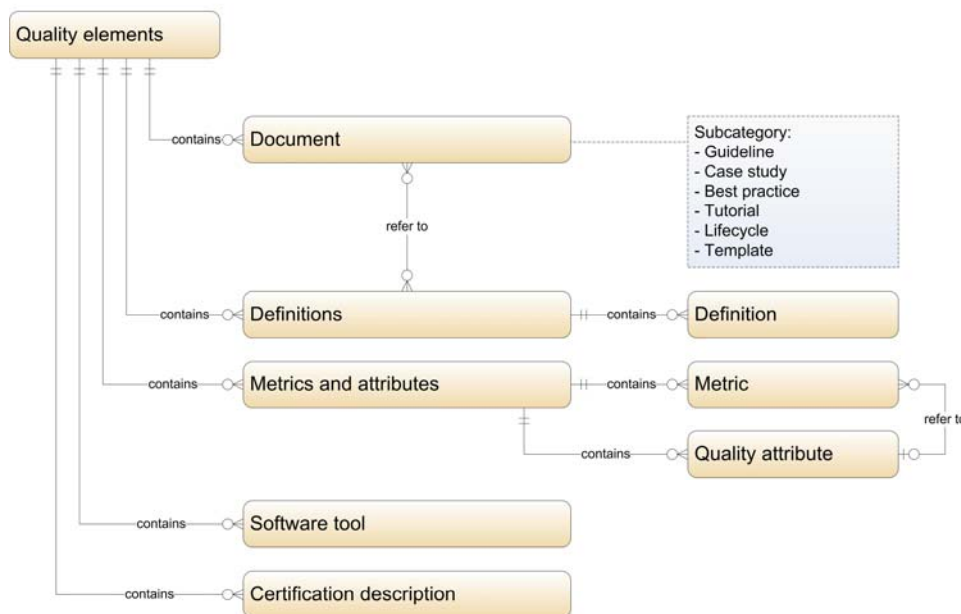


Figure 3: The database content of the QMIM tool: based on an elaboration of quality elements

### 3.3 The first prototype

Actually, the requirements specification and design has ended, and the development of the QMIM tool is in testing phase of its first prototype. The prototype data structure was developed upon the basis of the previously introduced data model. First, the GQM (Goal Question Metric), the EF (Experience Factory) and the PSP (Personal Software Process) approaches were prepared and uploaded into the software database. We intended to develop an easily learnable and useable user interface for the software. The Figure 4., 5. and 6. show some screenshots of the user interface.

Screen in Figure 4 shows the attributes of a quality element, selected from the list of quality elements appearing on the left side of the screen. The selected “quality element” belongs to GQM approach, and it is a “guideline” (referring to the usage of GQM). In the right –hand side of the screen the properties of the selected element are shown, such as: name, date of issue, language, authors etc. So, this screen shows a knowledge base level view of the quality attribute “guideline”. Screen in Figure 5. shows how the quality element (“guideline”) selected in the left-hand side of the screen is connected to the important elements of software quality captured by QMIM. It is a knowledge base level view, also. Screen in Figure 6 displays the contents of the quality element (“guideline”) selected. Similar screens are presenting elements stored at organizational and at project level.



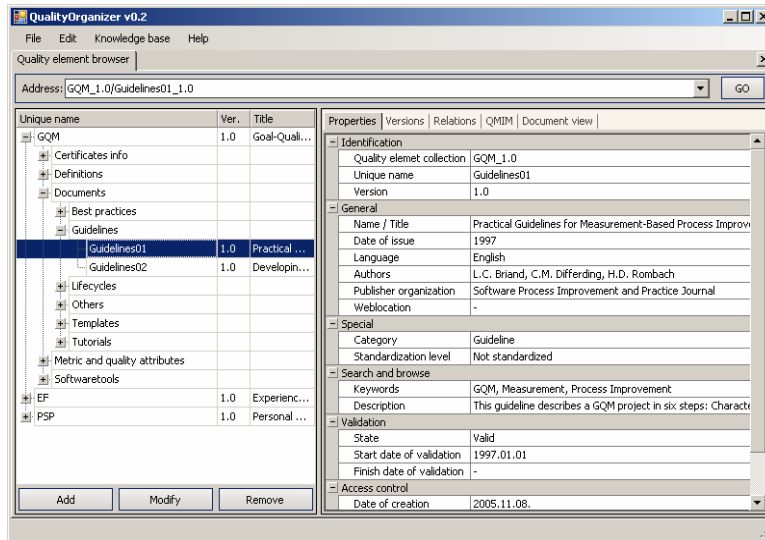


Figure 4: Properties of a quality element

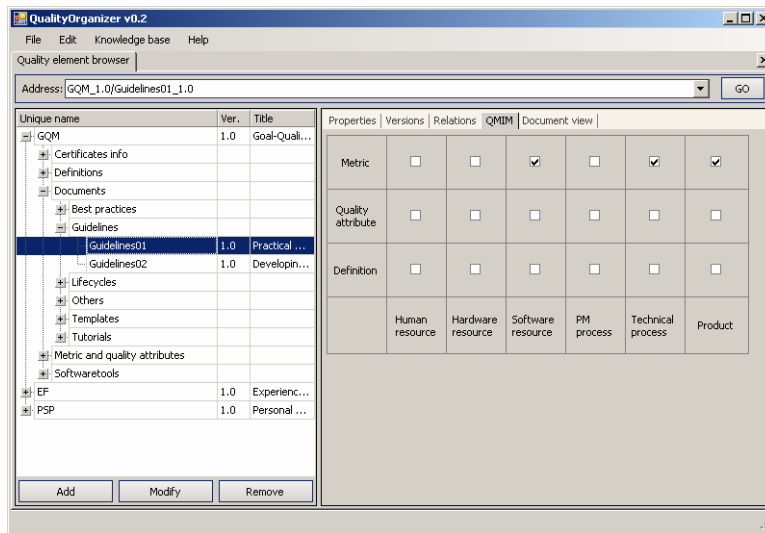


Figure 5: A quality element in the QMIM.

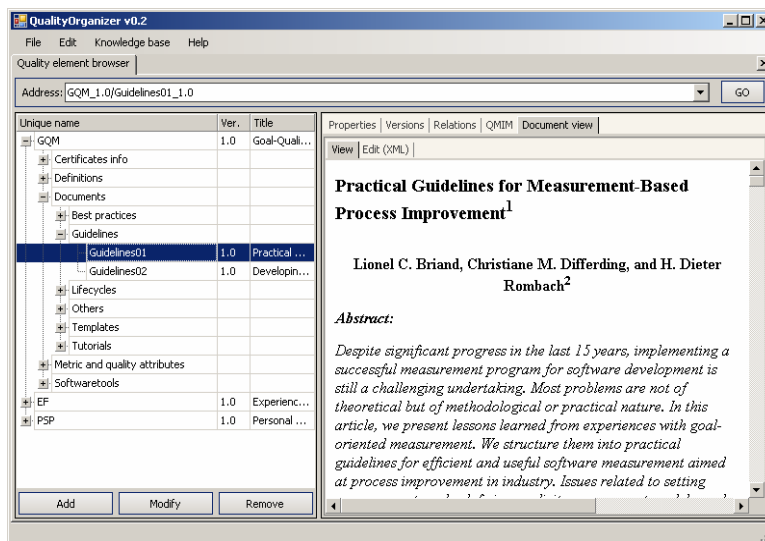


Figure 6: Contents view of a quality element

## **4 Using QMIM in real business environment**

In the article we presented QMIM from two aspects: as a theoretical framework (see 3.1) and as a tool that helps using the theoretical framework (see 3.2 and 3.3).

Using QMIM as a theoretical framework has been done in the environment where the framework was developed: at IQSOFT Ltd., Hungary. Regarding this, an 11 years long case study has been described in [14] and [18]. Between 1993-2004 a medium-sized Hungarian software company understood that software quality can be approached based on products, processes and resources. It made conscious choices over time in using a certain approach. First, project management, afterwards technical processes have been defined, associated quality attributes and some metrics have been collected. Trials have been made to define software product types and associated quality profiles. The popular quality standards and approaches of that time have been used: project management methodology, ISO 9001, CMM, function point counting (COSMIC method). We consider an important achievement that QMIM, as a theoretical framework helped the company to connect its business goals to concrete and measurable quality goals.

The company managed to structure its quality-related data in a manner that followed QMIM principles. This structuring was done based on QMIM theoretical framework, having no special QMIM - tool support (just IT support that can be considered usual in a software development company).

The tool described in 3.2 and 3.3 is under development, only a first prototype has been completed. It has not yet been tested in real environment. We describe our plan about testing it in chapter 4.

## **5 Conclusion and further work to be done**

In the article we pointed out the problem of choosing among quality approaches and integrating more models in a synergic way, faced by many software development companies. As a possible solution to this problem, we presented QMIM and described its basic elements and characteristics. We pointed out some first results of an R&D project, developing a software tool to aid QMIM usage.

As next steps of our R&D project we will populate the knowledge base with the most popular standards and models: ISO 9001:2000, ISO 90003, ISO 9126, ISO 12207, CMMI. We will include information about the quality elements each model addresses. We will prepare the "organization level" and "project level" of the database for storing elements of concrete quality a management systems in a structured way, also helping the user to understand how his / her concrete elements are connected to important elements of software quality and to popular software quality approaches.

The next milestone is publishing the first version of the knowledge base which will be tested by software quality experts from 3 software companies, and refined according to their feedback. In parallel, we work on the QMIM guidelines. These would provide guidance to the used of QMIM tool in steps to follow when choosing quality objects and approaches or standards.

## **6 Acknowledgements**

We would like to thank Peter Schroen (TUE), Daniel Zádor Kelemen (SQI, TUB) and Dirk van Driel (TUE) for their work done in the TST-GVOP-2004-K+F- 3.3.1, which was an important contribution in defining the requirements and the first prototype of the QMIM tool. Their results were used while writing this paper.

### 7 Literature

- [1] Fenton, N.E.: Software metrics - a rigorous approach. Chapman&Hall, 1992.
- [2] McCall J. A.: The utility of software metrics in large scale software systems development, IEEE Second software life cycle management workshop, August 1978.
- [3] EN ISO 9001:2000: Quality Systems, Requirements. CEN, Bruxelles, 2000
- [4] The Capability Maturity Model - Guidelines for Improving the Software Process. Addison Wesley, 2000.
- [5] Rout, T. P.: SPICE A framework for software process assessment. In: Software Process - Improvement and Practice 1(1), pp. 57-66, August 1995.
- [6] ISO/IEC 12207:1995. Information technology. Software life cycle processes.
- [7] ISO 9126: Software engineering — Product quality: 1: Quality model (ISO/IEC 9126-1:2001). 2: External metrics (ISO/IEC TR 9126-2:2003).3: Internal metrics (ISO/IEC TR 9126-3:2003). 4: Quality in use metrics (ISO/IEC TR 9126-4:2004).
- [8] M.B. Chrissis, M. Konrad, S. Shrum: CMMI. Guidelines for Process Integration and product Improvement. Addison-Wesley, September 2004.
- [9] ISO/IEC 90003: Software engineering – Guidelines for the application of ISO 9001:2000 to computer software. First edition, 2004-02-15.
- [10] Basili, V.: SEL's Software Process Improvement Program, IEEE Software, Nov. 1995.
- [11] People Capability Maturity Model® (P-CMM®) V. 2. <http://www.sei.cmu.edu/cmm-p/version2/>
- [12] Humphrey, W: Introduction to the Personal Software ProcessSM; Addison-Wesley ,1997.
- [13] Humphrey, W: Introduction to the Team Software ProcessSM. Addison-Wesley, 1999.
- [14] Balla K: The complex quality world. Developing quality management systems for software companies. Ph.D. thesis. Beta Books, Technische Universiteit Eindhoven, 2001. ISBN: 90-386-1003-3
- [15] Humphrey, W.: A Discipline for Software Engineering. Addison-Wesley, September 2000.
- [16] Standard CMMI® Appraisal Method for Process Improvement (SCAMPISM),Version 1.1: Method Definition Document.<http://www.sei.cmu.edu/publications/documents/01.reports/01hb001.html>
- [17] Schroen, Peter: Modeling and Managing Software Quality, an elaboration on QMIM. TU/e Internal Research Report. TU Eindhoven, 2005.
- [18] Balla K.: Software Process Improvement and Organizational Change. In: 50 jaar informatiesystemen 1978-2028. Liber Amicorum voor Theo Bemelmans. pp. 181-198. March 2004. Edited by Capaciteitsgroep Information Systems, Faculteit Technologie Management, Technische Universiteit Eindhoven. ISBN 90-386-1928-6, NUR 982

## 8 Author CVs

### Gábor Bóka

Gábor Bóka works for SQI- Hungarian Software Quality Consulting Institute Ltd. as the leading software developer of the tender "TST-GVOP-2004-K+F-3.3.1". He also works for the Department of Control Engineering and Information Technology at the Budapest University of Technology and Economics, where he takes part in research and education. He obtained his master degree as a system developer from the Budapest University of Technology and Economics in 2005. The title of his thesis was: "Software support for COBIT based IT security audit". The software was developed in C# programming language. During the years spent at the university he got deeper experience in server side database programming, in Java and C# programming languages. He made scientific investigation at IT security and software quality management domain. Prior, in 2002 he obtained his bachelor degrees as software developer and as embedded programmer from two faculties of the Széchenyi István University, Győr. In his thesis he dealt with questions of the client-server application development especially it's implementation in Borland Delphi environment.

### Dr. Katalin Balla

Dr. Katalin Balla is a lecturer at Technical University of Budapest, where she teaches software quality management and software testing since 2001. She is founding member and managing director of SQI- Hungarian Software Quality Consulting Institute Ltd. ([www.sqi.hu](http://www.sqi.hu)), where she works as a consultant in software process improvement. Prior, Katalin worked as a quality director at a Hungarian software company for 11 years. She conducted all the process improvement projects at the company (introducing a company-wide project management methodology, followed by development of ISO 9001:1994 and ISO 9001:2000- conform quality management systems). Katalin is a qualified ISO 9001:2000, Bootstrap and SPICE auditor, her training to become a SCAMPI Lead Assessor is in progress. She worked for 10 years as a programmer, a software system engineer and as a scientific columnist at newspapers and radio. K. Balla graduated as an informatician in 1984 from "Babes Bolyai" University of Science Cluj, Romania. She obtained a Ph.D. in 2001 at the Technical University of Eindhoven, the Netherlands, in the field of software quality.

### Prof. dr. Rob J. Kusters

Prof.dr. Rob J. Kusters (1957) obtained his master degree in econometrics at the Catholic University of Brabant in 1982 and his Phd. in operations management at Eindhoven University of Technology in 1988. He is professor of 'ICT and Business Processes' at the Dutch Open University in Heerlen where he is responsible for the master program 'Business process and ICT'. He is also an associate professor of 'IT Enabled Business Process Redesign' at Eindhoven University of Technology where he is responsible of a section of the program in management engineering and is an associate member of the research school BETA which focuses at operations management issues. He published over 70 papers in international journals and conference proceedings and co-authored five books. Research interests include enterprise modelling, software quality and software management.

### **Dr.ir. Jos J.M. Trienekens**

Dr.ir. Jos J.M. Trienekens (1952) is an Associate Professor at TU Eindhoven (University of Technology – Eindhoven) in the area of ICT systems development. He is responsible for a research program on ICT driven business performance and is an associate member of the research school BETA at TUE that focuses at operations management issues. Jos Trienekens published over the last ten years various books, papers in international journals and conference proceedings. He joined several international conferences as PC member and member of the organisation committee and is an experienced project partner in several European projects (ESPRIT/ESSI).

# European E-Security Manager Skills Card and Learning System

Alen Salamun<sup>1</sup>, Richard MESSNARZ<sup>2</sup>, Damjan Ekert<sup>2</sup>,

<sup>2</sup>*Real Security, Maribor, Slovenia*

*Email: [alen.salamun@real-sec.com](mailto:alen.salamun@real-sec.com)*

<sup>2</sup>*ISCN GesmbH, Schieszstattgasse 4, A-8010 Graz, Austria*

*Email: [rmess@iscn.com](mailto:rmess@iscn.com)*

**Abstract:** This paper presents a current project E-Security Manager which developed a skill card for an European e-security manager and is currently designing a course program which will be made available online in 2007. The project is being funded under the Leonardo da Vinci Program (SI/05/B/F/PP -176.008). This topic is important for SPI because a missing e-Security strategy can lead to losses of time, human resources, and data and thus in future we must consider to involve such processes in the SPI scope. A good e-security strategy can have a positive impact on productivity.

## 1. The Need for a Job Role and Skills Card in E-Security Management

In this age of continuous innovations and technological changes, the European expansion, and the movement of Europe into an e-Europe in the next few years, the topics of e-Security became an important issue on the European agenda. The strategy for an e-Europe will only work out if at the same time the e-Security issues in the national laws and European businesses are adapted to this new form of business and networked society.

The main target groups are SME IT departments, system administrators and IT students at technical colleges as well as managers and directors, regional managers, educational decision makers, and project managers of companies in the IT services sector who are forced by the current global and European developments towards an e-society to change and prepare their organisations from a traditional environment to an e-commerce environment with specific e-Security requirements. (some examples: protection against data theft, viruses, service attackers and hackers, etc.) . These groups generally comprise of users of IT, providers of IT products and services and learners of IT. In addition, each national government or appointed agency has some kind of e-security initiative that provides some governance function, i.e. to create or facilitate national or sector standards, to implement or influence international standards, Such standards cover overall strategy, processes and techniques for implementation.

The success of organisations (and sometimes even the survival) will depend on a proper set up of IT strategies and corresponding e-Security measures [2],[6],[8].

The need for an **e-Security Manager** [6],[11] is closely related to arising problem of poor and often underrated security education of IT staff employed in companies. Even managers often underestimate the importance of e-security and safety of data. Experiences have shown that smaller companies do not outsource their IT security related tasks due to high costs; they rather appoint the system administrator or someone of the existing IT team. This can lead to

serious security problems and also to a potential hazard to the whole company ranging from lose of data, system downtimes and even to industrial espionage. Another worse common practice is that even the basic security policies are ignored or not implemented.

Europe is developing towards an e-Europe with growing opportunities but at the same time with growing threats. Since 1999 the EU releases directives to attack these issues such as

- Directive 2000/31/EC of the European Parliament and of the Council of 8 June 2000 on certain legal aspects of information society services, in particular electronic commerce, in the Internal Market ("Directive on electronic commerce") [2]
- Directive 1999/93/EC of the European Parliament and of the Council of 13 December 1999 on a Community framework for electronic signatures [3]
- Official Journal L 013 , 19/01/2000 p. 0012 – 0020 [4]
- European Security Strategy endorsed by the European Council, December 2003, it is general security, but IT security is a part of that. [5]
- Etc.

While releasing such directives it is left open for each national state to implement the directive, resulting in different security strategy implementations in Slovenia, Hungary, Ireland, Austria, and Bulgaria (as well as in other countries). Austria released more detailed laws while in Germany and the UK a whole IT security guidebook was developed.

When looking at the political strategies and the different national educational programs in this area, one of the major problems is that the scope of an e-security manager education differs from country to country. Thus the project develops a first version of a cross-national set of skills and learning objectives [1], [6] which all European training institutions should cover to address a similar scope of skills.

A **skills definition concept** (skill card) [6], [11] is an approach where by using standards (NVQ - National Vocational Qualification Standards) a common sense is created among different countries and institutions (as a collection of good practices and required skills) and illustrated in a standard skills description structure.

This will allow a benchmarking of required skills against a European proposed skills set and the establishment of learning portfolios to upgrade the skills of nowadays innovation managers to a higher efficiency.

## 2. A skills card and skills portal for the e-security manager job role

The job role of an e-security manager (see Figures 2, 3) therefore is a specific position in an organisation who understands all these factors: Management and Policy (Management Awareness and Control, Security Policy Establishment, Education, Personal Security, EU and national standards and laws, physical security), System Administration (Antivirus and Content Filters, Updates, Authentication and Authorisation, Availability, Backup, Application Related Security) , Network Security (Firewalls and IDS/IPS Systems, Cryptography).

For structuring a skills set the EU leonardo da Vinci project E-Security Manager followed the EU standards for skills cards [6], [11] (see Figure 1).

A <b>Domain</b> , contains <b>Job Roles</b> , which contain <b>Units</b> , which contain <b>Elements</b> , which contain
-----------------------------------------------------------------------------------------------------------------------------------

**Performance Criteria, which must be proven by Evidences.**

Figure 1: Basic elements of the skills definition model

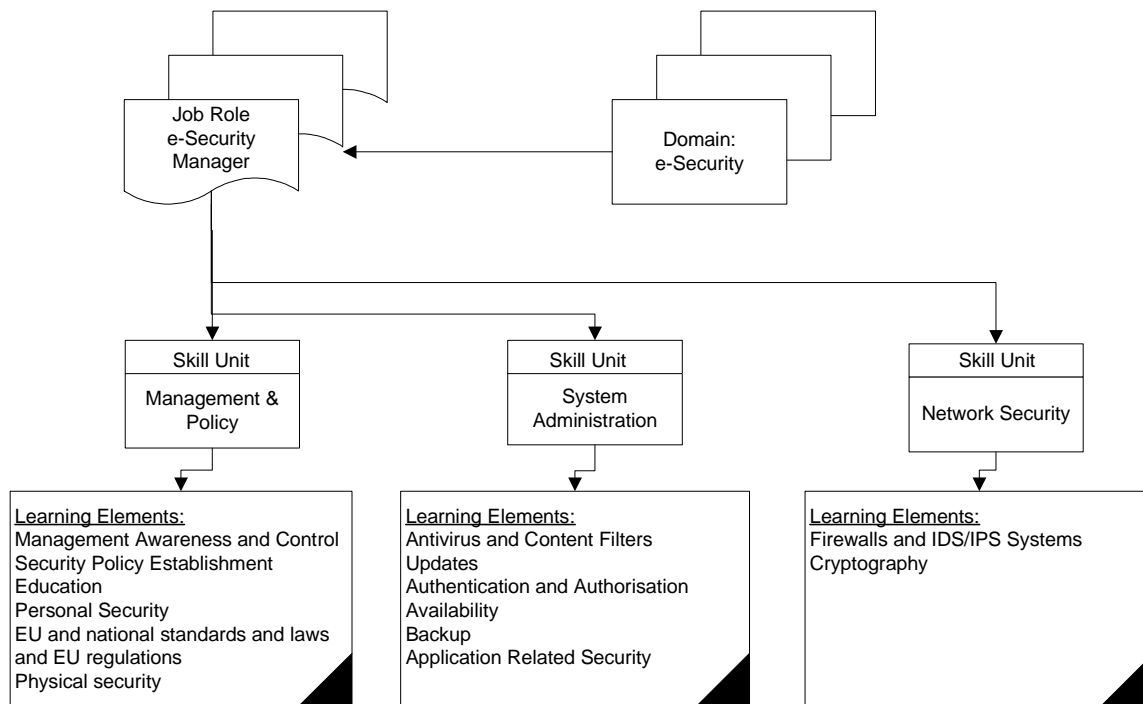


Figure 2: Skills Card of an Innovation Manager Job Role Part 1

**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 [9] 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.

In E-Security Manager we have identified 3 skills units and 14 learning elements for an e-security manager. These skills are available for online browsing and for online skills self-assessment.

For each learning element e-security manager develops a training module. The training module covers all performance criteria listed for the learning element in the skills card. A skills portal has been configured with the skills card and supports the steps of browsing required skills (see Figure 3), self assessment, formal assessment, evidence collection, generation of skills profiles (see Figure 2) for each learning element, and learning recommendations.



### 3. The Learning System Used to Offer E-Security Manager Education in the Future

A European skills assessment and learning portal system will support the skills card browsing and online courses from beginning of 2007. The portal is already running [6], [8], [9], [10] for a set of previously developed European professions (Certified EU innovation manager, certified EU project manager) and from October 2006 onwards will be handled by a European certification association (EU Certificates), also being a result of a Leonardo networking project in the educational strategy area.

**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 for Self Assessment 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. In case of the e-security manager project the registration will automatically assign a predefined service unit.

**Step 3 – Receive an Account for Self –Assessment and Evidence Collection :** With the registration you automatically receive an account to login to the 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.

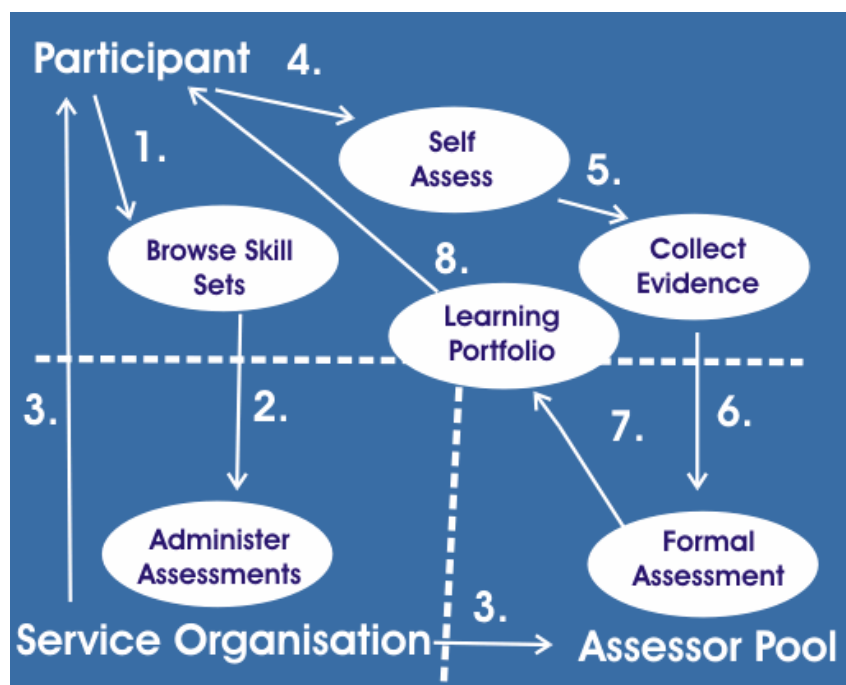


Figure 3: The basic steps in the learning system

**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.

Testing of Skills (Addition to Step 4) – The system provides a multiple-choice test for each performance criteria so that you can check your capabilities as realistically as possible.

**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, professional 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 fulfillment 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, as 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).

Since the end of 2005 this system has been interfaced with the most known e-learning portal system Moodle, supporting the learning cycle outlined in Figure 4.

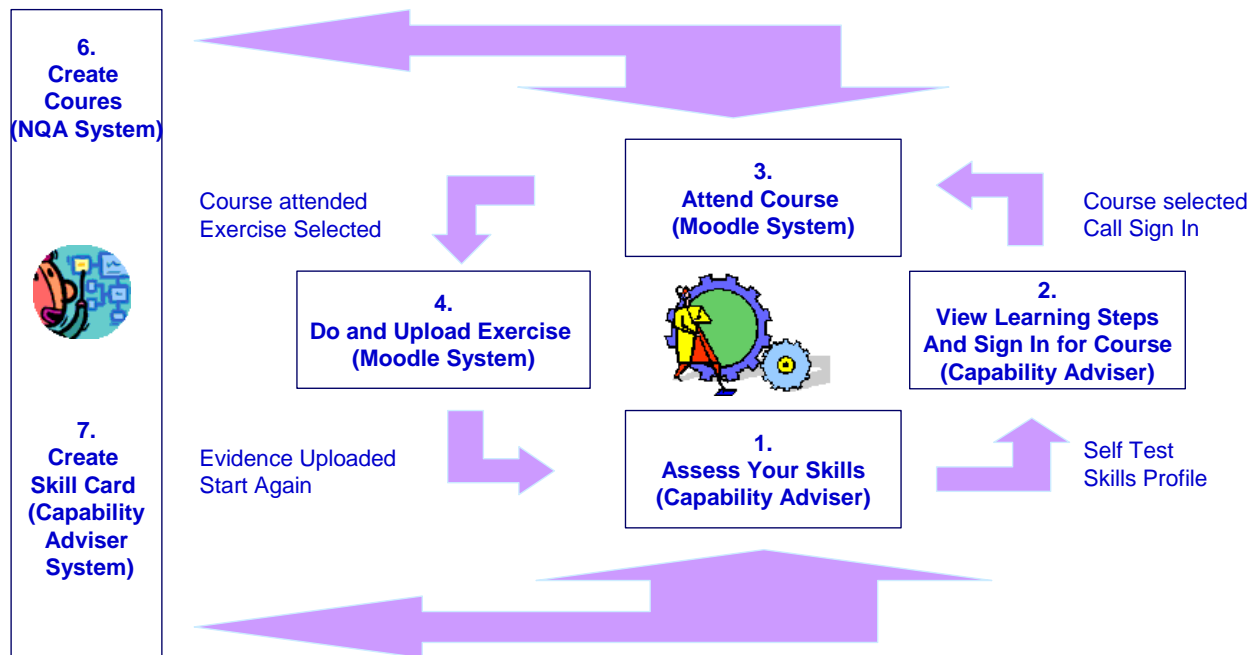


Figure 4: The basic steps in the learning system integrated with Moodle

1. Participants ( Learners ) log into the Capability Adviser, browse the skills tree, assess their skills against performance criteria, upload evidences to prove their skills, and print a skills profile.
2. Participants ( Learners ) select the “Learning Steps” option the Capability Adviser, access recommended learning references, and can call “Sign In” to log into courses on the Moodle web based training server system.
3. Users ( Learners ) on the Moodle System attend the courses , perform exercises, upload results of their homework, and receive feedback from the trainer.
4. Users ( Learners ) switch to the Capability Adviser window (if you did all in one session) or log into Capability adviser as participant and upload their homework results as evidence into the system to prove their competence.

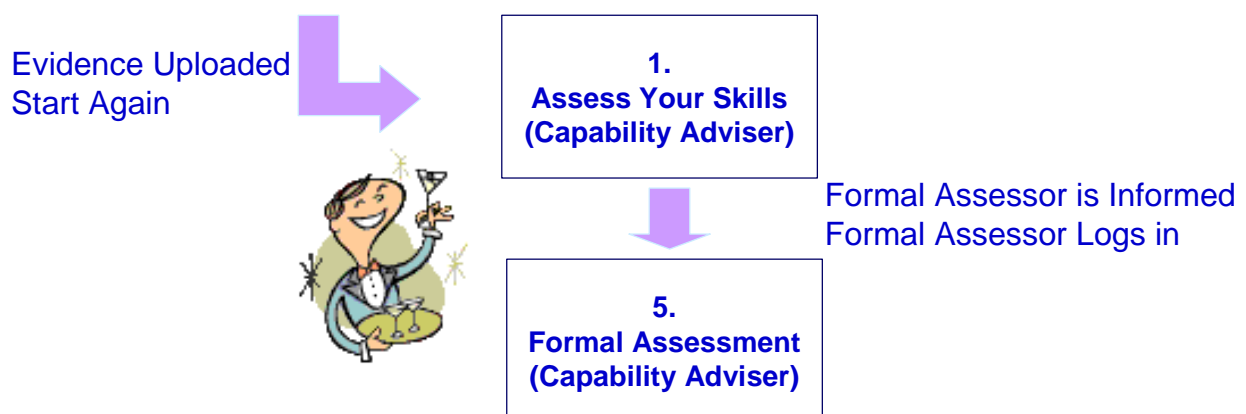


Figure 5: Online evidences of competence assessment by an assessor pool

5. Users ( Learners ) inform Assessors. Formal Assessors log into the Capability Adviser, assess the evidences, assess the performance criteria, and produce a formal skills profile of the user. The results of the formal assessor display separately.

## 4. Use case Scenarios

### 4.1 – Accreditation scenario examples

A skill card can be used to accredit training providers as well as to certify people.

For the job role (skill card) certified European innovation manager, for instance, already 476 attended the courses and 250 received certificates in Europe.

It is now planned to establish the same infrastructure and support for this new profession of an e-security manager.

Based on an agreed skill card different institutions from various countries use the same content and skill criteria.

Different participating universities use it to refine their study programmes to cover all aspects of the skill card and become comparable on a European level. And also training companies will use it to offer courses which can demonstrate the coverage of a minimum set of European requirements in the e-security area.

### 4.2 – Learning scenario examples – Learning from the work place

While universities create study programmes which focus on 4 to 5 year long learning schemes where students learn all skills related with general skill domains (a skill domain = computer science – telematics engineer, or computer science – software engineer, etc.), the industry (under constant time pressure and competition) has to focus and tailor much towards specific skills sets and job roles (software architect, tester, etc.) . And learning must become possible from the work place to upgrade skills for specific job roles.

The service portal of the e-security project will allow managers to access the server online, browse the required skills, do a self assessment, print an individual (confidential) skills profile and receive a generated learning plan / portfolio.

### 4.3 – Service Delivery Scenario – Virtual Competence Centers

Since July 2003 , for instance, a selected group of German companies (26 leading firms) started using cross company task forces with a virtual collaboration support to jointly develop key skills (partners of the project are involved). This model can apply for any group of companies and industrial segments, so we are setting it up for the e-security topics.

## 5. Acknowledgements

We acknowledge the support for this paper from the following project partners: Real Security (Slovenia), SZTAKI (Computer Automation Institute of the Hungarian Academy of Sciences, Hungary), ISCN (Austria, Ireland), University of Plovdiv (Bulgaria).

### References

- [1] M. Biro, R. Messnarz, A. Davison (2002) The Impact of National Cultures on the Effectiveness of Improvement methods - The Third Dimension, in Software Quality Professional, Volume Four, Issue Four, American Society for Quality, September 2002
- [2] Directive 2000/31/EC of the European Parliament and of the Council of 8 June 2000 on certain legal aspects of information society services, in particular electronic commerce, in the Internal Market ("Directive on electronic commerce")
- [3] Directive 1999/93/EC of the European Parliament and of the Council of 13 December 1999 on a Community framework for electronic signatures
- [4] Official Journal L 013 , 19/01/2000 p. 0012 – 0020
- [5] European Security Strategy endorsed by the European Council, December 2003, it is general security, but IT security is a part of that.
- [6] 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, Washington, 2002
- [7] Feuer E., Messnarz R., Wittenbrink H., Experiences With Managing Social Patterns in Defined Distributed Working Processes, in: Proceedings of the EuroSPI 2003 Conference, 10-12 December 2003, FTI Verlag, ISBN 3-901351-84-1
- [8] 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
- [9] 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
- [10] A Learning Organisation Approach for Process Improvement in the Service Sector , R. Messnarz. C. Stöckler, G. Velasco, G. O'Suilleabhain, A Learning Organisation Approach for Process Improvement in the Service Sector, in: Proceedings of the EuroSPI 1999 Conference, 25-27 October 1999, Pori, Finland
- [11] DTI - Department of Trade and Industry UK, British Standards for Occupational Qualification, National Vocational Qualification Standards and Levels
- [12] Gemünden H.G., T. Ritter, Inter-organisational Relationships and Networks, Journal of Business Research, 2001

# A Framework for Improving Effort Management in Software Projects

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

## **Abstract**

Rapid changes in Information Technology (IT) set challenges to software project effort estimation. Besides effort estimation, software projects involve various other effort-related functions which have an effect on the effort estimation. Some of these functions are required by the capability maturity models such as CMMI. However, both the capability maturity model requirements and research on software project effort is very fragmented. This paper proposes a framework for improving effort management in software projects. The framework assists both the total management and, on the other hand, concentration on selected areas of effort-related functions. The framework has been applied at TietoEnator Telecom & Media.

## **Keywords**

Effort management, framework, project management, SPI

### 1 Introduction

One of the most crucial tasks for software suppliers in the bidding and tender processes is to estimate the effort needed to produce software, and present the estimate to the customer. It is in both parties' interest that the effort is estimated initially as accurately as possible to ensure adherence to project's budget and schedules, and the success of resource allocation (Conte *et al.* 1986, Sommerville 2001). However, rapid changes in IT business domains, technologies, and customers set challenges to software project effort estimation.

Besides effort estimation, a software project involves various other effort-related functions throughout the project's lifecycle. During project planning, effort is initially estimated with the method in use. Effort can be estimated in several ways, and estimation techniques can be categorized in different ways. The estimate is revised during project execution with the collected effort data from the project. During project closure, the delivered project and its effort are analyzed, and a project final report is drawn. Some effort-related functions exist also before and after the project. These effort-related functions influence on effort estimation.

Software engineering literature (e.g. Royce 1998, Sommerville 2001, Pressman 2005) focuses on project management and effort, but the emphasis is on effort estimation and effort-related planning (e.g. scheduling) rather than on the total management of effort. This paper proposes a framework to improve effort management in software projects. The employment of the effort management framework assists in not only the total management of effort in a software project but also in concentrating on effort improvement topics which require most attention. To our best knowledge, a similar framework is not presented thus far in the context of effort management. Frameworks for improving effort management in software projects have been proposed (e.g. Fairley 1992, Vesterinen 1998, Tsoi 1999), but they differ from ours.

We have applied the framework to a particular set of software project activities, namely the non-construction activities (Haapio 2004), i.e. the activities which are not directly related to software construction or project management. These activities include various management and support activities such as orientation, planning activities, quality management, configuration management, customer support, and documentation, which are carried out by several members of the project. Our research aims to improve effort estimation and effort management by focusing also on activities other than the actual software construction and project management, and the whole effort lifecycle of the project. This approach makes the effort distribution proportions more stable, thus more predictable and controllable, from project to project (Haapio 2004). Neglecting the non-construction activities in project planning and effort estimation can result in significant effort proportion deviation between projects. Furthermore, regardless of the effort estimation method employed, even the best estimate cannot be accurate if the method neglects or undervalues the non-construction activities that are involved in the project. Effort estimation research has focused on software construction because the majority of the total effort is software construction effort (Boehm *et al.* 2000, MacDonell and Shepperd 2003). However, the rest of the effort is as important when a higher estimation accuracy is pursued. Considering the non-construction activities is useful also while comparing the efficiency of different projects.

This paper is structured as follows. Section 2 describes the research methodology. Section 3 outlines the framework and the software project engineering process where it is applied. Application of the framework is described in Section 4, followed with an analysis in Section 5, and a discussion in Section 6. A brief conclusion and suggestions for future research are given in Section 7.

### 2 Research Methodology

This study is based on constructive research (Iivari 1991), also referred to as design science (March and Smith 1995) which consists of two basic activities: building products (constructs, models, methods, and instantiations) and evaluating them. The evaluation of the built product is based on user value or utility, i.e. feasibility is demonstrated when the created artifact serves its purposes (Järvinen

2001, March and Smith 1995). Case studies can be used for evaluation (Kitchenham *et al.* 1995). In this paper a framework for improving effort management in software projects is proposed. The different elements of the effort management framework refer to constructs. A model expresses the relationships among these constructs. Here, the effort management framework refers to the model. The proposed framework is evaluated with case studies to demonstrate its feasibility in practical use.

### 3 Framework

#### 3.1 Framework Elements

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

- Framework perspectives, which include effort-related and activity-related perspectives. These perspectives are described in Section 3.3.
- Framework phases, which include a prior-project phase, phases of the project's lifecycle, and a phase after the project's existence.
- Software project lifecycle, which includes three phases: Pre-project, project, and post-project phase.
- Process, which is strongly related with the lifecycle of a software project. The process with the software project lifecycle is described in Section 3.2.
- Software project effort. Effort of a software project can be generally defined as the number of person days (or hours or months) consumed by the project. The software project effort is registered on the work time registry system's registration entities by the project group.
- Effort management activities. These activities include both effort-related (e.g. effort estimation) and activity-related (e.g. WBS establishment) activities. The activities are described in Section 3.2.
- Software project activities. A software project consists of activities, which can be arranged into a work breakdown structure.
- Registration entities. In our research, software project activities are called registration entities in the work time registry system (Haapio and Eerola 2006). The project group feeds the effort information on the entities.
- Activity sets and the activity set effort. A software project activity set is a specific set of project activities, which comprises a set of activities that are under examination. For example, the activity set of testing can include unit testing, integration testing, system testing, and customer acceptance testing. The activity set effort is the combined effort of the activities of the activity set in question (Haapio and Eerola 2006).
- Work breakdown structure (WBS). WBS is a particular hierarchy in which project activities are categorized. WBS is described in Section 3.4.1.

#### 3.2 Software Project Lifecycle and the Effort Management Process

The total project lifecycle can be divided into three main phases: pre-project, project, and post-project in which effort is first estimated, then collected, monitored and re-estimated, and finally analyzed.

During the pre-project planning, the project is planned and set up. Project planning includes the project activity planning and effort estimation sub-activities. Effort is initially estimated with the method in use with project information supplied by the customer, for example. At this point, the different activities concerning the project are also planned. These activities include the activities related to actual soft-



ware construction, project management, and other project-related activities. The planned activities are created, e.g. by the project manager, during the project setup into the work time registry system as registration entities for effort registrations during the project execution. These entities are organized into particular work breakdown structures.

During the actual project phase, the effort is collected, monitored, and re-estimated. Effort is collected during the project execution to monitor its realization. If effort realizes different than planned, the reasons are explored and necessary actions are taken. Furthermore, effort is collected for re-estimations with adjusting, project-specific data, and for effort analysis after the project has been completed. The effort collection includes effort registration on registration entities in the work time registry system. These entities are the activities of the software project. During project execution the project group registers effort on the activity entities that were created for the project in the work time registry system. The effort registration on old, familiar entities can begin immediately, and effort is usually registered on the correct entity. The new project activities and new sets of activities, however, require an adoption period before effort can be registered on the new entities. The adoption time varies but can at worst last the whole project execution which results in skewed effort data as effort is registered on wrong activities or is not registered at all. The correct registration of effort is essential for effort monitoring and effort re-estimations, since from this point on the project's own registered effort is the primary data for re-estimations.

During project closure, the post-project phase, the delivered projects are analyzed, and a project final report is drawn. The delivery usually requires an acceptance from the customer after which the project is considered as executed. As a sub-activity, effort analysis is conducted to produce input for a thorough, usually qualitative, project post-mortem analysis. In effort analysis the realized effort of project activities are compared with the estimated, and the reasons for the accuracy or inaccuracy are analyzed and explained. The analysis conducted by focusing to particular sets of activities at time is advantageous, because these sets differ from each other. Effort analysis can produce effort information for improving and calibrating the estimation method, thus improving the software process. Moreover, if new significant activities are identified, they are recorded for activity planning of future projects.

Effort management, however, starts before a project exists and continues also afterwards. These effort management activities include the establishment of the WBS. Also, the effort estimation method to be employed for the software project is either acquired or a proprietary estimation method is constructed, which after the method is initially prepared for estimations, for example with method effort or cost driver calibration. The method is also adjusted after the project has been post-mortem analyzed.

### 3.3 Effort Management Perspectives

The framework includes two perspectives into effort management: Effort distribution on particular activity sets and necessary activities during different framework phases. The different framework phases include a phase prior to a software project, the three phases of the software project lifecycle (pre-project, project, and post-project), and a phase after the project does not exist anymore. The effort-related and activity-related activities are listed in Table 1.

**Table 1: The effort-related and activity-related activities in the effort management framework.**

Framework Phase	Effort-Related Activities	Activity-Related Activities
Prior to Project's Existence	Method Preparation	WBS Establishment
Project: Pre-Project	Estimation	Registration Entity Creation
Project: Project	Collection, Monitor, Re-estimation	Adoption, Effort Registration
Project: Post-Project	Analysis	Analysis
After Project's Existence	Method Adjustment	WBS Adjustment

The other perspective, the effort distribution, is for dividing the whole software project effort into reasonable activity sets, which assists the concentration on the activity sets needing the most attention. Based on our research we propose that the effort of software construction, project management, and non-construction activities be analyzed separately to identify characteristics typical for each category, as it becomes possible to minimize the fluctuation within these categories between projects.

We have been applying these two perspectives together as an effort management framework matrix, which is presented in Section 4. The framework assists to both manage the whole area of effort and, on the other hand, to focus on sub-areas which need a special interest on a particular moment.

### 3.4 Theoretical Background on Key Effort Management Activities and Elements

#### 3.4.1 Work Breakdown Structure (WBS)

A WBS is a particular defined tree-structure the project work is broken into (Wilson and Sifer 1988). An early establishment of a WBS helps to divide the effort into distinct work segments that can be scheduled and prioritized (Agarwal *et al.* 2001). It is very useful to organize project activity elements into a hierarchical structure for project budgetary planning and control purposes. Therefore, the WBS should be sufficiently detailed and modular to cover most software project situations (Boehm 1981).

WBS is also required by the currently employed capability maturity models, e.g. the staged representation of CMMI (SEI 2002b) requires WBS on its lowest maturity level 1. However, no standardized way to create a WBS exists, and the software engineering literature provides only a few general WBS. The proposed work breakdown structures include ones presented with the two COCOMO models (Boehm 1981, Boehm *et al.* 2000), and the Rational Unified process (RUP) activity distribution (Royce 1998), for instance. It is noted that although the concept and practice of using a WBS are well established, the topic is largely avoided in the published literature, mainly because the development of a WBS depends on the project management style, organizational culture, customer preference, financial constraints, and several other project-specific parameters (Royce 1998).

#### 3.4.2 Effort Estimation

Effort can be estimated in several ways, and the estimation techniques can be categorized in different ways (Boehm 1981, Conte *et al.* 1986, Fairley 1992, Sommerville 2001, Pressman 2005). The two empirical estimation techniques: expert judgment including rules of thumb (Fairley 1992), and Function Point Analysis (Albrecht 1979, Albrecht and Gaffney 1983) are the most widely used methods (Gray *et al.* 1999). The regression-based composite estimation models COCOMO (Constructive Cost Model) (Boehm 1981) and COCOMO II (Boehm *et al.* 1995, Boehm *et al.* 2000) are also widely known (Royce 1998, Pressman 2005). The estimation methods and applications, however, produce estimates with varying accuracy. A reason for inaccuracy is the inadequacy of previous project's effort data collection when effort is collected without analyzing it properly. The collected effort data is used for calibrating the weights that adjust the factors and drivers which are used for the effort estimate derivation.

#### 3.4.3 Collecting, Monitoring and Re-Estimating Effort

The software engineering literature (e.g. Royce 1998, Sommerville 2001, Pressman 2005) describes the effort-related activities during project execution (collection, monitoring, and re-estimation) very superficially compared to the effort-related activities during project planning (e.g. effort estimation, scheduling etc.). The importance of the effort-related activities during project execution was raised in (Tsoi 1999), where a framework for software project development management was proposed. This framework concentrated on the pre-project and project phases (referred as acquisition and operation phases, respectively), and especially on the monitoring the effort in a software project, since there is a need for dynamic, continuous measurement on project progress to collect real-time information.

### 3.4.4 Post-Mortem Effort Analyses

In the software industry, it is customary to analyze a project after it has been completed. A post-project analysis, also called the project closure or the post-mortem analysis (Brady and DeMarco 1994, Collier *et al.* 1996, Jalote 2000), is a process of considering the project carefully in detail in order to understand and explain it to learn from the experience, and improve the process based on learning. A sub-process of the post-project analysis, the post-project effort analysis, focuses on the project effort, both the estimated and the realized. Gathering data on a project's actual effort and progress and compare these with the estimates is essential as the effort estimating inputs and techniques are imperfect, the completed project data are needed to improve them (Boehm 1981).

As the software engineering literature (e.g. Sommerville 2001, Pressman 2005) attempts to cover the whole spectrum of software engineering, it understandably provides only rough descriptions of post-mortem analyses. In 1990's, the interest in post-mortem analysis increased in particular (Brady and DeMarco 1994), moving from analyzing the developed system (Kumar 1990) to analyzing the process of how the systems were developed (Collier *et al.* 1996), and from closing the project to improving the software engineering process. A defined process for the project post-mortem review was proposed in the mid-1990's (Collier *et al.* 1996). However, as the proposed process describes post-mortem of the whole project, it provides general guidelines without a detailed method to analyze effort.

## 3.5 Software Process Improvement on Effort Management

Effort management and its improvement can also be the focus of the software process improvement activity. Software process improvement (SPI) means understanding the existing processes and improving them to achieve improved product quality and to reduce costs and development time, and it is usually an activity that is specific to an organization (Sommerville 2001). The SPI activity initiative can arise from employing a capability maturity model. The maturity models are used to improve the related processes. The most widely known and employed capability maturity models include ISO/IEC 15504 (formerly known as SPICE) (ISO 1993), CMM (Paulk *et al.* 1993), and CMMI (SEI 2002a, SEI 2002b) which evolved from CMM. CMMI requires elements of the effort management. Mostly, the effort management relates to the project management process area in CMMI. Like CMM, CMMI requires an organization's measurement repository, which is used to collect and make available measurement data on processes, e.g. effort and cost estimates, and the realized actual effort and costs, to analyze the measurement data (SEI 2002a). Moreover, the staged representation requires work to be arranged as work elements and their relationship to each other and to the end product, i.e. into a work breakdown structure to estimate the scope of the project (at maturity level 1), and to plan the project resources and to manage configurations (at level 2) (SEI 2002b).

SPI and the capability maturity model employment are under enormous interest of research. A number of studies have been conducted applying above mentioned models in software industry (e.g. Rautiainen *et al.* 2002, McBride *et al.* 2004). However, studies describing improvement activity in software project effort management context are not very common. It is also argued that maturity models provide a good basis for SPI, but also an excessive overhead if deployed in full, and on the other hand, do not take enough in consideration that different businesses and situations require different processes (Rautiainen *et al.* 2002). Furthermore, it is argued that maturity models may not be the best models to measure maturity in management processes such as project management, effort management included, as opposed to the technical processes of developing the software, since models have an underlying process model that view software development activities in an industrial production-like fashion, focusing attention on the flow of work from one process to another (McBride *et al.* 2004).

## 4 Application

The effort management framework has been employed at TietoEnator Telecom & Media Ltd. Telecom and media is the largest business area within TietoEnator Corporation, which is the largest IT services company in the Nordic countries. TietoEnator Corporation has over 15,000 employees and activity in

more than 25 countries worldwide. TietoEnator is building a common business system with reference model CMMI. Earlier, due to company diversity and acquisitions, TietoEnator applied both CMM and SPICE (ISO/IEC 15504). There are several internal research development projects on-going as a part of software process improvement including studies improving effort management in the software engineering process.

We have applied the effort management framework in a form of matrix (Fig. 1). In our research we have covered so far all five different phases described in the framework process: the prior project phase, the three project phases (pre-project, project, and post-project), and the after-project phase. Furthermore, we have achieved in identifying important framework activities and linking them together so that they form a robust chain in which the activities relate and give invaluable input to each other. This framework enables us to concentrate to the effort management topics that need most improving effort. Hence, we have at this stage limited our research to a particular set of activities. The activity set in question is the non-construction activities, i.e. the activities which are not directly related to software construction or project management (gray cells in Fig. 1) (Haapio 2004). Software construction involves the effort needed for the actual software construction in the project's lifecycle frame, such as analysis, design, implementation, and testing. Without this effort, the software cannot be constructed, verified and validated for the customer hand-over. Project management involves those activities that are conducted solely by the project group's project manager, such as project planning and monitoring, administrative tasks, and steering group meetings. All the activities in the project's lifecycle frame that do not belong to the other two main categories are non-construction activities. These activities include various management and support activities such as configuration management, customer-related activities, documentation, orientation, project-related activities, quality management, and other miscellaneous planning and support activities, which are carried out by several members of the project. Hypothetically, many of these activities can be eliminated from a software project and the software can still be constructed. In practice, however, a project would be more or less uncontrolled and unsupported without the non-construction activities. This increases both the software construction effort and project management effort needed to get the project accomplished.

		Prior Project	Project Phase			After Project
			Pre-Project (Project Planning)	Project (Project Execution)	Post-Project (Project Closure)	
Effort Distribution	Non-Construction Activities					
	Project Management					
	Software Construction					
		Estimation Method Preparation	Estimation	Collection, Monitor, Re-estimation	Analysis	Estimation Method Adjustment
		Effort-Related Activities				
		WBS Establishment	Registration Entity Creation	Adoption, Effort Registration	Analysis	WBS Adjustment
		Activity-Related Activities				

Figure 1: The applied effort management framework matrix.

Within the effort management framework, our research results include an established work breakdown structure (WBS) of the non-construction activities, the findings on effort distribution of the non-construction activities and their effects on effort estimation and suggestions to optimize and control these activities, the findings on efficient adoption of new effort registration entities of non-construction

activities, and a proposed post-mortem effort analysis method.

The establishment of a WBS makes a ground for the whole effort management and its framework. We established a coherent WBS for the non-construction activities. The identified activities were: configuration management, customer-related activities such as customer support or training, documentation such as manuals or system documentation, orientation, project-related activities such as project events or meetings, quality management, and other miscellaneous planning and support activities. This set of activities seems to cover most non-construction and non-project management activities in software projects.

The established new general WBS includes activities that can be new for some members of the projects where this WBS is introduced and applied into. These new activities require an adoption period. The study on adopting new registration entities in an efficient manner ensures correct effort registrations (Haapio and Ahonen 2006). This in turn ensures that effort data is more reliable for re-estimations and the project's post-mortem effort analysis.

A stepwise post-project effort analysis method was proposed in (Haapio and Eerola 2006). We employed the method with the non-construction activities and gained effort calibration and improving information as result to adjust the employed effort estimation method, and to evaluate the estimation method's weights that adjust the factors and drivers which are used for the effort estimate derivation. Moreover, new software project activities were identified as a result of using the proposed effort analysis method. This information could be used to adjust the established WBS, which in turn recalled a need for the efficient adoption of the new registration entities. Hence, the stepwise post-project effort analysis method proposed supplemented the research into effort estimation method calibration by providing effort calibration information as a result to populate the effort repository from which the effort estimation method's factor weights can be derived.

Our study on the effort of the non-construction activities and the means to optimize and control that effort (Haapio 2004) reported effort proportions of all three main effort categories (software construction, project management, and non-construction activities). The reported means to optimize and control the non-construction activities included a proper pre-project and early project preparation for the non-construction activities, fixed effort of the non-construction activities, establishment of long-term customer relationships, maintaining and increasing project group member's professionalism, and outsourcing the non-construction activities.

## 5 Analysis

This paper introduces a framework to improve effort management in software projects. A comparison between the proposed framework and effort management frameworks is complicated by their differences. For instance, the Software Development Management (SDM) framework (Tsoi 1999) is limited mostly to the pre-project and project phases whereas our framework includes phases before and after these two phases. Moreover, our framework is dynamic considering the effort-related activities, thus the employment of the framework depends on the needs of the organization which wants to improve its effort management. Other proposed frameworks to improve effort management include (Fairley 1992, Vesterinen 1998). These frameworks, however, concentrate mostly on the effort estimation activity whereas the framework proposed in this paper encourages to identify the whole range of effort-related activities concerning a software project. Moreover, the other perspective of our framework, effort distribution, is not included in the other frameworks as a key approach into effort management.

However, the feasibility of the proposed framework is shown in our case studies applying the framework. Moreover, our research on effort management has provided valuable lessons for TietoEnator which, in fact, we believe are extendable to software industry in general. These pragmatic results were gained in research which was directed by the effort management framework. The framework guided us to seek improvement issues on effort management in a focused manner.

Although the findings of our studies can be used for any activity set, our studies were mainly based on cases where the employed activity set was the non-construction activities. Moreover, these studies were focusing only in one of the different phases the framework suggests at a time. However, our research is not limited to just one matrix cell (i.e. the effort activities in improvement focus are not lim-

ited to only one activity set and only one phase). Instead, our studies also include crossing the different activity sets and phases. For instance, our study on the effort of non-construction activities and the means to optimize and control that effort (Haapio 2004) reported effort proportions of all three main effort categories (software construction, project management, and non-construction activities) confirming the effort distribution findings of MacDonell and Shepperd (2003), for instance.

## 6 Discussion

In our research, the SPI activity has focused on improving effort management, an area of project management. The organization in question, TietoEnator Telecom & Media, employ CMMI for assessing and improving its software development processes. However, capability maturity models such as CMM, CMMI and ISO/IEC15504 (SPICE) are focused in the improvement of the technical processes of software development rather than project management (McBride *et al.* 2004). CMMI does not provide direct tools to improve effort management rather than requirements for the software engineering process to consider effort management, including the requirement of organization's measurement repository, a managed process is institutionalized by monitoring and controlling the performance of the process against plans for performing the process and taking corrective actions (SEI 2002a), or the requirement of work to be arranged into a work breakdown structure (SEI 2002b). In other words, the capability maturity models tell 'what' to do (or should be done) rather than 'how' it should be done. In this particular case, the CMMI requirements for one initiated SPI activity to seek out means to improve processes involved in software engineering, project and effort management included. This implies that a maturity model can be a force either to launch or maintain a higher quality level of project management, which was also concluded by McBride *et al.* (2004) with a finding of a correlation between organizational process maturity and the number of monitoring techniques used by project managers. However, our research is rather a SPI activity initiated by CMMI than employment of CMMI, since CMMI considers effort management activities in a fragmented manner. Our SPI activity in turn improves the effort management process which is more in the field of maturity models as being assessed continuously from project to project.

This study complements the existing research on software process improvement and capability maturity models in particular by focusing on a special case of effort management improvement in software projects. Although there has been a vast interest in the concepts of software process improvement and effort in software projects, researchers have, by and large, overlooked a more complete framework of effort management. A more complete view on the software project effort in the form of effort management framework combines the traditionally separate elements of managing the project effort to a single entity where elements influence each other. While the analysis of the studies of the effort management framework presented in this paper are grounded to the particular cases of TietoEnator Telecom & Media, we believe that the framework can be generalized to be employed in other cases within the software industry. The framework is independent from the organization and is relatively general and likely applicable to similar situations in software development. The practitioners, the project or quality managers, can apply the results of this research to derive own adapted version of the effort management framework, e.g. with a suitable effort distribution categorization, in which we believe this study gives a good input.

## 7 Conclusion and Future Research

This paper presented a framework for improving effort management in software projects. In addition, this research supplements also into the research of effort distribution by introducing a coherent set of project activities, namely non-construction activities, which is frequently ignored both in research and the effort management in software industry.

Further, research within the framework on non-construction activities effort is needed. A deeper investigation into non-construction activities and their optimal effort proportions is currently being carried out. Moreover, research is needed to show how the project activities divided into the three categories proposed in this study bring stability into effort proportions and thus better predictability to effort esti-

mation.

### **8 Acknowledgements**

I thank Professor Anne Eerola (University of Kuopio) for her invaluable comments on the draft of this paper. I also thank the support of the KYTKY program (EU funded development program for research co-operation between the software industry and the University of Kuopio). My thanks also go to Minna Haapio, MA (University of Joensuu) for her linguistic advice.

## 9 Literature

- Agarwal R, Kumar M, Yogesh, Mallick S, Bharadwaj R, Anantwar D. 2001. Estimating software projects. *ACM SIGSOFT Software Engineering Notes* **26**(4): 60-67.
- Albrecht A. 1979. Measuring application development productivity. In *Proceedings of the Joint SHARE, GUIDE and IBM Application Development Symposium*, Monterey, California, IBM: 83-92.
- Albrecht A, Gaffney J. 1983. Software function, source lines of code, and development effort prediction: A software science validation. *IEEE Transactions on Software Engineering* **9**(6): 639-648.
- Boehm B. 1981. *Software Engineering Economics*. Prentice Hall: Englewood Cliffs, NJ.
- Boehm B, Clark B, Horowitz E, Westland E, Madachy R, Selby R. 1995. Cost models for future life cycle processes: COCOMO 2. *Annals of Software Engineering* **1**: 57-94.
- Boehm B, Horowitz E, Madachy R, Reifer D, Clark B, Steece B, Brown A, Chulani S, Abts C. 2000. *Software Cost Estimation with COCOMO II*. Prentice Hall: Upper Saddle River, NJ.
- Brady S, DeMarco T. 1994. Management-aided software engineering. *IEEE Software* **11**(Nov.): 25-32.
- Collier B, DeMarco T, Fearey P. 1996. Defined process for project post-mortem review. *IEEE Software* **13**(July): 65-72.
- Conte S, Dunsmore H, Shen V. 1986. *Software Engineering Metrics and Models*. Benjamin/Cummings Publishing Company: Menlo Park, CA.
- Fairley R. 1992. Recent advances in software estimation techniques. In *Proceedings of 14th International Conference on Software Engineering*, Melbourne, Australia, ACM Press: 382-391.
- Gray R, MacDonell S, Shepperd M. 1999. Factors systematically associated with errors in subjective estimates of software development effort: The stability of expert judgment. In *Proceedings of IEEE 6th International Metrics Symposium*, Boca Raton, Florida, USA. IEEE Computer Society: 216-227.
- Haapio T. 2004. The effects of non-construction activities on effort estimation. In *Proceedings of the 27th Information Systems Research in Scandinavia (IRIS'27)*, Falkenberg, Sweden.
- Haapio T, Ahonen J. 2006. A case study on the success of introducing general non-construction activities for project management and planning improvement. In *Proceedings of the 7th International Conference on Product Focused Software Process Improvement (PROFES2006)*, Amsterdam, Netherlands, Springer Lecture Notes in Computer Science Series 4034, Springer-Verlag: 151-165.
- Haapio T, Eerola A. 2006. Post-project effort analysis method. In *Proceedings of the 1st Iberic Conference on Information Systems and Technologies (CISTI2006)*, Esposende, Portugal, Vol. I (Organizational Models and Information Systems), Microsoft: 3-19.
- Iivari J. 1991. A paradigmatic analysis of contemporary schools of IS development. *European Journal of Information Systems*, **1**(4): 249-272.
- ISO. 1993. ISO/IEC TR2 15504, Part 1 - Part 9, Information Technology - Software Process Assessment. ISO.
- Jalote P. 2000. *CMM In Practice: Processes for Executing Software Projects at Infosys*. Addison-Wesley: Reading, MA.
- Järvinen P. 2001. On Research Methods. Opinpajan kirja: Tampere, Finland.
- Kitchenham B, Pickard L, Pfleeger S. 1995. Case studies for method and tool evaluation. *IEEE Software*, **12**(4):52-62.
- Kumar K. 1990. Post implementation evaluation of computer-based information systems: Current practices. *Communications of the ACM* **33**(2): 203-212.
- MacDonell S, Shepperd M. 2003. Using prior-phase effort records for re-estimation during software projects. In *Proceedings of the Ninth International Software Metrics Symposium (METRICS'03)*, IEEE Computer Society: 1-13.
- March S, Smith G. 1995. Design and natural science research on information technology. *Decision Support Systems*, **15**(4): 251-266.



## Session 10: SPI and Measurement

- McBride T, Henderson-Sellers B, Zowghi D. 2004. Project management capability levels: An empirical study. *In Proceedings of the 11th Asia-Pacific Software Engineering Conference (APSEC'04)*, Washington, DC, USA, IEEE Computer Society: 56-63.
- Paulk M, Curtis B, Chrissis M, Weber C. 1993. Capability Maturity Model for Software, Version 1.1. Technical Report, CMU/SEI-93-TR-024, ESC-TR-93-177, CMU/SEI.
- Pressman R. 2005. *Software Engineering: A Practitioner's Approach*. 6th edn. McGraw-Hill: New York.
- Rautianen K, Lassenius C, Vähäniitty J, Pyhäjärvi M, Vanhanen J. 2002. A tentative framework for managing software product development in small companies. *In Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS'02)-Volume 8*, Washington, DC, USA, IEEE Computer Society: 3409-3417.
- Royce W. 1998. *Software Project Management: A Unified Framework*. Addison-Wesley: Reading, MA.
- SEI. 2002a. Capability Maturity Model Integration (CMMI), Version 1.1, Continuous Representation. Technical Report CMU/SEI-2002-TR-028, ESC-TR-2002-028, CMU/SEI.
- SEI. 2002b. Capability Maturity Model Integration (CMMI), Version 1.1, Staged Representation. Technical Report CMU/SEI-2002-TE-029, ESC-TR-2002-029, CMU/SEI.
- Sommerville I. 2001. *Software Engineering*. 6th edn. Pearson Education: Harlow, UK.
- Tsoi H. 1999. A framework for management software project development. *In Proceedings of the 1999 ACM symposium on Applied computing (SAC'99)*, San Antonio, Texas, USA, ACM Press: 593-597.
- Vesterinen P. 1998. A framework and process for effort estimation. *In Proceedings of the 9th European Software Control and Metrics Conference, Proceedings of the 1999 ACM symposium on Applied computing*, Rome, Italy.
- Wilson D, Sifer M. 1988. Structured planning-project views. *Software Engineering Journal* **3**: 134-140.

### **10 Author CVs**

#### **Topi Haapio**

Topi Haapio is a project manager at TietoEnator Telecom & Media Ltd. and a postgraduate at the University of Kuopio, Finland. He received his BS and MS degrees in 1997 and 1998, respectively, both in computer science. Currently, he is writing a dissertation on the effort management in software projects, and his main research interests include software project effort estimation and cost modeling, software process improvement, quality, and project management. His published research includes conference proceedings in IRIS (Information Systems Research in Scandinavia), PROFES (Product Focused Software Process Improvement), and CISTI (Iberic Conference on Information Systems and Technologies).



# Statistical Process Control for Software Development – Six Sigma for Software revisited

Dr. Thomas Fehlmann

Euro Project Office AG  
Zeltweg 50, CH-8032 Zürich, Switzerland  
[thomas.fehlmann@e-p-o.com](mailto:thomas.fehlmann@e-p-o.com)  
[www.e-p-o.com](http://www.e-p-o.com)

**Abstract.** Six Sigma is a popular management strategy for the shop floor. Can it be applied to software? Software development is knowledge acquisition – it is different from industrial engineering. This paper explains the mathematical foundations for statistical thinking when lack of data samples prevents the use of proper statistics. The approach is based on Deming's process chains and Akao's Quality Function Deployment (QFD), using its ability to track requirements consistently through a complex structure of functional topics, and enhanced by Six Sigma metrics for both the quality of goal setting as for execution. The investigation of the mathematical foundations for QFD yields a surprising explanation how QFD and Six Sigma statistics are connected.

## 1 Introduction

When we write Software, we use processes. We have an input, usually customer requirements or – at least – expectations, we expect output such as code, a computer system running some new applications, or services, and we have a set of controls such as time, cost, and other quality attributes. Thus writing software resembles at least in theory a process, as we know from industry. Nevertheless, statistical process control, which has proved to be so useful in many other industries, seems to fail completely, when applied to software engineering.

### 1.1 The Nature of Software Development

Software development is knowledge acquisition – it is different from industrial engineering. Industrial engineering is a transition from an imagination into reality – first is the idea, then the detailed specification, the build and testing, improving the construction if needed until it works, then operation.

Software is different. Initially, we have but a rough idea of what we want, then we must think about it in more detail, find out what resources we need, what technology to use, what the constraints are in terms of time and cost. The more we know about the problem, the better we can define it and break into parts, the closer we get to the solution.

### 1.2 Statistical Process Control

In production, statistical process control is a technique based on collecting data from the output that the process produced, such as physical quality attributes like size, weight, or form, and measure variations. By comparing those measured variations with various other process parameters, root causes for variations become apparent. Eliminating root causes limits these variations. We iterate that until all variations encountered remain within tolerable limits. Such limits we call *tolerance interval*. Process results that lie outside of the tolerance interval we call *defects*. A defect therefore is a strange behavior of the process results that affects the customer or user.

### 1.3 The Metrics for Variance

From statistics, the Six Sigma approach uses the following metrics for variance:

$$\sigma = \sqrt{\frac{n * \sum x_i^2 - (\sum x_i)^2}{n^2}} \quad (1)$$

where  $n$  is the size of the data sample, and  $x_i$  are the measured data components, for  $i=1\dots n$ . The variance  $\sigma$  corresponds approximately to  $1/6$  of the bell's curve base length, if the data distribution is statistically independent. This is why we say, if  $\sigma$  is small enough to fit six times into the tolerance interval, our process has Six Sigma.

### 1.4 Statistical Process Control for Software

Trying to apply this paradigm to software development, we encounter many difficulties. The first problem is, that there is an important time gap between the software development process and the time the software is being used. When testing software, it is difficult to assess which kind of behavior the future users will perceive as strange, and thus rate as a defect. During tests, we only can compare with specifications or requirements, and unwanted behavior detected in reviews or tests are called *bugs*, and usually we try to remove bugs before they can be delivered to customers, who will eventually see them as defects.

However, neither requirements nor specifications are usually detailed enough, we don't know how many defects they contain, and perceptions change over time. Even if we put a lot of effort into strong and detailed specifications and testing during the creation phase of the software life cycle, the bugs detected do not behave statistically "correct". They tend to be everything but statistically independent from each other (for instance, see Fenton [6]).

The next problem is, that it is very difficult to understand what a defect is in software. According Six Sigma, a defect affects the customer's ability to use the software. We distinguish two kinds of failure possibilities:

- *A-defects*, i.e. requirements not detected or nor understood; and
- *B-defects*, i.e. bugs in the delivered functionality.

Some bugs delivered to customers do not affect their ability to use the software, because there is redundancy and a work-around exists. Therefore, it is not even clear whether such failures are B-defects. A-defects, in turn, are even more difficult to detect. The users, without explanation, might reject using software with A-defects.

Writing software requires a highly disciplined approach, just as industrial engineering does. Processes for software development are necessary, but their output is hardly predictable with standard statistical process control.

## 2 Passing Knowledge from One Function to Another

Knowledge is the ability to relate things together. Understanding the cause for one phenomenon enables mankind to predict the likely outcome of similar phenomena – weather forecast is one example, predicting global warming is another one. Parents and schools pass such knowledge to their children in order to help them better to survive and become more successful.

Writing software is similar: developers pass such knowledge to machines. The only particularity is that developers usually know their machines better than the topics they try to teach them. Thus developing software is not just the work of individuals, it is rather a chain of processes that eventually end in some useful software.

### 2.1 Deming's Process Chain

For software, we typically distinguish three kinds of process areas [4] that matter:

1. *Decision processes* – including the market, competition and organizational needs. Usually the deciders materialize in the form of customer requirements, explicitly or implicitly stated needs of the customer. We use the term *customer's needs* to refer to the results of the decider's processes
2. *Enabling processes* – process capability and maturity enabling an organization to write software. Those process kinds typically result in a certain *capability* profile.
3. *Realization processes* – software has to go through a number of processes that transform the customer's needs into design, business scenarios and use cases. In turn, each of those part results connects with its specific test: design with integration test, business scenarios with application tests, and use cases with functional tests.

Fig. 3 contains a sample Deming process chain for software development.

Processes are interdependent. The output of one process constitutes the input of the subsequent process. In the context of software development, we use the name *functional topics* for input and output of the development processes. Functional topics depend from the specific software development processes in use; it might be different whether we develop scenario-based applications that support users in their business, or rather software that controls mechanical or logical functions. In addition, decision processes depend from whether software is developed in-house for internal purpose, or rather as a product or service component.

## 2.2 A Formal Notion of Knowledge – Defining Knowledge Acquisition

For two functional topics  $G$  and  $X$ , *knowledge* is the set of cause/effect relationships, which characterize how  $X$  relates to  $G$ . We need this formal definition for speaking about knowledge acquisition.

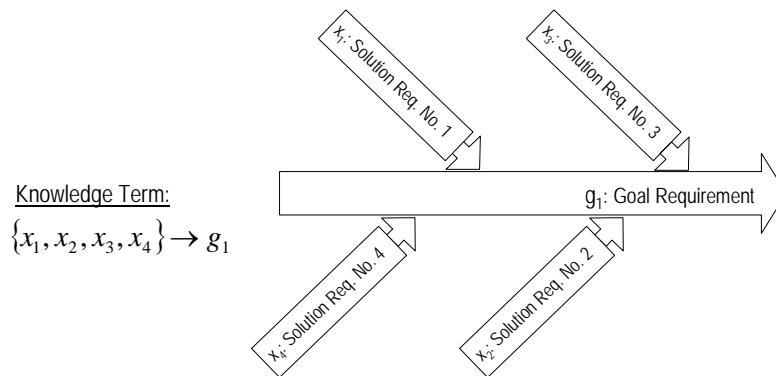
The functional topics consist of *requirements*. A functional topic may have a potentially unlimited number of requirements. We write  $g \in G$  when  $g$  is some requirement for the functional topic  $G$ .<sup>1</sup>

Requirements are bound to functional topics. User requirements and the many kind of technical requirements are different. The failure to understand that difference ([11], [19]) is the main reason why software development processes are difficult to manage.

The generic term “cause” also needs some explanation what it means for software development. The “cause”, why software supports a requirement, is the set of requirements in the function topic one step below. For instance, a certain number of Use Cases supports a required business scenario. Each Use Case, in turn, depends from the classes that implement the functionality required in the Use Case. Knowledge embraces all possible *solution requirements* needed for a goal requirement with regard to the respective functional topics. As an example, take a business transaction for a goal requirement, and the use cases that prepare, execute, and assess the results of that transaction as solution requirements.

Knowledge is always limited and finite, even if functional topics are potentially infinite. The knowledge  $X \rightarrow G$  about a process consists of a finite set of *knowledge terms*  $\{x_1, \dots, x_m\} \rightarrow g$  where  $x_1, \dots, x_m \in X$  and  $g \in G$ ;  $m$  being a finite number.<sup>2</sup>

The knowledge term  $\{x_1, \dots, x_m\} \rightarrow a$  is the formal representation of an *Ishikawa diagram*, where the requirements  $b_1, \dots, b_m$  of functional topic  $G$  are the solution requirements for the requirement  $a$  in functional topic  $X$ .



**Fig. 1.** The Ishikawa diagram, or fishbone diagram, is a tool for cause/effect analysis. It provides a method for visualizing the causes that produces the effect; in our case associating the solution requirements to the goal requirements that they support

The solution requirements  $x_1, \dots, x_m$  are not equally weighted, because they do provide specific contributions to the goal. It is a common practice to distinguish three different levels of cause/effect relationship: weak, medium, and strong. Strong relationship means that the respective solution requirement is a strong influence factor for the respective goal requirement; this corresponds to weight 9. Medium relationship means, it is an indispensable service but might be used in other contexts too; this gives weight 3. “Weak” means useful but not vital, and we assign weight 1. No cause/effect relationship corresponds to weight zero<sup>3</sup>.

<sup>1</sup> Those readers familiar with set theory may think of  $a$  being an element of set  $G$  when reading  $g \in G$ . In this understanding, the functional topic  $G$  consists of all its true statements.

<sup>2</sup> Knowledge Terms are the famous Arrow Terms of Engeler [4] and have been introduced as a foundation for Quality Function Deployment in [7] and [8]. For software, we prefer the notion of “knowledge” rather than the more formal “Arrow”.

<sup>3</sup> See [2], [10], [16], and [23] for a discussion of variations of that practice.

To the knowledge term  $\{x_1, \dots, x_m\} \rightarrow g$  we assign the scalar weights  $\alpha_1, \dots, \alpha_m$ , where the weights  $\alpha_j$  range in  $\{0, 1, 3, 9\}$ , and each scalar  $\alpha_i$  describes the strength of relationship between the cause  $x_i$  and the effect  $g$ .

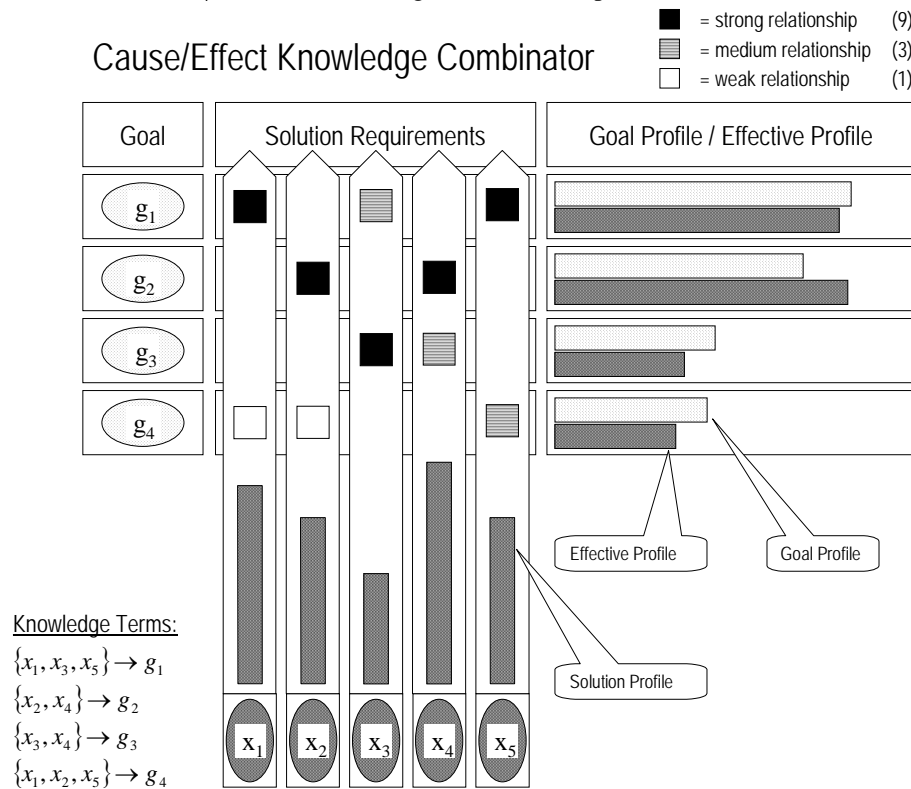


Fig. 2. Generic Cause/Effect combinator relating requirements  $\langle g_1, \dots, g_4 \rangle$  with solution approach  $\langle x_1, \dots, x_4 \rangle$

Now assume that you have a set of  $n$  requirements  $\{g_1, \dots, g_n\}$  from functional topic  $G$ , and for each representative  $g_i$  some knowledge terms  $\{x_{i,1}, \dots, x_{i,m}\} \rightarrow g_i$ , assuming  $i=1, \dots, n$ .

Let  $\alpha_{i,1}, \dots, \alpha_{i,m}$  be the corresponding relationship weights. We can arrange them in a  $n \times m$ -matrix:

$$\varphi = \begin{bmatrix} \alpha_{1,1} & \dots & \alpha_{1,m} \\ \dots & \alpha_{i,j} & \dots \\ \alpha_{n,1} & \dots & \alpha_{n,m} \end{bmatrix} \quad (2)$$

where  $\alpha_{i,j} \in \{0, 1, 3, 9\}$  for  $i=1, \dots, n$  and  $j=1, \dots, m$

We call the matrix  $\varphi$  of relationship weights a *knowledge combinator* (see Fig. 2). Six Sigma knows about such matrices as the method of Quality Function Deployment (QFD) [2]. QFD is widely used as the vital part of “Design for Six Sigma”, see for instance [17], [18], and [22]. According a communication of Prof. Akao, the QFD matrices were invented originally as a convenient form to combine several Ishikawa diagrams for the same functional topic at once.

Well-established techniques exist for characterizing functional topics with only a few requirements [21], [10]. By choosing comprehensive requirements for functional topic  $G$ , you can keep the number of requirements low for that functional topic and thus describe Deming processes by just a few characteristic requirements on both the input and the output side.

### 2.3 Topic Profiles – Measuring Knowledge

With QFD, we have to possibility to measure knowledge acquisition and its variation along the Deming process chain. This constitutes the basic idea behind “Design for Six Sigma”. We do not need to count knowledge in some way. It is sufficient to study the process itself and its results, the requirements.

Let G be a functional topic as before. The topic G may represent the Customer's Needs (CN) that result from the decision processes in the Deming process chain. The requirements of G are not equally graded; there exist different weights. If the effects are not equal, the solution requirements cannot be either. Thus for the solution requirements X in the knowledge  $X \rightarrow G$  there must also exist profiles that give suitable weights to the requirements of X. In practice, this means that the resulting effect on G depends both from the selection of requirements in X, and from their respective weights. Such weight distributions we call (*requirement*) *profiles*.

Usually you have a goal profile for the requirements  $g_1, \dots, g_n \in G$ ; this is the scalar vector  $\langle g \rangle = \langle \gamma_1, \dots, \gamma_n \rangle$ . This vector represents the profile of the desired effects.

Given any solution requirements  $x_1, \dots, x_m \in X$ , we would like to know its solution profile that produces an effect as close as possible to the goal requirements (see Fig. 2). Assume the profile for the cause vector<sup>4</sup> is  $\langle x \rangle = \langle \xi_1, \dots, \xi_m \rangle$ , the  $\xi_j$  representing scalar values for the weights of the solution requirements  $x_j, j=1, \dots, m$ .

The matrix (2) allows for the calculation of the resulting profile when applying knowledge  $X \rightarrow G$  to the functional topic X. Then we calculate the weights of the resulting effects as follows:

$$\varphi(\langle x \rangle) = \langle \varphi_1, \dots, \varphi_n \rangle = \left\langle \sum_{j=1}^m \alpha_{1,j} * \xi_j, \dots, \sum_{j=1}^m \alpha_{n,j} * \xi_j \right\rangle \quad (3)$$

component wise: 
$$\varphi_i = \sum_{j=1}^m \alpha_{i,j}, i=1, \dots, n$$

Again, the  $\alpha_{i,j}$  denote the QFD matrix elements, which represent the weights of the relationship between solution requirements and effects ( $i=1, \dots, n; j=1, \dots, m$ );  $\varphi$  is a mapping from a vector with m components into another vector with n components. We call the resulting profile  $\varphi(\langle x \rangle) = \langle \varphi_1, \dots, \varphi_n \rangle$  the *effective profile*. Unfortunately, the effective profile will not automatically match the original goal profile  $\langle g \rangle$ . With  $\langle x \rangle$  selected arbitrarily, there is a gap between the vectors  $\langle g \rangle$  and  $\varphi(\langle x \rangle)$ . The only commonality between the two vectors is that they both have the same number of components, namely n.

Formula (3) allows assessing the effects of measured topics, back up the process chain. For instance, we can measure the functional size [1], [12] of the specified software that is the result of the Use Case (UC) process (see Fig. 3). This is knowledge acquisition of the form  $UC \rightarrow BS$ , where we must find the best selection of use cases that meet the requirements for the Business Scenarios (BS). Let  $\langle u \rangle$  be the functional size profile for a representative set of use cases. Using the knowledge combinator  $\beta$  from  $UC \rightarrow BS$ , their effective profile is  $\beta(\langle u \rangle) = \langle \beta_1, \dots, \beta_n \rangle$ , according formula (3).

The effective profile  $\beta(\langle u \rangle)$  tells us the relative cost distribution for the business scenarios. This technique is very effective; it solves the problem for many CIO how he should value ICT services.

However, this is not all. We can track the effective profile  $\beta(\langle u \rangle)$  yet another step back and, using the knowledge combinator from  $BS \rightarrow CN$ , find out which customer's needs receive most attention by the functionality provided. Thus, we can track the use cases' cost profile back to the value perceived by the customer. Such transparency avoids spending effort for features and functions that yield no value for the customer. For product managers, this information is of indispensable.

It takes idling out of the software development process. Similarly, we may measure test results, compute defect density and track measurement results back to see whether what we tested was of any importance to the customer.

## 2.4 Comparing Profiles – Analyzing Knowledge

However, how do we know whether our knowledge combinators are accurate enough to allow for such backtracking? We need a metric that tells us how well our knowledge terms models the software development processes. For this, we need statistical process control for the requirement profiles.

To compare vectors, it is not sufficient to compare their difference. Although you can compute the difference vector as soon as you have the same amount of components, the result may be useless unless the components of the two vectors are of comparable size. In order to achieve that, we need *normalization*.

When drawing a profile, it will not change its graphical appearance when multiplying each coefficient by the same number; only the size of the graphics will adapt. In order to compare two profiles, we need to gauge the profiles to the same size.

The *length* of a profile  $\langle g \rangle = \langle \gamma_1, \dots, \gamma_n \rangle$  is denoted as  $|\langle g \rangle|$  and defined by:<sup>5</sup>

<sup>4</sup> We use the square brackets  $\langle \dots \rangle$  to distinguish vectors from scalars.



$$|\langle g \rangle| = \sqrt{\sum_{i=1..n} \gamma_i^2} \quad (4)$$

Because QFD teams generally prefer scaling importance in the range 0...5, we normalize our profiles to vectors of length 5 using the formula

$$\frac{\langle g \rangle * 5}{|\langle g \rangle|} = \left\langle \frac{\gamma_1 * 5}{|\langle g \rangle|}, \dots, \frac{\gamma_n * 5}{|\langle g \rangle|} \right\rangle \quad (5)$$

With this normalization (5) done, we can compare two profiles based on their direction in the vector space only, because all these vectors have length 5. Two profiles pointing in the same direction represent the same knowledge about the process and its results.

The requirement profiles, as normalized vectors, show the direction our project has to go on our quest for knowledge acquisition. Furthermore, it is possible to eliminate requirements that are not contributing to the desired effect. If the weight of some requirement becomes zero, we may not need that and save all cost related to it. If our solution approach does not point into the right direction, we must change our selection of solution requirements. This simply means, find a better solution.

### 2.5 The Convergence Factor – Improve the Cause/Effect Relationship

We need a metric to measure how well our choice of solution profile matches the goal. This metric we call the *convergence factor*. It is the length of the profile difference between goal profile and effective profile, divided by the number of profile coefficients.

Let  $\langle g \rangle = \langle \gamma_1, \dots, \gamma_n \rangle$  be the goal profile, let  $\langle x \rangle = \langle \xi_1, \dots, \xi_m \rangle$  be the solution profile and  $\varphi(\langle x \rangle) = \langle \varphi_1, \dots, \varphi_n \rangle$  be the effective profile. Then the *convergence factor* is the square root of the length of its profile difference  $\langle g \rangle - \varphi(\langle x \rangle)$  divided by the number of goal coefficients n:

$$\kappa = \sqrt{\frac{\sum_{i=1..n} (\gamma_i - \varphi_i)^2}{n}} \quad (6)$$

(Convergence Factor)

A convergence factor ( $\kappa$ ) of zero means complete convergence.  $\kappa = 0.2$  is generally considered good;  $\kappa = 0.5$  is acceptable, as the latter indicates a deviation of direction by 10%.  $\kappa$  greater than one indicates a significant difference between the goal profile and the profile achieved with the chosen solution approach. It is management's decision how accurate the development shall keep its original direction.

When we have a matrix with a bad convergence factor, there are two distinct solution requirements:

1. Add better requirements to the solution profile that better supports the goal profile (e.g. better fit customer's needs), until the convergence factor decreases. This is the preferred way experienced by QFD practitioner and works in a workshop setting.
2. Use the convergence factor as the minimum target for linear optimization. There are standard mathematical algorithms that reliably decrease the convergence factor by optimizing the solution profile.

Linear optimization finds a local optimum but does not detect all possible solutions. It cannot replace the search for better solution requirements. For more details regarding linear optimization with QFD, see [8].

---

<sup>5</sup> For n = 2, this is the theorem of Pythagoras:  $|\langle y \rangle| = \sqrt{\zeta_1^2 + \zeta_2^2}$ .

## 2.6 The Three Metrics for Six Sigma for Software – Controlling Development

Based on the convergence factor, a network of knowledge combinators allows both forward setting the required profiles for the requirements, and backwards controlling whether software development met those profiles, see Fig. 3.

Let  $\langle g \rangle$  be the goal profile,  $\langle x \rangle$  the optimized solution profile and  $\langle x_0 \rangle$  the measured solution profile. Then  $\varphi(\langle x \rangle)$  is the effective goal profile and  $\varphi(\langle x_0 \rangle)$  the measured impact profile. With these three vectors we define three metrics:

- *Convergence Factor*: The difference between goal profile  $\langle g \rangle$  and effective goal profile  $\varphi(\langle x \rangle)$  shows how accurate the cause–effect – relationship is. It is a metric for the achievement of objectives under ideal conditions.
- *Effective Deviation*: The difference between goal profile  $\langle g \rangle$  and measured impact profile  $\varphi(\langle x_0 \rangle)$  shows how well the target goals are effectively met. It is a metric for the achievement of objectives as perceived by others.
- *Process Maturity Gap*: the difference between effective goal profile  $\varphi(\langle x \rangle)$  and measured impact profile  $\varphi(\langle x_0 \rangle)$  shows the gap between what your development processes are aiming for and what they achieved in the real world. It is a metric for the maturity of your development process.

It is possible to use more than one measurement per functional topic, and thus combine measurements and get effective deviations from different solution topics.

The convergence factor is the most important metric, as it tells whether the knowledge combinators explain the knowledge acquisition process sufficiently well. It indicates whether the knowledge combinators are good enough.

The effective deviation reports adherence to the goals set for development and depends from software metrics collected during software development.

The process maturity gap also depends from software metrics. It is a metric for process capability. It is similar, but not equal, to the CMM capability metrics [13]. In contrary to CMM, it measures the process capability on those functional topic levels that provide the business benefit. It may even complement the CMM metrics, as it transforms the continuous CMMI capability metrics into benefits for business, see [7].

## 2.7 The Deming Process Chain for Software Development

The knowledge combinators provide the metrics that define – using formula (6) – the statistical variance that we must expect when developing software. The Deming process chain yields metrics that let us control the deployment of requirements throughout the software development. Thus, formula (6) plays the same role as the formula (1) for such processes that do not produce statistically relevant data samples. The Six Sigma metrics for software indicate whether the development processes produce results that meet customer's expectations, or whether it went off way.

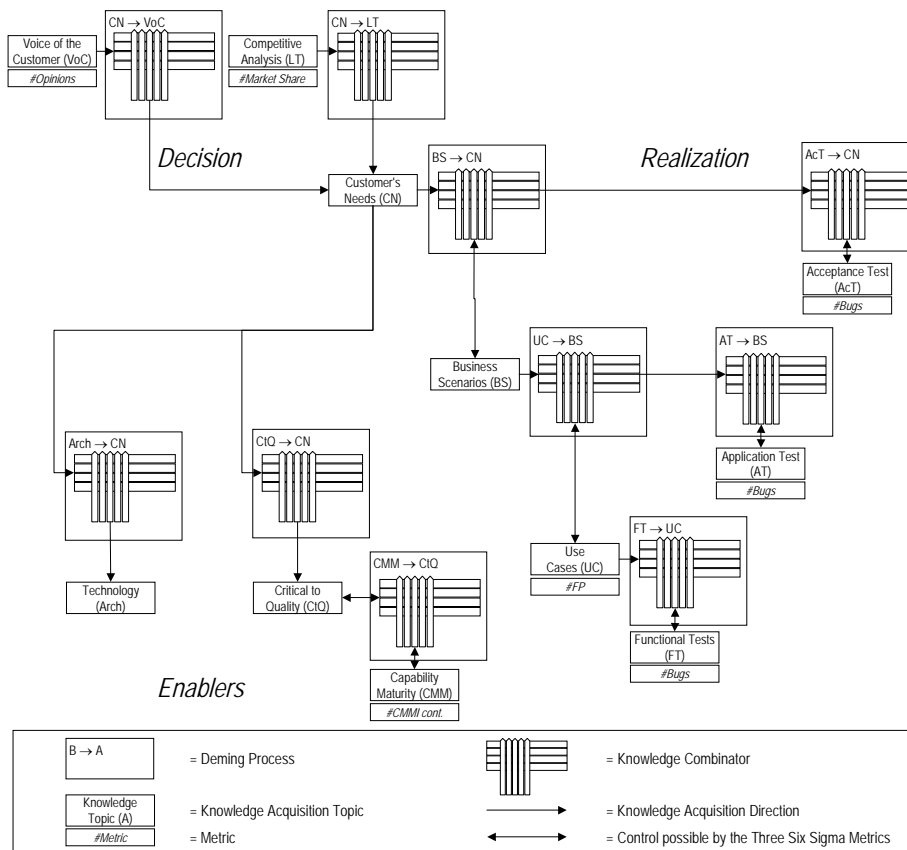


Fig. 3. Deming process chain for software development with its knowledge combinators

## 3 Benefits

### 3.1 Practical Experiences

The framework outlined in this report has evolved over time in more than 15 years. It has been extensively tested in a number of projects, and it proved to be of outstanding value. These experiences had been reported on several occasions [9].<sup>6</sup> However, the understanding why it worked grew only over time.

The striking correspondence between formula (1) and (6) explains these successes. What we observed in practice as a working theory is in fact based upon the same mathematical structure.

### 3.2 Conclusion

We conclude from the observations made in practice that the theory seems to serve well as a model for software process improvement.

Deming's generic process model is very useful to describe software development. The key insight provided by that model is that requirements for software are bound to functional topics. Experience and recent advantages in software engineering [3] demonstrate that solution requirements drive the software development processes.

The metrics derived from Akao's Quality Function Deployment provide the same benefits as statistical process control but rely on analyzing knowledge acquisition rather than statistical regression analysis. Mathematically, the metrics are very similar in structure, provided one assumes the normalization formula (4) originally proposed in [8]. For software development process engineering, this is a break-through, because it allows doing statistical process control based on cause/effect analysis rather than statistical data samples.

<sup>6</sup> Experience reports are available on the web site of the author ([www.e-p-o.com](http://www.e-p-o.com)).

Although the mastery of both Six Sigma statistics and Linear Algebra is a challenge for Six Sigma Green or Black Belts, as well as for software engineering process groups, it is worth the effort because software is the most important single component for most products and services. Six Sigma will be expanded into software, and thus, mastery of software development and operational use will become decisive for the economic future of our societies.

## References

1. Abran, A. et. al. (2003), COSMIC-FFP Measurement Manual - The COSMIC Implementation Guide for ISO/IEC 19761: 2003, Version 2.2, The Common Software Measurement International Consortium (COSMIC), Montréal, Kanada
2. Akao, Y. et. al. (1990), *Quality Function Deployment (QFD)*; Productivity Press, University Park, IL
3. Beck, K. (2002), *eXtreme Programming explained*, Addison-Wesley, Boston, MA
4. Deming W. Edwards, "Out of the Crisis", MIT Center for Advanced Engineering Study, Cambridge, Mass., 1982
5. Engeler E. (1995), *The Combinatory Programme*, Birkhäuser, Basel, Schweiz
6. Fenton, N.; Krause, P.; Neil, M. (1999), "A Probabilistic Model for Software Defect Prediction", IEEE Transactions on Software Engineering, New York, NY
7. Fehlmann, Th. (2003), "Strategic Management by Business Metrics: An Application of Combinatory Metrics", *International Journal of Quality & Reliability Management*, Vol. 20 No. 1, Emerald, Bradford, UK
8. Fehlmann, Th. (2004), "The Impact of Linear Algebra on QFD", in: *International Journal of Quality & Reliability Management*, Vol. 21 No. 9, Emerald, Bradford, UK
9. Fehlmann, Th. (2005), *Six Sigma in der SW-Entwicklung*, Vieweg-Verlag, Braunschweig-Wiesbaden
10. Herzwurm G., Mellis, W., Schockert, S. (1997): *Qualitätssoftware durch Kundenorientierung*. Die Methode Quality Function Deployment (QFD). Grundlagen, Praxisleitfaden, SAP R/3 Fallbeispiel.
11. Humphrey, W.S. (1989), *Managing the Software Process*, Addison-Wesley, Boston, MA
12. International Function Points User Group IFPUG (2004), IFPUG Function Point Counting Practices Manual, Release 4.2, Princeton, NJ
13. Kulpa, M., Johnson, K. (2003), *Interpreting the CMMI*, Auerbach Publications, CRC Press, Boca Raton, FL
14. Lauesen, S. (2002), *Software Requirements: Styles and Techniques*, Addison-Wesley, Boston, MA
15. Mizuno, Sh. ed. (1988), "The 7 New QC Tools", in: *Management for Quality Improvement*, Productivity Press, University Park, IL
16. Mizuno, Sh. and Akao, Y. ed. (1994), *QFD: The Customer-Driven Approach to Quality Planning and Deployment*, translated by Glenn Mazur, Tokyo: Asian Productivity Organization, Tokyo, Japan
17. Oudrhiri, R. (2005): "Six Sigma and DFSS for IT and Software Engineering", in: TickIT International, 2Q05, TickIT Office, London, UK, ISSN 1354-5884
18. Pande, P.S., Neuman, R.P., Cavanagh, R.R. (2002), *The Six Sigma Way – Team Fieldbook*, McGraw-Hill, New York, NY
19. Paulk, M.C., Curtis, B., Chrissis, M.B., and Weber, Ch. V. (1993), *Capability Maturity Model for Software, Version 1.1*, Software Engineering Institute, CMU/SEI-93-TR-24, DTIC Number ADA263403, Carnegie-Mellon University, Pittsburg, PA
20. Rico, D.F. (2004), ROI of Software Process Improvement: Metrics for Project Managers and Software Engineers, J. Ross Publishing, Boca Raton, FL.
21. Saaty, Th. (1999), „Fundamentals of the Analytic Network Process“, in: ISAHP 1999, Kobe, Japan
22. Töpfer, A. ed. (2004), *Six Sigma – Konzepte und Erfolgsbeispiele für praktizierte Null-Fehler Qualität*, 3. Auflage, Springer Verlag Berlin Heidelberg New York
23. Zultner, R. E. (1992), "Quality Function Deployment (QFD) for Software: Structured Requirements Exploration", in: Schulmeyer/McManus (ed.): *Handbook of Software Quality Assurance*, 2<sup>nd</sup> Ed., Zürich 1992, S. 297-319



# Generating a Work Breakdown Structure: A Case Study on the General Software Project Activities

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

## **Abstract**

A proper work breakdown structure (WBS) is essential in performing successful project effort estimation and project management. The use of WBS is also required on the level 1 of CMMI. However, general WBS for software projects are in short supply. In this paper, 26 different software projects were examined in a case study to generate a WBS. The activities of the generated WBS were general software project activities, which are necessary but not directly related to the actual software construction. The generated WBS was compared with four existing work breakdown structures provided by the software engineering literature.

## **Keywords**

WBS, Grounded Theory, effort management, project management, SPI, COCOMO, RUP

### 1 Introduction

Since the beginning of software projects and their effort estimations, work has been broken down into smaller entities (project activities and phases) to manage and estimate software project effort and cost easier. The work breakdown structure (WBS), a particular defined tree-structure the project work is broken into (Wilson and Sifer 1988), was applied even in the early effort estimation models, e.g. the Wolverton Model (Wolverton 1974) and COCOMO (Boehm 1981). An early establishment of a WBS helps to divide the effort into distinct work segments that can be scheduled and prioritized (Agarwal *et al.* 2001). It is very useful to organize project activity elements into a hierarchical structure for project budgetary planning and control purposes. The WBS should be sufficiently detailed and modular to cover most software project situations (Boehm 1981). It is noted that a good WBS and its synchronization with applied process framework are critical factors in software project success (Royce 1998), especially in the case of a project employing a spiral software engineering process.

WBS is also required by the currently employed capability maturity models, e.g. the staged representation of CMMI (SEI 2002b) requires WBS on its lowest maturity level 1. However, no standardized way to create a WBS exists, and the software engineering literature provides only a few general WBS. It is noted that although the concept and practice of using a WBS are well established, the topic is largely avoided in the published literature, because the development of a WBS is dependent on the project management style, organizational culture, customer preference, financial constraints, and several other project-specific parameters (Royce 1998). Hence, the applied activity sets in software projects are either too general or project-specific. To benefit effort management in the project, the activities should be broken down into a suitable granularity level, which in turn leads to activity sets that differ from each other from project to project. However, some of the software project activities are general activities that occur in most projects. General WBS enables realized project effort to be comparable between projects, thus improving future effort estimations.

The generation of general project activities into a coherent WBS is examined in this study. These activities, namely the non-construction activities, include various management and support activities, each carried out by several members of the project, i.e. these activities are not directly related to software construction or project management (Haapio 2004). Neglecting the non-construction activities in project planning and effort estimation may result in significant effort proportion deviation between projects. Furthermore, regardless of the effort estimation method employed, even the best estimate cannot be accurate if the method neglects or undervalues the non-construction activities that are involved in the project. These activities comprise a relevant proportion (mean 18.5%) of total project effort (Haapio 2004). Effort estimation research has focused on software construction because the majority of the total effort is software construction effort (Boehm *et al.* 2000, MacDonell and Shepperd 2003). However, the rest of the project effort is just as important when a higher estimation accuracy is pursued. A simple fixed effort proportion overhead is not sufficient in reaching accurate estimates, since the effort proportions of the non-construction activities vary from project to project. Hence, their effort has to be estimated, collected, and analyzed project-specific and per activity.

This study is a part of a larger project aiming to improve effort estimation and effort management by focusing on activities other than the actual software construction or project management. The non-construction activities have been previously identified and examined (Haapio 2004, Haapio and Ahonen 2006, Haapio and Eerola 2006). This study, however, takes a new approach to identify the non-construction activities by employing loosely the coding procedures of the Grounded Theory methodology (Glaser and Strauss 1967). The elements of the Grounded Theory methodology were chosen to achieve reliable results by applying a practical approach. The methodology provides a systematic, yet an easy-to-adapt formal approach for identifying and classifying different categories. It has been applied increasingly in software engineering and information system research (Hansen and Kautz 2005).

The rest of the paper is structured as follows. Section 2 describes the research methodology and the theoretical background. The case study is presented in Section 3. The analysis of the study is presented in Section 4, followed with a discussion in Section 5. A brief conclusion and suggestions for future research are given in Section 6.

## 2 Research Methodology and Theoretical Background

### 2.1 Research Methodology

The primary research methodology for this paper is case study (Järvinen 2001), in which the elements of the Grounded Theory methodology's procedures are adapted to emerge the results from the data. The aim of Grounded Theory (Glaser and Strauss 1967) is to develop a theory from the data rather than to gather and use data to test a theory. The theory is grounded in the reality as represented in the data. The transformation of the data to a theory is achieved with coding (Goede and de Villiers 2003). Three different types (phases) of coding have been identified for the transformation: open coding, axial coding, and selective coding. These codings are done partly simultaneously, and the division between them is vague due to the iterative nature of the process. During open coding, the data material is opened up (Goede and de Villiers 2003, Hansen and Kautz 2005). Categories emerge from similar concepts that have similar properties. Preliminary categories can be added based on the knowledge on software engineering literature, for example, but the categories are grounded only as the observations emerge from the data, and are thus verified. Open coding fragments the data. The purpose of axial coding is to reassemble data that was fractured during open coding. The new categories are formed by linking categories with the same properties. The last coding phase, selective coding, is a process of integrating and refining the theory. In previous coding phases the different categories have been formed. These categories are only descriptions of data and not yet a theory. To form the theory, various categories need to be integrated. One of these categories becomes the central category that represents the main theme of the research. Coding is iterated until the categories are saturated, i.e. more coding does not alter the descriptions of the categories (Goede and de Villiers 2003). In this paper, the above mentioned coding procedures of the Grounded Theory are applied in a rather loose manner to keep the coding as pragmatic as possible.

### 2.2 Theoretical Background

In many cases, the activities and lifecycle phases of software projects have served as the basis for the different factors used for effort-counting and as a basis for a WBS. However, there are not many general software project work breakdown structures available. The proposed work breakdown structures include ones presented with the two COCOMO models (Boehm 1981, Boehm *et al.* 2000), and the Rational Unified process (RUP) activity distribution (Royce 1998), for instance.

A WBS provides an information structure which includes a delineation of all significant work, a clear task decomposition for assignment of responsibilities, and a framework for scheduling, budgeting, and expenditure tracking. It is argued that the conventional work breakdown structures frequently suffer from three fundamental flaws: (1) they are prematurely structured around product design, (2) they are prematurely decomposed, planned, and budgeted in either too much or too little detail, and (3) they are project-specific, and cross-project comparisons are usually difficult or impossible (Royce 1998). Therefore, in an evolutionary WBS the planning elements should be organized around the employed process framework rather than the product framework. The WBS should consist of three levels: The workflows (management, environment, requirements, design, implementation, assessment, and deployment), each phase of the project lifecycle (inception, elaboration, construction, and transition), and the artifacts of each phase (Royce 1998). These recommendations were the premises for RUP WBS for spiral software engineering process.

Simplified, a WBS is a hierarchy of elements that decomposes the project into discrete work tasks (Royce 1998). In other words, tasks are defined for the product as a whole or for individual functions in a form of a WBS (Pressman 2005). There can be many parameters that drive the decomposition of work into discrete tasks: functions, components, lifecycle phases, for instance (Royce 1998).

Boehm (1981) suggested two hierarchies: the product hierarchy and the activity hierarchy. The product hierarchy indicates how the various software components fit into the overall software system, and the activity hierarchy indicates the various activities which may deal with a software component. In



practice, WBS elements—in both hierarchies—are defined only down to the level necessary for effort control and reporting.

Boehm (1981) suggested a WBS for projects employing a waterfall project lifecycle process. The WBS is applied with the COCOMO cost estimation model as the model estimates how effort is distributed on different activities between eight major categories, namely requirement analysis, product design, programming, test planning, verification and validation, project office functions, configuration management and quality assurance, and manuals, each having specific activities in the four project lifecycle phases.

COCOMO was enhanced to COCOMO II in the mid-1990's in order to develop the model to address the new needs of evolving software engineering such as distributed software and component techniques. COCOMO II has been developed to be usable by projects employing either waterfall or spiral software engineering processes. In the waterfall approach of COCOMO II, software activity work was divided into five major categories: management, system engineering, programming, test and evaluation, and data. This breakdown was adapted from the previous COCOMO's WBS eight categories, which were partly reorganized and renamed. The requirements, product design, and configuration management and quality assurance categories were organized as system engineering subactivities. 'Verification and validation' was renamed 'test and evaluation', 'manuals' became 'data', and 'project office functions' became 'management' (Boehm *et al.* 2000).

The COCOMO II spiral approach is based on the approach applied in RUP. The RUP default WBS is based on seven main categories, namely management, environment, requirements, design, implementation, assessment, and deployment. Each of these seven categories includes activities relating to four project phases (inception and elaboration of the engineering stage, and construction and transition of the construction stage) (Royce 1998). The COCOMO II WBS for spiral process is based much on the same seven categories, yet RUP's 'environment' became 'environment and configuration management' in COCOMO II. Also, the activities within the four phases partly differ (Boehm *et al.* 2000).

Software project work breakdowns can also be in the focus of the software process improvement activity. The activity initiative can arise from employing a capability maturity model. The maturity models are used to improve the related processes. Software process improvement (SPI) means understanding the existing processes and improving them to achieve improved product quality and to reduce costs and development time, and it is usually an activity that is specific to an organization (Sommerville 2001).

The most widely known and employed capability maturity models include ISO/IEC 15504 (formerly known as SPICE) (ISO 1993), CMM (Paulk *et al.* 1993), and CMMI (SEI 2002a, SEI 2002b) which evolved from CMM. The staged representation of CMMI requires work to be arranged as work elements and their relationship to each other and to the end product, i.e. into a work breakdown structure to estimate the scope of the project (at maturity level 1), and to plan the project resources and to manage configurations (at level 2) (SEI 2002b).

SPI and the capability maturity model employment are under enormous interest of research. However, studies describing improvement activities in the context of generating a WBS are not very common.

### 3 The Case Study

The study investigated the generating of a set of project activities, namely the non-construction activities, into a WBS at TietoEnator Telecom & Media. Telecom and media is the largest business area within TietoEnator Corporation. TietoEnator Corporation is the largest IT services company in the Nordic countries with over 15,000 employees worldwide, and activity in more than 25 countries. TietoEnator is building a common business system with reference model CMMI. Earlier, due to company diversity and acquisitions, TietoEnator applied both CMM and SPICE (ISO/IEC 15504). There are several internal research development projects going on as a part of software process improvement including studies to improve effort management in the software engineering process.

This study is qualitative, with the aim of generating a work breakdown structure for general software project activities. The study applied the Grounded Theory methodology loosely. In the strict sense and spirit of this methodology, the theory emerges from the data without external influences when the

three coding procedures of the Grounded Theory have been applied. In this study, however, these procedures were used with the pre-knowledge of software engineering literature (Royce 1998, Sommerville 2001, Pressman 2005) to generate the final work breakdown structure. The research process with the results are described in the following.

**1. The gathering of data sample.** The gathering of the data sample consisted of selecting the projects into the sample, and importing the data from the work time registry system.

The dataset consisted of 26 custom software development projects which took place in 1999-2005. These projects were delivered to five different Nordic customers who operate mainly in the telecommunication business domain. The duration of the projects was between 2 and 53 months (mean 15.3). The projects were carried out by different teams within the same business unit. The delivered software systems were based on different technical solutions. However, the two most common technologies were based either on J2EE or on client/server with transaction management.

All projects used a waterfall-like software development process. The project lifecycle started either from the analysis phase or the design phase, and continued with implementation, verification and validation, and the deployment phases. This span forms the project's lifecycle frame, which is quite typical for delivery software projects for this organization.

One key demand in Grounded Theory is that the theory is discovered in the data, and not to test a prewritten hypothesis by gathering appropriate data (Goede and de Villiers 2003). In this study, the data sample, i.e. projects and related project data, was gathered random from the work time registry system. The project personnel feed their work hours for different project-specific activities (registration entities) in the work time registry system at half-hour precision. The project data was exported into project-specific spread sheets. The project data included several columns of data, of which the information on project's name, the registration entity titles and the one sentence entity descriptions were captured on new spread sheets for the analysis.

**2. The definition of the activity category for the WBS.** The definition of the activity category for WBS consisted of defining the main software project activity categories and deciding upon the one for generating the WBS.

In this study, the project activities in the focus for the WBS are the non-construction activities, i.e. the activities which are not directly related to software construction or project management (Haapio 2004). These three main categories (software construction, project management, and non-construction activities) comprise the total software project WBS. Software construction involves the effort needed for the actual software construction in the project's lifecycle frame, such as analysis, design, implementation, and testing. Without this effort, the software cannot be constructed, verified and validated for the customer hand-over. Project management involves activities that are conducted solely by the project group's project manager, such as project planning and monitoring, administrative tasks, and steering group meetings. All the activities in the project's lifecycle frame that do not belong to the other two main categories are non-construction activities. These activities include various management and support activities, which are carried out by several members of the project.

**3. The coding of the activities within the main category.** The coding of the activities within the non-construction activity category included identifying and extracting the activities from the data, a uniform renaming of the activities, and iterative categorizations into logic entities.

In the first coding phase, which relates to the Grounded Theory's open coding, the activities of 26 projects were coded into three main categories: the activities related to the software construction, the project management and the non-construction activities, based on the definitions described above. The 26 projects comprised 1,390 activity registration entities in total (minimum 9, maximum 275, median 32.5, and mean 53.5 entities per project) of which 282 were non-construction activities (minimum 1, maximum 27, median 8, and mean 10.8 entities per project).

During the second coding iteration, these 282 non-construction activities were reduced down to 20 categories by renaming uniformly the activities meaning the same. Out of the total 282 non-construction activities, 30 different entities appeared more than once with exactly same title and description. The titles and descriptions of the entities individually considered, 34 different entities appeared more than once with same title and 32 appeared more than once with same description. The resulted twenty activity categories within the non-construction activities category were: Configuration and Version Management, Configuration Management, Customer Inquiries, Customer Support, Cus-

tomer Training, Deployment, Deployment and Version Management, Documentation, Orientation, Project Events, Project Group Meetings, Project Group Working, Project Start-Up, Quality Assurance, Reviews, Reviews & Quality Assurance, Technical Environment Maintenance, Technical Environment Setup, Version Management, and Miscellaneous. The miscellaneous category involved individual activities which occurred in and were created as registration entity for a particular project. The projects involved different compounds of the quality assurance and the review activities: both of them independently, only one of them, or both of them combined.

The third coding iteration with 20 activity categories resulted into final seven categories, which after these categories were defined. The definitions and re-coding were partially based on the input from the software engineering literature. The categories with their definitions include:

- **Configuration Management** = {Configuration and Version Management, Configuration Management, Deployment, Deployment and Version Management, Technical Environment Setup, Technical Environment Maintenance, Version Management}. Configuration management (CM), or software configuration management (SCM), is strongly related with change management. In fact, (S)CM can be considered either as change management (Pressman 2005) or change management as a part of CM (Sommerville 2001). CM is the development and application of standards and procedures for managing evolving system product. CM consists of configuration management and its planning, change management, version and release management, and system building (Sommerville 2001). A slightly different perspective into CM is provided by Pressman (2005), where software configuration management (SCM) is defined as a set of umbrella activities that have been developed to manage change throughout the lifecycle of software. These activities include identification of objects in the software configuration, change control, version control, configuration auditing, and reporting. In this paper, CM includes also the management of the technical surroundings of the software. However, the actual change management work is here excluded, since it is not in the scope of the original effort estimation, and programming work is included into the software construction category. CM had been created as a registration entity in the work time registry system for 23 out of 26 projects, i.e. the frequency of the CM activity was 88.5%.
- **Customer-Related Activities** = {Customer Inquiries, Customer Support, Customer Training}. The customer-related activities include activities that are dependable of the customer, i.e. these activities exist only because of the customer involvement in the project. The customer-related activities include activities with customer and end-user interaction. These stakeholders are in many cases the same but can also be from different organizations (Pressman 2005). Within the projects, the frequency of the customer-related activities was 57.7%.
- **Documentation** = {Documentation}. Documentation includes all other documentation than actual software analysis or design documentation, e.g. system documentation, manuals, standards etc. For instance, the software user manual provides the user with the reference documentation necessary to support the delivered software (Royce 1998). Within the projects, the frequency of the documentation activity was 69.2%.
- **Orientation** = {Orientation}. Orientation is here considered as the knowledge transfer of software's business domain, technology etc. to a project group member, i.e. all learning activities involved with the particular project. Within the projects, the frequency of the orientation activity was 65.4%.
- **Project (Group and Working) Related Activities** = {Project Events, Project Group Meetings, Project Group Working, Project Start-Up}. All activities related to the functionality of a particular project as an organization are included in this category. These organizations are called software teams (Pressman 2005) or project groups. Within the projects, the frequency of the project group related activities was 73.1%.
- **Quality Management** = {Quality Assurance, Reviews, Reviews & Quality Assurance}. In the software engineering literature, quality management (QM) is considered as an umbrella activity consisting of the management and assurance of quality, and including the reviews of documentation and software code (Sommerville 2001, Pressman 2005). Simplified, QM involves defining procedures and standards which should be used during software development and checking that these have been followed (Sommerville 2001). QM has three principal activities: Quality assurance, quality planning, and quality control. Quality assurance (QA) defines a framework for achieving software quality (Sommerville 2001). QA consists of a set of auditing and reporting functions that assess the effectiveness and completeness of quality control activities such as formal technical reviews (Pressman 2005) or peer inspections (Royce 1998). In this study, the reviews and quality

assurance activities were first coded into own categories, but as the coding proceeded, it was found out that in some projects they were combined, i.e. the project managers view these activities be strongly related with each other. Therefore, these activities were combined into one Quality Management activity. Within the projects, the frequency of the QM activity was 73.1%.

- **Miscellaneous** = {Miscellaneous}. Project-specific, individual activities are included in the Miscellaneous category. These activities included traveling and sales efforts, for instance. Within the projects, the frequency of the miscellaneous activities was 30.8%.

#### 4 Analysis

This research provided valuable lessons for TietoEnator regarding WBS, but the results are extendable to the software industry in general. The main findings concerning the generation of WBS are reported in the following.

A small, yet sufficient, number of registration entities (activities) promotes and ensures effort registration on correct entities in the work time registry system (Haapio and Ahonen 2006). If assumed that the case study's 26 projects would have employed the non-construction activity WBS proposed in this paper, the total number of the non-construction activity entities in these 26 projects would have decreased from 282 to 182 activities. In 15 projects the number of non-construction activity entities would have reduced down to seven. On the other hand, they would have increased up to seven in 10 projects. It is doubtful that these projects would not have involved the most of the seven non-construction activities. Hence, it is very likely that effort had in these cases been registered on other activities or left unregistered, which skews the effort data.

It was found out that many of the registration entity titles and descriptions had the same meaning, but they had been misspelled, or in some cases they were written in singular form and in the other cases in plural form. This implies that the use of registration entity (activity) templates is advantageous to ensure uniformity among the general project activities, which in turn promotes correct registration of effort on the entities and efficient effort collecting, monitoring, and post-mortem analysis.

The non-construction activities identified in this study correspond mostly with the activities included in the earlier proposed waterfall-based work breakdown structures. This might not be a coincidence, since the projects in the sample were based on a waterfall software engineering process. The existing work breakdown structures are well-suited for projects which are either product-oriented or process-oriented because the premise for developing these structures was either a software product to be constructed or a particular process the software is constructed in. The previously established product-oriented work breakdown structures include the ones developed for the original COCOMO model (Boehm 1981) and for the COCOMO II model (Boehm *et al.* 2000) for projects applying the waterfall project lifecycle process. For projects applying an iterative spiral project lifecycle process, the previously established process-oriented work breakdown structures include the one developed for COCOMO II model for spiral process projects (Boehm *et al.* 2000) and a WBS for RUP (Royce 1998).

The comparison between the non-construction activities identified in this paper and the activities in the existing work breakdown structures provided by the software engineering literature implies that some of the identified non-construction activities seem to be previously neglected (Table 1). A reason for this can be that the proposed work breakdown structures are either product-oriented or process-oriented, and not project-oriented. The neglected activities include especially project-related orientation. Also the project-related activities, i.e. project functions by the project group members, have been disregarded as the emphasis has been on the project manager's functions. Moreover, it is likely that 'Customer interface' activity is merely the project manager's interaction with the customer (e.g. reporting) and does not include customer training, for instance.

**Table 1: Comparison of how the non-construction activities relate with four existing WBS.**

Non-Construction Activity	COCOMO (waterfall)	COCOMO II (waterfall)	COCOMO II (spiral)	RUP (spiral)

## Session 11: SPI and Management

Configuration Management	Configuration management and quality assurance CM/QA	System Engineering / Configuration management	Environment and Configuration & Management & Deployment	Environment & Deployment & (Assessment) & (Management)
Customer-Related Activities	Project office functions	Management / Customer interface	(Management)	-
Documentation	Manuals	Data / Manuals	(Management) & (Deployment)	Deployment & (Assessment)
Orientation	-	-	-	-
Project-Related Activities	(Project office functions)	(Management)	(Management)	(Management)
Quality Management	Configuration management and quality assurance CM/QA	System Engineering / Quality assurance	Assessment & (Management)	not a separate function

The activities, used in the waterfall work breakdown structure for the original COCOMO model comprise activities that correspond with the non-construction activities presented in this paper. These activities comprise project office functions, configuration management and quality assurance, and manuals. However, important activities such as orientation were excluded from the WBS for the original COCOMO model (Boehm 1981). According to the results of this study, orientation can be considered as a non-construction activity which is strongly related to software projects, and thus should be included to the employed WBS.

A criterion for a good WBS includes that it encapsulates change and evolve with the appropriate level of detail (Royce 1998). The generated WBS for the general project activities meets this: it is sufficiently detailed but open to include the majority of new common project functions into it.

## 5 Discussion

This study complements the existing research on software process improvement and capability maturity models in particular by focusing on a special case of effort management improvement in software projects, especially the case of generating a WBS.

Although there has been a vast interest in the concepts of software process improvement and effort in software projects, researchers have, by and large, overlooked introducing practical sufficiently-detailed and -described work breakdown structures for the general software project activities, and ways to create one. While the identified general project activities presented in this paper are grounded to the particular case and projects at TietoEnator Telecom & Media, they and the basic elements of the method can be generalized to other cases of generating a WBS within the software industry. The practitioners, the project or quality managers, can apply the results of this research to derive an organizational plan to generate a WBS, which this study gives a good input.

From the research point of view, the comparison between the four work breakdown structures provided by the software engineering literature (Boehm 1981, Royce 1998, Boehm *et al.* 2000) and the non-construction activities identified in this study is somewhat difficult since the definitions of the WBS activities in the literature are vague. This, however, has a practical implication. The undefined activities in the employed WBS result in assumptions on the activity. These assumptions can be either right or wrong. Hence, it is essential to open up and define the activities in the WBS guidelines for common comprehension.

The Grounded Theory methodology provided invaluable tools for this study. The methodology, however, was applied rather loosely in order to achieve pragmatic results. For one, the presumptions, previous studies (Haapio 2004, Haapio and Ahonen 2006, Haapio and Eerola 2006), and software

engineering literature was let to influence on the categorization in this study deliberately. This adapted approach turned out to be both successful and practical, and therefore applicable in other similar studies in the software industry. This two-phase approach from literature to practice, where the categories are developed from the literature and attempted to ground them in reality seems practically possible but is against the true spirit of Grounded Theory (Goede and de Villiers 2003). Secondly, the software engineering literature pre-knowledge and previous research on the non-construction activities had another effect on the coding. For instance, the last coding phase, selective coding, is in Grounded Theory methodology a process of integrating and refining the theory (Goede and de Villiers 2003). In previous coding phases the different categories had been formed. These categories are only descriptions of data and not a theory. To form the theory, various categories need to be integrated. One of these categories becomes the central category that represents the main theme of the research. Here, however, the central category (non-construction activities) was pre-stated. The aim of this study was not to create a theory in the form of a new main activity category but to generate a structure for the pre-stated category. Thirdly, the axial coding is far more detailed in the true Grounded Theory methodology. For instance, the relationships between the different coded categories were examined only intuitively in this study. Also, the work time registry system provided semi-open coded data to start with, as the registration entities were named and structured within the projects, although the naming and structuring varied between the projects.

One concern relates to the saturation of the coding: how many projects is sufficient for reaching saturation? For these 26 projects the saturation was reached within the projects, i.e. the categorization for these 26 projects was successful. This was accomplished when the 'Miscellaneous' category was included. The 'Miscellaneous' category is, however, problematic considering the Grounded Theory, since it is a 'dump' category, and therefore the registration entities (activities) in that category are not linked to each other and their properties. In this study, however, it was decided that individual, project-specific activities are categorized into the 'Miscellaneous' group to keep the final amount of different categories (activities in the WBS) limited.

Another concern is that the study was limited to one organization. Therefore, some observations and data might be unique to the organization in question, and this must be considered in the generalization of the results. However, the generality of the subject, i.e. the WBS was generated for the general software project activities, and the heterogeneity of the projects in the data sample defend the assumption of that the results can be in fact generalized.

## **6 Conclusion and Future Research**

The generation of WBS of general software project activities was explored in this paper. In addition, this research supplements into the research of effort distribution by introducing a coherent set of project activities, namely non-construction activities, which is frequently ignored both in research and the effort management in software industry.

Further, research on non-construction activities effort is needed. A deeper investigation into non-construction activities and their optimal effort proportions is currently being carried out. Moreover, research is needed to show how the project activities divided into the three categories bring stability into effort proportions and thus better predictability to effort estimation.

## **7 Acknowledgements**

I thank Professor Anne Eerola (University of Kuopio) for her invaluable comments on the draft of this paper. I also thank the support of the KYTKY program (EU funded development program for research co-operation between the software industry and the University of Kuopio). My thanks also go to Minna Haapio, MA (University of Joensuu) for her linguistic advice.

### 8 Literature

- Agarwal R, Kumar M, Yogesh, Mallick S, Bharadwaj R, Anantwar D. 2001. Estimating software projects. *ACM SIGSOFT Software Engineering Notes* **26**(4): 60-67.
- Boehm B. 1981. *Software Engineering Economics*. Prentice Hall: Englewood Cliffs, NJ.
- Boehm B, Horowitz E, Madachy R, Reifer D, Clark B, Steece B, Brown A, Chulani S, Abts C. 2000. *Software Cost Estimation with COCOMO II*. Prentice Hall: Upper Saddle River, NJ.
- Glaser B, Strauss A. 1967. *Discovery of Grounded Theory: Strategies for Qualitative Research*. Aldine Transaction: Chicago, IL.
- Goede R, de Villiers C. 2003. The applicability of Grounded Theory as research methodology in studies on the use of methodologies in IS practices. In *Proceedings of the 2003 annual research conference of the South African institute of computer scientists and information technologists on Enablement through technology (SAICSIT'03)*: 208-217.
- Haapio T. 2004. The effects of non-construction activities on effort estimation. In *Proceedings of the 27th Information Systems Research in Scandinavia (IRIS'27)*, Falkenberg, Sweden.
- Haapio T, Ahonen J. 2006. A case study on the success of introducing general non-construction activities for project management and planning improvement. In *Proceedings of the 7th International Conference on Product Focused Software Process Improvement (PROFES2006)*, Amsterdam, Netherlands, Springer Lecture Notes in Computer Science Series 4034, Springer-Verlag: 151-165.
- Haapio T, Eerola A. 2006. Post-project effort analysis method. In *Proceedings of the 1st Iberic Conference on Information Systems and Technologies (CISTI2006)*, Esposende, Portugal, Vol. I (Organizational Models and Information Systems), Microsoft: 3-19.
- Hansen B, Kautz K. 2005. Grounded Theory applied - studying information systems development methodologies in practice. In *Proceedings of the 38th Hawaii International Conference on System Sciences (HICSS-38'05)*, Hawaii, USA, IEEE Computer Society: 1-10.
- ISO. 1993. ISO/IEC TR2 15504, Part 1 - Part 9, Information Technology - Software Process Assessment. ISO.
- Järvinen P. 2001. *On Research Methods*. Opinpajan Kirja: Tampere, Finland.
- MacDonell S, Shepperd M. 2003. Using prior-phase effort records for re-estimation during software projects. In *Proceedings of the Ninth International Software Metrics Symposium (METRICS'03)*, IEEE Computer Society: 1-13.
- Paulk M, Curtis B, Chrissis M, Weber C. 1993. Capability Maturity Model for Software, Version 1.1. Technical Report, CMU/SEI-93-TR-024, ESC-TR-93-177, CMU/SEI.
- Pressman R. 2005. *Software Engineering: A Practioner's Approach*. 6th edn. McGraw-Hill: New York.
- Royce W. 1998. *Software Project Management: A Unified Framework*. Addison-Wesley: Reading, MA.
- SEI. 2002a. Capability Maturity Model Integration (CMMI), Version 1.1, Continuous Representation. Technical Report CMU/SEI-2002-TR-028, ESC-TR-2002-028, CMU/SEI.
- SEI. 2002b. Capability Maturity Model Integration (CMMI), Version 1.1, Staged Representation. Technical Report CMU/SEI-2002-TE-029, ESC-TR-2002-029, CMU/SEI.
- Sommerville I. 2001. *Software Engineering*. 6th edn. Pearson Education: Harlow, UK.
- Wilson D, Sifer M. 1988. Structured planning-project views. *Software Engineering Journal* **3**: 134-140.
- Wolverton R. 1974. The cost of developing large-scale software. *IEEE Transactions on Computers* **23**: 615-636.

### 9 Author CVs

#### **Topi Haapio**

Topi Haapio is a project manager at TietoEnator Telecom & Media Ltd. and a postgraduate at the University of Kuopio, Finland. He received his BS and MS degrees in 1997 and 1998, respectively, both in computer science. Currently, he is writing a dissertation on the effort management in software projects, and his main research interests include software project effort estimation and cost modeling, software process improvement, quality, and project management. His published research includes conference proceedings in IRIS (Information Systems Research in Scandinavia), PROFES (Product Focused Software Process Improvement), and CISTI (Iberic Conference on Information Systems and Technologies).





# A Risk Management Process

*J. Horstkötter, J. Fleckner<sup>1</sup>*

## Abstract

Risk management is an often ignored, but valuable process to keep the project on track and reduce the costs. Projects only run into problems unexpectedly, if you ignored risks and these get reality.

The following lecture describes how risks can be rated and how they can be tracked during the project lifetime.

Tables for rating a risk which can be used for both, a rough (qualitative) and a quantitative rating are presented. You can gain experience with the qualitative rating to come up with more and more quantitative ratings in the next projects.

The rating table ensures that the risk priorities are consistent, i.e. risks of the same risk height cost the same amount of money.

It should be checked regularly if the defined measures are executed or new risks can be identified. Then a new rating of the risks can be done. The risk ratings over the time are condensed into a metric which shows how the project is able to control the risks. The metric can visualise the risk status for the management in form of traffic lights.

The process described is based on the experience made with risk management processes in several projects in various companies, it includes enhancements made from bad experience in practice.

## Keywords

Risk Management Process, risk metric, quantitative rates of risks, qualitative rates of risks, risk tracking, Risk Process Grading

---

<sup>1</sup> Fleckner und Simon Informationstechnik GmbH  
Im großen Rohr 8  
65549 Limburg  
06431 - 40 90 110 (Zentrale)  
[Josef.Horstkoetter@flecsim.de](mailto:Josef.Horstkoetter@flecsim.de)  
[Joachim.Fleckner@flecsim.de](mailto:Joachim.Fleckner@flecsim.de)

### 1 Introduction

#### Definition:

- A risk is a possible future event with negative impacts if it arises.
- If a risk arises it will be called a problem.

“How big is the reserve to solve problems in your project? Isn't that too high?”

“What is the biggest risk in your project? How big is it? How can you get rid of it?”

“Are you sure you have all project risks under control?”

These are questions most project leaders cannot answer, but an answer is possible and it is no magic to come up with qualitative or even quantitative answer. We want to present you a procedure to rate your risks and to track them.

Measures to reduce risks are executed in each SW project anyway. Finally, only the code has to be produced. Everything else done during development are measures to reduce the risk, that the code does not meet the expectations of the customer. But often the measures are only executed as a result of bad experience (=problems) in earlier projects, not because the risks have been analyzed and the measures found suitable to reduce current risks.

E.g. tests are executed independent of the probability to find an error and the typical costs of correcting this error if it was found later (not every correction of an error found by the customer is really expensive).

That's why a risk management process is needed which identifies and rates the risks. Subsequently measures can be chosen depending on the risk height; and the height of the remaining risks can be calculated. Reserves are considered in the project plan, risks can be taken intentionally.

Often no risk management is done because of lack of time, the expectation that all risks are under control or risk management didn't work in former projects.

Bad experience often results from risk management executed wrongly:

- The biggest risks are ignored, because nobody knows how to reduce them
- The rating of the risks is only done unsystematically and roughly, and it is never actualized
- No measures are defined or the defined measures are not executed because of lack of time
- The management is not interested in risks, especially those risks they have to manage.

But risks do not disappear, if they are ignored, they are always there. Problems do not appear from nowhere, known change into problems. In other words: The project is late or exceeds budget, because you ignored some risks, not because an unavoidable event occurred.

The risk management is in the responsibility of the project manager or even the purchaser, it can only partly be delegated:

**The responsibility for risk management falls to that party which has to pay the price for ignored risks.**

T. DeMarco, T. Lister

## 2 Rating risks

Most risks are known in the project long before they become a problem.

How these risks can be identified is a lecture of its own. For the purpose of this talk let us skip this question and assume we have identified them all.

After the risks are identified a rough qualitative rating of the risks is done.

But a rough rating is only good enough to prioritise the risks, not to plan a reserve for your project. Your aim should be to gain experience and to get a quantitative rating of the risks in future time.

With a quantitative statement about the risk heights you can decide if the cost for the measures pay off. Also a new rating, after measures are taken or some time is gone, is easier.

But a quantitative rating of the risks is complex and needs a lot of experience. Though a rough rating should be done during the risk workshop and can be used as a basis for quantitative ratings done by experts.

### 2.1 Table for rough rating the probability and severity

**Definition:**

- The probability of occurrence (PO) is the percentage, how many risks of the same kind will change into a problem.
- The severity is the disprofit caused by a problem (the amount of money necessary to solve it).

As done in an FMEA the risks are rated by probability of occurrence (PO) and severity (S). Both are used to calculate the priority of the risk and the risk height. For the detection value of the FMEA there is no equivalent, only in some cases you can settle the start of measures at early indicators.

For a rough rating there should be a choice out of four or five different values. Practice has shown, that a rating with three values is too rough. With only three values you are not able to calculate a adequate priority of the risks.

For the probability of occurrence following table has been proven suitable:

No. PO	% rough PO	Description PO
1	0,50%	unlikely
2	5,00%	possible
3	12,50%	probable
4	50,00%	pretty certain

**Table 1: values for a rough rating of PO**

In practice the highest probability is defined for a range from 80 to 100. But then you speak about a problem, not about a risk. Even a probability of 50% is only acceptable, if you have a small severity. Otherwise you should inform your management and start measures at once.

For the severity the following equivalent table can be used:

No. S	Description S
1	small
2	middle
3	high
4	Extreme high

**Table 2: values for a rough rating of severity**

## 2.2 Additional hints how to rate the severity

The severity can be related to three classes:

- costs
- time
- other quality criteria like functionality, absence of errors or maintainability.

If all three severity classes are rated, the maximum value of these severities is chosen.

The aim is to come up with percentage values comparable to the values for the probability of occurrence.

### 2.2.1 Costs

The costs can be rated easily on a quantitative scale. Just relate the costs caused by the problem with

- the costs of each produced piece
- the overall costs of the project
- the engineering costs of the project

You come up with a percentage number, comparable to the probability of occurrence. Compare the damage you get if the risk occurs with the calculated costs, you come up with a severity level as shown in the following table.

E.g. the development costs of a SW project are 775.000 €. To solve a problem would cost 14.000 €. This leads to a severity of 1 (1,5 %).

No.	Description S	%	Cost per piece	Overall or engineering cost
			18,50 €	775.000,00 €
1	Small	1,50%	0,28 €	11.625,00 €
2	Middle	12,50%	2,31 €	96.875,00 €
3	High	50,00%	9,25 €	387500,00 €
4	Extreme high	200,00%	37,00 €	1.550.000,00 €

Table 3: Identifying the risk severity based on the costs

### 2.2.2 Time

A quantitative rating is more difficult, if the problem takes time. As long as the additional time is only an additional effort for the project, it can be calculated like the costs. You just have to distinguish, whether the complete project is shifted (all developer work longer for the project) or only some developer do. In the first case you have to rate the extra time to the complete project time, in the second case you have to rate the extra effort to the complete project effort. You can also calculate the amount of money needed for the additional efforts and then use the table used for costs.

No.	Description S	%	240 days	1512 person days
1	Small	1,50%	3,6	23
2	Middle	12,50%	30,0	189
3	High	50,00%	120,0	756
4	Extreme high	200,00%	480,0	3024

Table 4: Identifying the risk severity based on the time delay.

But how to rate a problem, if customer milestones cannot be met? You can try to estimate the costs which result from actions to correct the problem like

- Dissatisfied customer = marketing efforts
- Reputation damage of your company = marketing efforts
- Recalls and guarantee cases = costs for repair + marketing efforts
- Reflashing after delivery = flashing cost per item
- Customer Queries = demands of the helpdesks

If you can quantify the costs resulting from the time delay then use the cost table to rate the risks.

If you cannot calculate the time delay as a percentage of costs or development time you have to use other quality criterias to define the severity. E.g. the following table shows the rating if a milestone is deferred. Four values won't help for such a rating, that's why we exceed the table to 10 values as used in the FMEA. The values for the delayed weeks and the milestone have to be combined.

Milestone	Delay				
	< 2 weeks	< 4 weeks	< 6 weeks	< 8 weeks	>= 8 weeks
uncritical milestone	4	5	6	7	8
critical milestone	6	7	8	9	10
highly critical milestone	8	9	10	10	10
essential milestone	10	10	10	10	10

Table 5: Identifying the risk severity based on the missed milestone.

### 2.2.3 Other quality criteria

For other quality criteria the same method as described for a deferred milestone can be used. If no quantitative rating is possible, a qualitative rating depending on the effect of the problem to the customer has to be used.

As an example the following table shows severities from the automotive industry.

No.	Definition of Severity used for an FMEA from VDA Schrift 4, Teil 2, 1. Auflage 1996
1	Very low: very low functional impairment, can only be detected by experts
2	Lower boundary of 3
3	Low: low functional impairment of the car, correction on the next garage date, functional impairment of service and comfort systems.
4	Lower boundary of 5
5	Middle: functionality of car reduced, go to garage as soon as possible, functional impairment of important service and comfort systems
6	Upper boundary of 5
7	High: functionality of car strongly reduced, go to garage immediately, functional impairment of important system parts.
8	Same as 7, but the error is hidden, it is not obvious
9	Very high: security risk, legal requirements not fulfilled, car may get stuck
10	Same as 9, but the error is hidden, it is not obvious

Table 6: Identifying the risk severity based on the error effect

### 2.3 Common table for a quantitative rating

For a quantitative rating four values are not sufficient, but 10 values as used in the FMEA work fine.

It is not wise to use a linear scale for PO or severity, because then you cannot differentiate the real big risks from smaller ones, or all small risks have the same value, while is not true in reality.

We have found that a factor of two gives you a wide range for rating the probability of occurrence and the severity. One table can be used for both, probability of occurrence and severity. The tables can be used for a first rating of a risk and for tracking the risks. The detailed values can be used in the risk workshop to settle on one rate. The Min and Max values can be used to agree to one rate if you calculate the average of the percentage in the risk workshop. Also during tracking the risk you might calculate percentage values which you want to assign to one level.

The severity starts with four times the values of the PO, because it can exceed 100% of the related costs<sup>2</sup>, the probability of cause can not.

The table can be used for both, a rough and a quantitative rating. You can gain experience with the qualitative rating and in the long run with more and more quantitative ratings in the next projects.

No. PO (rough)	Description PO	No. S (rough)	Description severity	% rough	% detailed	Min	Max
1					0,10%	0,00%	0,15%
2					0,20%	0,15%	0,30%
3 (1)	unlikely	1		0,50%	0,40%	0,30%	0,60%
4		2			0,80%	0,60%	1,20%
5		3 (1)	small	1,50%	1,60%	1,20%	2,40%
6 (2)	possible	4		5,00%	3,20%	2,40%	4,80%
7		5			6,40%	4,80%	9,60%
8 (3)	probable	6 (2)	middle	12,50%	12,80%	9,60%	19,20%
9		7			25,60%	19,20%	38,40%
10 (4)	pretty certain	8 (3)	high	50,00%	51,20%	38,40%	76,80%
		9			102,40%	76,80%	153,60%
		10 (4)	Extreme high	200,00%	204,80%	153,60%	307,20%

Table 7: Common table for rating risks

### 2.4 Risk height and Prioritisation

**Definition:**

- The risk height<sup>3</sup> is a percentage number. The size of the number represents the average effort to solve the problems raised by these risks.
- The risk priority is a subsumption of risk heights into classes. All Risk heights of one class are handled the same way.
- The statistical costs is the risk height times the base used to calculate the cost severity.

In practice often a table is used for the prioritisation of a risk, where high priorities are only given for projects with high probability and high severity. But that's not sufficient. If you have an "unlikely" prob-

<sup>2</sup> I've worked in a project where the severity of a problem exceeded the development costs several times

<sup>3</sup> Often "risk" is defined identical to our definition "risk height". We use the term "risk height" to emphasise the value represented by the risk.

ability and an “extreme high” severity, the risk can even be higher than for a “probable” risk with “high” severity.

E.g. in the automotive business, the probability of exceeding the RAM or ROM of the microprocessor may be unlikely, but the severity, having to deliver a larger microprocessor for the same price, might be rather high (several times the SW development costs). Would you call that a medium risk?

You have to use a risk height representing the statistical costs of the risk. It is defined by the product of the probability of occurrence (%) and the severity (%) of the risk.

Using the common rating table ensures that the risk heights are consistent. If the probability of occurrence or the severity of the risk is reduced by n%, both lead to the same new priority.

Same risk height values are assigned to risks with the same statistical cost. E.g. you get the risk height 0,10 % from a risk with low PO (0,40 %) and a high severity (25,60 %) or with a high PO (12,80 %) and a low severity (0,80%).

The risk height shows the cost which will typically be necessary to solve the problems, if they arise . You can call it also statistical cost. The risk height also shows the maximum amount of money which should be spent to reduce or eliminate the risk. Normally the cost for measures should be much less then the risk height.

We’ve transformed the common table for rating risks into a matrix. The severity gives the columns of the matrix, the PO gives the lines. You can use the matrix to identify the risk height if you have only qualitative values (even if this does not represent a real percentage of costs).

To get an overview over all risks and for communication with the management it is helpful to divide the risk height into priority groups. Typically three groups, represented by traffic light colours are used:

- Priority 1, critical (red, white in the graphic):  
for these risks measures have to be taken to reduce them  
risk height  $\geq 0,20\%$
- Priority 2, medium (yellow, grey in the graphic):  
these risks need at least attention, measures should be taken as far as possible.  
 $0,02\% < \text{risk height} \leq 0,20\%$
- Priority 3, negligible (green, hatched in the graphic):  
these risks can be accepted  
risk height is  $< 0,02\%$

		Severity →									
		1	2	3	4	5	6	7	8	9	10
		0,40%	0,80%	1,60%	3,20%	6,40%	12,80%	25,60%	51,20%	102,40%	204,80%
PO ↓	1	0,10%	0,00%	0,00%	0,00%	0,01%	0,01%	0,03%	0,05%	0,10%	0,20%
	2	0,20%	0,00%	0,00%	0,00%	0,01%	0,01%	0,03%	0,05%	0,10%	0,20%
	3	0,40%	0,00%	0,00%	0,01%	0,01%	0,03%	0,05%	0,10%	0,20%	0,41%
	4	0,80%	0,00%	0,01%	0,01%	0,03%	0,05%	0,10%	0,20%	0,41%	0,82%
	5	1,60%	0,01%	0,01%	0,03%	0,05%	0,10%	0,20%	0,41%	0,82%	1,64%
	6	3,20%	0,01%	0,03%	0,05%	0,10%	0,20%	0,41%	0,82%	1,64%	3,28%
	7	6,40%	0,03%	0,05%	0,10%	0,20%	0,41%	0,82%	1,64%	3,28%	6,55%
	8	12,80%	0,05%	0,10%	0,20%	0,41%	0,82%	1,64%	3,28%	6,55%	13,11%
	9	25,60%	0,10%	0,20%	0,41%	0,82%	1,64%	3,28%	6,55%	13,11%	26,21%
	10	51,20%	0,20%	0,41%	0,82%	1,64%	3,28%	6,55%	13,11%	26,21%	52,43%

**Table 8: Risk height as a function of probability of occurrence and severity, priority as groups of risk heights.**



### 3 *Considering risks in the project effort*

After the risks are rated and prioritized, measures to reduce the PO and/or the severity have to be defined beginning with the biggest risks. These measures have to be integrated into the project activities and increase the project effort.

The effort for each contingency plan times the probability of the risk has to be added as a reserve to your project effort.

For all risks that will not be eliminated after the measures have been executed, the remaining risk height has to be added to the project effort as a reserve, and statistical costs of this height will occur as problems and have to be paid for.

### 4 *Tracking the risks*

#### 4.1 *Checking the risk height*

To track the risks following actions have to be taken into account:

- 1) Check, that the defined measures are executed and effective
- 2) Check, whether new risks can be identified
- 3) Check, whether a risk changes into a problem and start the contingency plan
- 4) Do a new rating of the risks

##### 4.1.1 *Check the execution of measures*

As done for other project activities regularly checked that the defined measures to reduce the risks are executed. Also prove that the measures are effective and the risk is reduced as expected. After the measures are completed, some risks are settled (value 0), others remain with a small risk height until the project ends.

##### 4.1.2 *Check for new risks*

Regularly check if new risks can be identified. This can be done in regular meeting which are executed anyway or in new risk workshops at the end of a development cycle. The new risks can be divided into two types:

- The risk existed at the beginning of the project, it has been overseen, forgotten, etc.
- A new risk came up after an event

This distinction is necessary, because both types are handled separately.

- Was a risk only overseen, it is added to the list as if it had been there from the beginning. Both, the risk height and the target curve in the risk metric increase.
- The real new risks are tracked after they had been identified, only the risk height changes, not the target curve.

##### 4.1.3 *Check for mitigating risks*

For each risk check, if it occurred or will occur in the near future. If so the defined contingency plan

has to be executed.

The rating of an occurred risk (a problem) depends on the height of its effect. The probability of occurrence is defined to one, the effect is defined to the real effect (first it is estimated, then the progress can be measured monthly). This value keeps constant during the rest of the project.

### 4.1.4 New risk rating

For all risks a re-evaluation of a risk is executed regularly. Often the probability of occurrence or the potential effects are reduced over time. Sometimes it can also be constant until a milestone is reached and then it ends. Of course both values are reduced by respective measures. For each risk the values should be recalculated regularly, the result gives the actual risk height of the project.

Example:

- The probability that an important team member drops out decreases linear with the time spent.
- The probability that a part from another supplier is not delivered in time may be considered constant, until the part is really delivered.

In addition for each risk the severity and the probability of occurrence should be proven (are the current values still valid). If the probability of occurrence increases during the project, this is an indicator that the risk starts to change into a problem.

If measures are missing for a critical risks or a medium risk with rising probability, the risk height of this risk should be doubled.

After a development cycle is finished, a new risk workshop should check if new risks can be identified and do a reevaluation of the known risks.

If the measure is a contingency plan the risk is rated in the following way:

- the probability of occurrence is calculated as described before.
- the severity is calculated as if the consistency plan had been performed, this means that you use the severity you expect after the occurrence of the risk and the execution of the contingency plan.

## 4.2 The risk metric

To identify and rate the risks helps. Tracking the risks and executing the measures is even better. But do you reach your target? Is the risk management process in plan? These questions can be answered easily if the tracked risks are visualised in a metric showing the risk height over the project time.

The risk height of each risk is calculated as described in chapter 2.4. The risk height of the project is the sum of the risk heights for all individual risks. For tracking the risks the risk height of the first risk workshop is standardized to 100%<sup>4</sup>.

At the end of the project all risks will be eliminated and some efforts have been done for occurred risks (problems), so a target line can be drawn from the date of the risk collection, starting at 100%, to the date of the project end, ending at the height of the remaining risk (the planned reserve of your project).

The sum of the risks heights measured at any date should not exceed this target line. Typically two lines are drawn to grade the risk process for the management in form of a traffic light. If the risk height exceeds the desired curve for more than 5%, a yellow traffic light is set, if it exceed the line for more than 20%, a red traffic light is set. The traffic light signals the management missing progress in reducing the risks.

---

<sup>4</sup> As long as you are not able to come up with accurate probability and severity values, the risk height of projects cannot be compared. Coming up with a higher number doesn't mean that one project has more risks than another. The quality criterion to be checked is that measures are executed to reduce the identified risks. That's why the risk height is normalised to 100%.

The following metric shows the values of one of the project I've advised<sup>5</sup>.

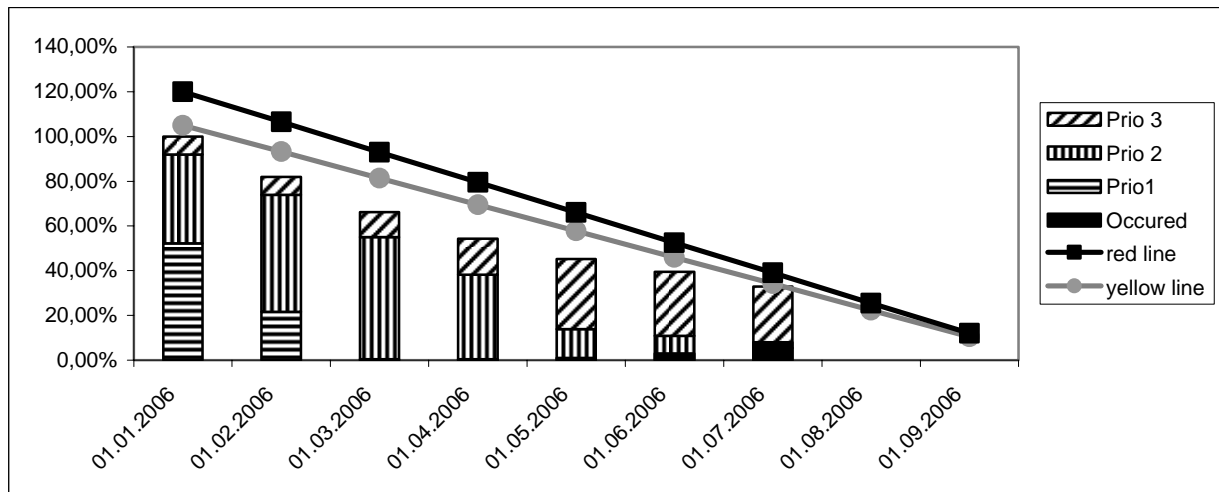


Figure 1: Diagram of the risk metrik

## 5 Lessons Learned

At the end of any project a lessons learned workshop should be executed. One part of the workshop should be the verification of the risk management process to enhance experience.

- Have risks occurred, which are not listed in the risk checklist?
- Have measures been defined and executed?
- Have all risks been identified before they occurred?
- Have the risks occurred in average with the estimated probability?
- Had the occurred risks in average the estimated severity?
- Did the measures reduce the probability of occurrence and the severity as estimated?
- Did the contingency plan work?

## 6 Conclusion

Remember the following rules if you have to handle risks:

- If you rate probability and severity, use at least 4 levels. Gather experience and try to come up with more precise values.
- Risks of the same priority should have the same statistical costs, independent of probability and severity. Reducing the probability for 20% should lead to the same new priority than reducing the severity for 20%.
- Collecting risks at the beginning of the project is not sufficient, they have to be tracked regularly.
- Use a metric to visualise the current risk height and compare it with you targets.

<sup>5</sup> The values sampled during project time have been recalculated to fit the described process tables. The "occurred" value represents the risk height when the problem rose up. The dates have been changed to make the project anonymous.

I started this talk with some questions. If you use this procedure, you can answer them now:

*“How large is the reserve to solve problems in your project? Isn't that too high?”*

Using the process you can calculate the necessary reserve from the risk height. If somebody says, it is too high, you can discuss the rating of the risks.

*“What is the largest risk in your project? How large is it? How do you want to get rid of it?”*

The periodic rating of the risks always shows you the current largest risk. And you can quote the risk. Ok, how to get rid of it isn't handled here (was discarded, because the lecture became too large).

*“Are you sure you have all project risks under control?”*

Following this process all identified risks are always under control. And I've not been in any project, where the risk management process worked and not all relevant risks have been identified.

An excel sheet as a template for the risk management process can be downloaded from our homepage [www.flecsim.de](http://www.flecsim.de). There you find also a complete description of the risk management process including risk workshop techniques and how to define and plan measures.

### **7 Literature**

T. DeMarco, T. Lister: Bärenango Carl Hanser Verlag Münche Wien, 2003 (Original: Waltzing with Bears: Managing Risk on Software Projects, Dorset House Publishing Co., Inc., 2003)

En.Wikipedia.org: Risk Management

PMI: Project Management Body of Knowledge Project Management Institute, Newtown Square

Georg E. Thaller: Drachentöter, Verlag Heinz Heise

Gerhard Versteegen: Risikomanagement in IT-Projekten, Springer Verlag

Ernst Wallmüller: Risikomanagement für IT- und Software-Projekte, Hanser Verlag

Tony Moynihan: Coping with Is/It Risk Management, Springer Verlag

Markus Gaulke: Risikomanagement bei Softwareentwicklung und -einführung, KES Nr4, 2001

Dr. R.F. Erben, F. Romeike: Risk-Management-Informationssysteme

## 8 Author CVs

### Josef Horstkötter

- Study of electrical engineering, emphasis on data technology in Paderborn,
- 17 years of experience in the field of SW development
- 14 years of experience in software engineering and quality assurance, with emphasis on testing
- 7 years of experience in project management support
- Certified SPICE-Assessor
- Last projects
  - Performed software quality assurance for an automotive supplier with direct project support and coaching of developers
  - Performed management consulting of project management in an automotive supplier project
  - Developed a SPICE conformant SW development manual and implemented it in a pilot project
  - Participated in a project supporting and controlling team at an automotive manufacturer

### Dr. Joachim Fleckner

- Study of Physics, doctorate in 1979 in Theoretical Nuclear Physics
- 35 years of experience in software development
- 25 years of finding solutions in software engineering
- Business manager since 1/1/2000
- Certified "Trained Bootstrap und SPICE-Assessor" 2001
- iNTACS Competent ISO / IEC 15504 Assessor
- Significant projects:
  - Embedded Systems: Development of drivers for a proprietary operating system (79 - 81)
  - SW Design: Participation in designing a control center for power supply companies (81 - 85)
  - SW Processes: Project consultant for the development of controls for a CNC universal machine tool (86 - 92)
  - SW QS: Quality assurance (QA) for software in an "air space surveillance system" project (84 - 92) and in a telecommunications network node project (91 - 96)
  - Automotive: QA in several projects for the development of multi-purpose instruments, radios, navigation systems and devices in the car body sector (96 to present); preparation of development manuals, execution of training measures
  - Improvements: Preparation and presentation of projects in SPICE assessments by manufacturers in the automobile industry
  - Improvements: Introduction of SPICE-compliant development processes for suppliers



# Adoption of Agile Development in Practice: a Qualitative Inquiry

*Inga Lagerquist, Margareta Lindmark, Janis Stirna, Jaana Nyfjord*

## **Abstract**

A qualitative study about how agile information system (IS) development methods are used in practice was conducted. We addressed the motivation to choose agile methods, their contribution to IS development process in terms of support for managing changes and avoidance of unnecessary functionality, improved communication within the team and the customer, tailoring of the agile methods, as well as the use the agile best practices. We also analysed the characteristics of individuals working in agile development teams.

## **Keywords**

Agile methods, adoption, qualitative study



### 1 Introduction

Agile software development methods such as eXtreme Programming (XP) [3], Scrum [17], Agile Modeling (AM) [2], and Dynamic System Development Method (DSDM) [18] emerged as a reaction against the ineffectiveness of plan-driven information system (IS) development approaches such as Rational Unified Process (RUP) [9] and numerous in-house approaches based on some theoretical IS development model, e.g. the waterfall model [16]. The emergence of the agile methods was a response to the opinions that the plan-driven methods are too complex and unnecessarily bureaucratic, because they tend to plan the most of the system development process to a great level of detail thus slowing down the project. The plan-driven approaches also seemed less suited for the turbulent business environment of these days when customers want to change the requirements as soon as their business encounters new challenges. This created the need to make the IS development processes more effective in addressing the customers' higher demands on flexibility and individual treatment. As a result, agile approaches to IS development emerged.

Agile methods have been tested and applied in practice for a considerable time now (see e.g. [1], [10], [19]). The purpose of this study is to investigate how they are used. More specifically, this paper addresses the following research questions: what motivates the choice of agile methods, how do agile methods contribute to the product development process, how do agile methods help in managing changes, how do companies deal with the "good enough" principle, what is the effect of agile methods on improved communication, how do companies tailor the agile methods, how do they use the agile best practices, and what are the characteristics of individuals working in agile development teams?

The reason for choosing a qualitative research approach is that most of the knowledge concerning the practical application of agile methods lies in the heads of the practitioners in the form of tacit knowledge such as beliefs, experiences and opinions. We used Grounded Theory [8] to capture and analyse this knowledge. In the course of our research we interviewed 17 experts in agile methods from 13 firms. Due to limited space this paper provides an overview of the findings; the complete study is available in [11].

The rest of the paper is organised as follows. Chapter 2 provides a brief background to agile methods. Chapter 3 presents a summary of the findings of the study, which is followed by concluding remarks in Chapter 4.

### 2 Background to Agile Development

Agile methods such as Agile Modeling [2], eXtreme Programming [3], Crystal Clear [6], Scrum [17], and DSDM [18] all share the same values, e.g. to be flexible and to be able to deal with changes during the course of a software project. The highest goal of the agile methods is to deliver value to the customer early and continuously. Based on these assumptions the agile methods then provide a set of principles, techniques and best practices for IS development. Some of them focus on providing support for different phases of the IS development and some of them can easily be combined with other methods, techniques and practices to further support IS development (see e.g. [1]).

Agile approaches suggest delivering the solutions in smaller parts on a frequent basis before the whole product is delivered. These small deliveries allows for early customer feedback [18]. Because changes are generally unavoidable the agile methods require close cooperation with the customer to determine which features are going to add the most business value to the product at the end of the next iteration [4].

According to [4] an agile development method must have the following characteristics – (1) iterative (several cycles), (2) gradually growing (not to deliver the whole product at once), (3) self-organizing (the team in cooperation with the customer decides and prioritises the tasks and organizes themselves along with the decisions taken), and (4) adaptive (processes, principles, work structures are adapted as needed when identified during the project instead of being fixed early or before the need is identified).

In the following a brief overview of a few agile methods is presented.

Agile Modeling is an approach that gives practical advice and best practices of how to perform effective modelling and documentation in projects that use agile methods. According to [2] the two most important reasons for modelling are (1) to understand what is going to be developed, and (2) to communicate with the customer and within the team. The general aim is to keep the amount of models as low as possible, to keep them simple as well as to work iteratively and incrementally.

eXtreme Programming [3] is a collection of well known development best practices, such as short iterations, pair programming, test-driven development, refactoring, continuous integration, close customer participation during the whole development project, and rapid feedback, etc. The core values of XP are communication, simplicity, feedback, courage, and respect which clearly states the purpose and the goal of each activity in XP projects.

Scrum [17] is a method for managing software development projects. Projects are divided in short iterations (called "sprints") to allow for feedback and flexibility. The developers are encouraged to choose the techniques and tools that suit the team in the development process in order to achieve the expected results. The team synchronizes their job every day in a stand-up meeting lasting 15 minutes. Every team member answers three questions – *What did you do since last meeting, what will you do before next meeting, and do you have any obstacles?* The answers help to control if the work is proceeding as scheduled.

Dynamic Systems Development Method [7] is a method developed by a consortium where all the members can participate in the further development and refinement of the method. The main idea of DSDM is to fix the project resources and the time. The requirements are then prioritised according to what can be done within the time and resource constraints. DSDM defines project participant roles that are set by the customer – the orderer, visionary, ambassador user, and reference user.

The Crystal family [6] consists of a number of different methods that depend on the team's size. Each method is named after a colour – Clear, Yellow, Orange, Red, etc. Crystal Clear is a method for small project groups including up to eight members which therefore needs less coordination than larger teams.

### 3 Agile Development in Practice

This chapter presents a summary of the research findings supported by citations from the interviews with 17 agile practitioners in Sweden. Each interviewee is represented by a number in the range of 1-17. Profiles of the quoted interviewees are presented in Section 5.1.

#### 3.1 Motivation for Choosing Agile Methods

The study shows that a general motivation for choosing agile methods is that companies are dissatisfied with the projects that has used plan-driven IS development methods. The major drawbacks were mentioned as (1) long time to start and to carry out the project, and (2) too much emphasis on planning and documentation instead of software development. In addition, these approaches were perceived as overly bureaucratic, complex and consuming resources for project management and therefore taking away the focus from development.

*"...then it was also a dislike to RUP and the difficulties with introducing the UP, you can say. A method that needs two years and five consultants to be introduced in an organisation, shows that there it is something crazy with it. And then it does not matter how much information it contains." [I3]*

As a result the companies were seeking for development methods that *"look like a framework"* and *"tell what should be done in the project"* but at the same time *"allow developers to decide the use of appropriate tools and the ability to steer the tasks on their own"*.

The companies that did not have a defined IS development approach prior to agile development had

two main goals – (1) to have a method in order to describe their way of working to the customers, and (2) a concrete description of the development process for the developers. These companies did not consider introducing a plan-driven method due to the perceived complexity of such methods.

The companies considered the ideas and principles defined in the Agile Manifesto [12] and the methods that were based on these principles combined with industry best practices easier to understand and practically appealing. They saw the agile methods as something that could give the benefits in practice, because these methods are created on the basis of the experiences and best practices of their authors.

### 3.2 More Effective Product Development Process

The companies reported that that the developers working with agile development could work more effectively and focus on the improvement of the product throughout the whole IS development process. Time boxing and short iterations (2-6 weeks) resulted in early product deliveries, which in turn helped the development team to see concrete results and which also raised the team's inspiration and work morale. This is aligned with Cockburn's [6] argument that the tested code works as an early victory for the project.

In contrast, some interviewees warned against being “*too effective*” and delivering product increments too often because the customer may not have the time to validate the functionality after each such iteration.

*“There can be a problem. Sometimes we are too ambitious and deliver a large amount and often – so much that the customer does not have time to process it. I have experienced it a couple of times. We have gone forward too quickly and the customer could not catch up with us and you discover afterwards that they had not tested everything that they should have done. They are not able to manage it. They need to take care of their business and then you realize how difficult it is [for the customer] to remove the people away from their daily jobs.” [17]*

In such cases the customer often becomes too stressed and feels reluctant to engage in the new projects. This situation could result in missing feedback from the customer. To address this problem [17] points out that the goal should be iterations that last 30 days.

The companies were positive to work closely with the customer and to involve the customer in the project in order to receive the immediate feedback and to handle the changes in the requirements. As a result, the customer felt involved and discovered that he/she could affect the product during the development process. The customer felt more engaged when concrete tasks were allocated to him/her.

The companies used prototypes to get feedback from the customer early in the development process. This allows the customer to build an opinion and to learn and to understand the real product. Some companies had tried to describe the product by using various kinds of documents prior to using agile approaches. This was less efficient because it often resulted in losing valuable feedback and a final product that did not meet the customers' requirements.

*“We tried many times before to ask people about their viewpoints before we started to build the product, but it was so imprecise. It was difficult to get any relevant viewpoints. Then we developed the product and [gave it to the customer], and then we got the first viewpoints, but then it was too late to make big changes.” [16]*

DSDM [18] recommends prototyping already in the early stages of the development process to show the customer that their requirements are correct. This shortens the development process and raises the confidence that the right solution is going to be delivered.

As an additional requirement for choosing agile methods the companies stated the necessity of the incremental deliveries as a way to keep the high speed of work. Keeping within the time and budget limits were regarded to be of high importance for building long-term relations with the customers.

### 3.3 Manage Changes

One of the situational factors that made the companies choose the agile way of working was the need to manage changes effectively. A common situation was that additional and more common requirements emerged during later stages of the project and sometimes after the introduction of the product on the market. The companies needed support for dealing with requirement changes. The agile way of working that they adopted did not require fixing requirements in the beginning of the project. Instead the customer was able to provide feedback during the whole development cycle, thus continuously improving the quality of the product. The cumulative effect of this process is that the customer learns about the system and its role of supporting the business progressively, as the system is being built.

Another effect is that the development group got a shorter project start when they did not have to finalise all requirements up front and to plan the project in detail. Instead the customers validate the correctness of the outcome of each iteration and then assign priority to the requirements. In the opinion of our interviewees such feedback resulted in higher quality of the product.

*“A big factor is quality. We get better quality when we use the agile approach as we can change much faster. If there is something wrong, we can change it. And also because we show the product to the customer much earlier, we can adapt it to the customer wishes, and there is no great amount of change requests which are never carried out due to the tight time frames or which you cannot afford. [The suggestion is] deliver early and then be prepared to change and to elaborate the product further.” [11]*

A drawback was also mentioned in this context. Some customers may often lack the knowledge about what it means to use the agile approach and how to deal with changes throughout the whole development cycle. Even if they sympathize with the five core values of agile development it is still easy to fall back to the plan-driven way of working.

*“The customer says that it is very good, but when we start working, it is the same as before – we need all the requirements before we can start working and they need to make all the decisions. But hello! Now you have contradicted yourselves. You follow the waterfall [model]. [11]*

In this context we recommend not to underestimate the time needed to change the culture at the customer’s company and to dedicate a conscious effort towards explaining the agile way of working to the customer.

### 3.4 Ability to Work Simply and the “Good Enough” Principle

The companies agreed that in general there is no need to develop more functionality than what the customer needs at the moment, which is the essence of the “good enough” principle. It is grounded in the assumption that only about 20% of all functionality of modern IS is actually used to create business value (see [15]). Similarly Beck [3] and Boehm and Turner [4] emphasize the value of simplicity in development. Design and writing the code for the future requirements is in fact spending resources on functionality that might never be needed. Cockburn [6] points out that, with requirements and functionality, it is easier to discover that something is missing rather than something that is not needed. According to the interviewees the project plan has to balance the time and cost constraints and *“not the best quality, but “good enough” quality”*.

The ability to work simply is a matter of attitude and the culture of the company, e.g. to hold simple, but effective stand up meetings, to make priorities in the requirement list, to create documents and models that have a clear purpose, regular automatic tests, etc.

While it sounds obvious not to develop unneeded functionality, our study shows that in practice it is difficult to follow this guideline. This is because inexperienced developers lack the concrete description of the processes. The customers have difficulties in knowing and in saying what is “good enough”. This challenge is particularly evident in large and complex IS development projects (see [13]). Our study shows that the developers experience fewer problems applying the “good enough” principle to documentation. All interviewees had opinions aligned with Beck [3], i.e. emphasizing that to work sim-

ply and according to the “good enough” principle does not mean to work half-heartedly and to develop a “so-so” product.

Moreover, because new knowledge and requirements may emerge at any stage of the development process, a “larger perspective” is needed to know what is going to be needed for the final product. To achieve this larger perspective, our recommendation is to use Enterprise Modelling in order to elicit strategic business goals and business processes and then link them to the IS requirements. In this way the team can clearly see what is the business value of a particular requirement and how does it contribute to the business vision. More about Enterprise Modelling and its application to IS requirements elicitation is available in [5] and [14].

### 3.5 Improved Communication

The agile approach demands close communication among the developers and with the customer. This was an important factor for the decision to adopt agile development. To improve communication the developers held daily stand-up meetings, the group was sitting in the same room, the active customer was often represented on-site, etc. A common information point was often used in the form of a whiteboard or a whole wall. The project was summarised in diagrams, user stories, schemas, etc. The objective of this approach is to visualize the project status and other relevant documentation and information in an open, communicative and easily accessible way.

The companies agreed that the stand up meetings that lasted 15 minutes gave a positive effect, e.g. a general view what is happening in the project on a daily basis and what are the current or emerging problems. These meetings also contributed to the team spirit and the overall work morale.

*“This is really good. It is so embarrassing to stand in front of the group and to say the same things again and again, and to admit that you still have not done the task already for several days. Then the people really want to get rid of the task.” [12]*

The interviewees admitted that in the beginning the developers were reluctant to change the meeting technique, e.g. to stand in the ring and to talk in front of the group. In addition, some of them also felt the performance anguish. For example, if they had promised to do the task until the next meeting, then they tried to complete it even if it required overtime.

The main objective is to make the customer an integral part of the development team. The interviewees gave examples of how to involve the customer in the project. The customer should (1) listen and to correct the misunderstandings concerning the business requirements, (2) supply the developers with up-to-date business information, (3) prioritise the requirement list, (4) promote the system to the end users, etc.

There is some danger (see e.g. [4]) that this strict demand to have the customer on-site may result in the customer company sending over the most replaceable employee. Such a person often is unmotivated and lacks the business knowledge required by the development team. Recently the companies and the customers are becoming aware of this.

*“... if the person does not understand that the active participation is important, then they have found the wrong person. The person who has the formal responsibility for the final product should not have anything more important than to steer the direction of the product development.” [13]*

### 3.6 Method Modifiability

The companies using agile development approaches make their own modifications depending on the situation, the project, the group distribution, the expected results, etc. Therefore the modifiability of the method also is an important factor influencing the choice of the method. The study showed that most often the companies introduced these methods in a pilot project, evaluated the pilot results, and then tailored and adopted the method throughout the company. In this regard, the interviewees emphasized the importance of not to follow the method descriptions blindly, but to understand the principles and

practices and then adopt them according to their specific needs. The quote below shows the need to tailor and combine various methods and method chunks.

*“The idea that there are only some best practices that everyone must follow starts to die away. Everyone needs to find what is that suits me. And of course, there is no sense to discover the bicycle again. The software industry starts to become sober and realize that you can not choose one method and then follow it [uncompromisingly], for example Scrum or RUP.” [18]*

### 3.7 The Use of Agile Best Practices

Most companies had not defined their own development method due to lack of time. When the agile methods became popular many companies discovered that they had used some of agile principles and values [12] in their work. The interviewees named examples of best practices such as stand-up meetings every day [17], collective ownership of code and pair programming [3] as useful practical advises, which were easy to adopt.

*“The ground in DSDM is much practical experiences from different types of companies and different types of projects.” [15]*

*“In my opinion it was a good approach, we could continue to work as we always did, but now to have more structure for doing our job.” [16]*

The interviewees recognised that one of the strengths with agile methods, which makes their adoption easy, is that they are based on “common sense” in IS development and that they give concrete, practical and hands-on advice.

*“And when we looked at this, it was quite difficult to use these agile methods. Nobody understood how this was going to work. We could not understand what was important, we read about it, it sounded interesting, but we did not started to use XP in practice. But via one project we learned about it and discovered that actually it was very easy to use. And it was XP we went for.” [14]*

The following problems concerning the adoption of best practices were encountered. For example, the stand-up meetings often did not work due to the fact that it was a new meeting technique. It was difficult to prioritise the requirements and some team members were unwilling to write the code in pairs. Such problems sometimes led to delays of the delivery.

Concerning the adoption of pair programming the interviewees reported that Beck’s [3] advice that one experienced developer works as a mentor for one inexperienced programmer might be too difficult to justify in terms of cost-benefit. The interviewees viewed the pair programming as very dependent on the individuals and on the project phase and context.

*“I love the idea [of pair programming]. But how can I report to the employer and to the customer. Yeah..., it was 100 hours extra, but we wanted that this guy should learn, we did not tell him what was right from the beginning. This is not realistic. In pair programming the real genius, as the principle say, they excel each other. They can save three days work.” [18]*

### 3.8 The Individuals

According to [2] and [6], plan-driven methods rely on a well defined development process and roles. The agile methods differ from this by emphasizing the importance of the individuals, their qualities, and the group dynamics. Our interviewees agreed that the personal qualities have the same value as the experience and the technical knowledge.

The companies have experienced that when using plan-driven approaches, a lot of support came from the methods, e.g. which kind of documents should be produced, which roles in the project need to be set. The common opinion was that agile methods tend to “go over to the other side” and to talk only about the people – “it is only people, no processes”. While admitting that the people are the most im-

portant resource in the project, the interviewees also expressed the need for stricter context-based method guidance.

The success of using agile methods depends to a large extent on the team's ability to cooperate and collaborate to carry out the suggested practices, e.g. pair programming. To have successful teams the interviewees suggest bringing together different types of individuals, including people that can and do dare to disagree or question common beliefs and who also support their ideas with facts and knowledge. The following personal qualities (highlighted in *italic*) are sought after when staffing an agile development project:

*Motivated* to learn and to share knowledge, because much of the project and product knowledge is implicit and not explicitly written down in the documents. *Cooperative* to find one's place in the team and to start working. If the team is unable to cooperate then the other important values such as communication and effectiveness are lost. *Sensitive listener* to be able to hear what the customers say and to sense what they do not say. *Willing to change* in order to try out new solutions and abandon them if they do not work. *Modest* to be able to ask for help. Individuals who are conceited or want to show-off by accomplishing a task alone have problems being effective in agile teams. This quality was paramount in the companies that used XP and collective ownership of code. *Communicative* to be able to communicate orally and not only via documents and e-mail. The team should work as much as possible in the same room. *Responsible* to take responsibility willingly for one's own work and also to share responsibility for the whole project. *Creative* to be able to find one's own solutions to various types of problems.

The experience is of great importance as the work is conducted in smaller and self-organizing teams. These types of teams also require another type of project manager, i.e. more of a coaching leader than controller and role-planner. The technical knowledge is essential in order to deliver expected results throughout the whole development process.

*"To work with these methods, you need to have some years of experience. As the project manager I want that my developers can tell me when they will complete the tasks. If you have not worked with some platform or language then it is difficult to estimate the time and you have "trial-and-error" all the time" [17]*

This is consistent with [4] and [13] stating that the agile methods require experienced developers because the methods rely on the implicit knowledge of the developers instead of documents.

#### **4 Concluding Remarks and Future Work**

This study shows that the application of agile methods to IS development gives results with business value. The positive effects include making the IS development process more customer-steered and code-oriented, avoiding development of unnecessary functionality, increased ability to deal with change, as well increased cooperation within the development team. Agile methods are used in a variety of projects and in a variety of contexts. The study also shows that using only one method (e.g. XP) uncompromisingly and to rely on it for all the necessary guidance in the practical work is generally not possible. Most likely it will have to be combined with other methods, techniques and practices. In the case of XP, a method for project management such as Scrum or DSDM can be an appropriate complement. Most companies also tailor their agile methods according to their situational and intentional contexts and the need for different techniques and practices that support development. The study also points out the need for having a certain number of experienced and technically skilled members in agile teams.

Future research should address issues concerning the applicability, adoption and adaptation of agile methods, including method tailoring and integration of methods into agile method alliances. Many of the tasks are dependent on the project and organisational context, which requires more research about the situational and contextual properties that influence the use of agile methods in practice.

## 5 Literature

1. Abrahamsson, P., Warsta, J., Siponen, M.T., Ronkainen, J., New Directions on Agile Methods: A Comparative Analysis, In Proceedings of Conference on Software Engineering, IEEE Computer Society, 2003
2. Ambler, S.W., Agile Modelling Effective Practices for Extreme Programming and the Unified Process, John Wiley & Sons, 2002
3. Beck, K. Extreme Programming Explained: Embrace Change, Addison-Wesley, 2004
4. Boehm, B., Turner, R., Balancing Agility and Discipline A Guide for the Perplexed, Addison-Wesley, 2004
5. Bubenko, J. A. j., Persson, A. and Stirna, J., EKD User Guide, Hypermedia and Pattern Based Knowledge Management for Smart Organisations, no. IST-2000-28401. Dept. of Computer and Systems Sciences, Royal Institute of Technology, Sweden, 2001
6. Cockburn, A., Crystal Clear: A Human-Powered Methodology for Small Teams, Addison-Wesley, 2005
7. DSDM Consortium, <http://www.dsdm.org/>
8. Glasser B.G., Strauss, A.L., The Discovery of Grounded Theory, Strategies for Qualitative Research, Aldine de Gruyter, 1967
9. Jacobson, I., Booch, G., Rumbaugh, J., The Unified Software Development Process Reading, MA: Addison-Wesley; ACM Press, 1999
10. Jönsson, M., Agile modeling in Sweden – from practices to principles. Master Thesis, Dept. of Computer and Systems Sciences, Stockholm University, 2004
11. Lagerquist I., Lindmark M., Agile Software Development Methods in Practice: A Qualitative Inquiry, in Swedish, MSc thesis, Dept. of Computer and Systems Sciences, Stockholm University, 2006
12. Manifesto for Agile Software Development, <http://agilemanifesto.org/>
13. Nerur, S., Mahapatra, R., Magalaraj, G., Challenges of Migration to Agile Methodologies, Communications of the ACM, May 2005, Vol.48, No5.
14. Persson, A. Stirna, J. An explorative study into the influence of business goals on the practical use of Enterprise Modelling methods and tools, In proc. of the 10<sup>th</sup> Int. Conf. on Information Systems Development (ISD2001), Kluwer, 2001
15. Lean Software Development, <http://www.poppendieck.com/>
16. Royce, W.W., Managing the Development of Large Software Systems: Concepts and Techniques. Proc. IEEE Westcon, 1970
17. Schwaber, K., Beedle, M., Agile Software Development with Scrum, Prentice Hall, 2002
18. Stapleton, J., DSDM Business Focused Development, Addison-Wesley, 2003
19. Svensson, H., Developing Support for Agile and Plan-Driven Methods, PhD thesis, Dept. of Computer and Systems Sciences, Royal Institute of Technology, Sweden, 2005

### 5.1 Profiles of Quoted Interviewees

- [I1] Two analysts/programmers, small software development company, 14 years of experience in software development and 5 years with the agile methodologies.
- [I2] Systems Developer/Programmer, small software development company, 9 years of experience in software development and 5 with the agile methodologies.
- [I3]. Two Developers/Programmers, medium sized international consulting company, approx. 11 years and 10 years of experience in software development and 2 years with the agile methodologies, and the last 4 months working only with agile approaches
- [I4] Systems Developer/Instructor, small IT consulting company, 6 years of experience in software development and 2 years with the agile methodologies.
- [I5] One Project Manager and one System Developer, small IT consulting company, both approx. 7 years of experience in software development with the agile methodologies.
- [I6] Project Manager, software development in a medium size production company, approx. 25 years of experience in software development and 1.5 years with the agile methodologies.
- [I7] Project Manager/Systems Developer, small IT consulting company, approx. 20 years of experience in software development and 10 years with the agile methodologies.
- [I8] Project Manager, medium size software development company, approx. 10 years of experience in software development and 3.5 years with the agile methodologies.





# Combining Agile Software Projects and Large-Scale Organizational Agility

*Petri Kettunen, Maarit Laanti*  
*Nokia Corporation*  
*P.O. Box 301, 00045 NOKIA GROUP, FINLAND*  
*petri.kettunen@nokia.com, maarit.laanti@nokia.com*

## **Abstract**

Many new product development (NPD) software projects apply nowadays agile methodologies. Their basic premise is that a small, co-located team working closely together with the business customer can produce high-value, high-quality software efficiently with rapid iterations and frequent feedback. However, in large NPD organizations developing complex products those basic assumptions are not necessarily sufficient, and a more comprehensive view of agility is needed. Based on industrial experiences, this paper presents a number of pragmatic suggestions about how and when agility could be utilized in large-scale (embedded) software product development. We propose a framework for understanding the multidimensional nature of agility in such environments for successful software process improvement (SPI), and highlight certain practical considerations. Some industrial case examples are illustrated. We conclude that in large-scale NPD organizations a holistic system-wide view is needed in order to really be able to take advantage of and improve software development agility.

## **Keywords**

Agile software methodologies; Software engineering management; Agile organization; New product development

### 1 Introduction

In the '90s, several new lightweight software process models such as XP, Scrum, and FDD emerged. As the software features have become the competitive edge of products, the ability to quickly implement and test the right features has become a key competence. In new product development (NPD) embedded software projects these agile methodologies are now widely taken into use [1].

Originally the agile methodologies evolved to fulfill the needs of small flexible software teams. Currently, as larger projects and organizations are getting more interested in applying them, there are several attempts to enlarge agile methodologies, such as Industrial XP and Scrum of Scrums [2, 3].

However, in large-scale product development projects, the original assumptions of the agile methodologies do not necessarily hold, and additional prerequisites and organizational conditions have to be satisfied in order to achieve the full benefits of agility. Successful software process improvement (SPI) in such NPD environments requires wider understanding of agile organizations and their enabling factors as well as the factors that prevent large companies from achieving agility.

In this paper, we examine these questions from a holistic point of view. The rest of the paper is organized as follows. Section 2 explores the background and related work, and sets the exact questions. Section 3 then proposes a framework for guiding SPI activities with respect to agility in large software product development organizations, and augments it with examples of practical considerations. Section 4 illustrates an industrial product development case. Section 5 discusses the propositions. Finally, Section 6 concludes with implications and pointers for further research.

### 2 Background and Related Work

#### 2.1 Agile Software Product Development

There is no one universal definition of agility. The concept was first applied in agile manufacturing [4]. The agile manufacturing literature gives many possible definitions, varying in scope and points of view; See for example [5, 6, 7, 8]. However, all definitions usually incorporate the basic concepts of speed and flexibility for responding to changes in dynamic market environments. The fact that the definitions of agility vary considerably indicates that the concept really is complex and multidimensional (i.e., not simply about responsiveness to changes). For the purposes of this investigation, we follow the definition by Conboy and Fitzgerald [5]. The customer/market interface is important [7]. The ultimate goal is profitable business [6].

In software development the term 'agility' was later adopted independently. Currently there is a range of publicly documented agile software process models (methodologies) available. While they all share certain common principles, there is no standard way of defining agile software processes. Consequently, it is not so straightforward to compare the different methodologies. We have addressed this question elsewhere [9]. One kind of agility method rating is proposed in [10].

Not all problems in software projects can be solved by software process methodologies [9, 11, 12]. Different agile software process models have different assumptions and prerequisites, depending on the project environment and characteristics. Because of these fundamentals, many companies have to first rethink their organization and management principles in order to be able to successfully adopt any one of those agile software methodologies [13]. Like for example Cockburn has demonstrated, different software projects have their speed-limiting factors on different areas [14]. The key issue is to match the software process with the true needs, and adopt changes to the process when needed. This, of course, cannot be done without thorough understanding of one's business, product and organization. The organization should support the agility. For example, Industrial XP suggests a Readiness Assessment for this [2]. Pikkarainen and Passoja have studied a suitability assessment

method for agile software development practices [15].

In software development the agility is mostly concentrated only on the project-level software process, but in manufacturing the concept of agility is really much wider covering also

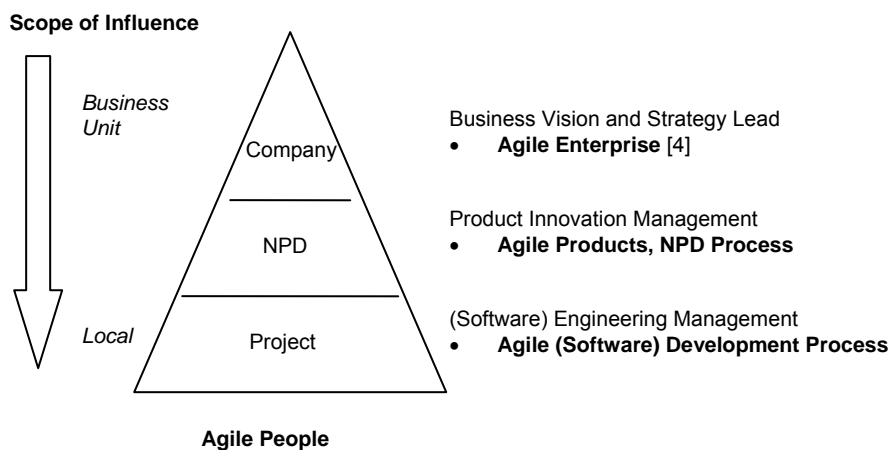
- (Agile) Product
- (Agile Competitive) Environment and
- (Agile) Enterprise.

It is noticeable, that in Agile Manufacturing the very idea of making only the process agile without thinking the business and product would make only little sense. This thought reversed, leads to conclusion that thinking the business and organization enablers as well as the agility of the product would be greatly beneficial to any Agile Software Company.

## 2.2 System-Wide Agility in NPD

New product development (NPD) companies make their business by continuously introducing new innovative products (e.g., consumer electronics). In current global, turbulent market environments these companies face many sources of uncertainty [16]. Agility is thus often a significant competitive edge for them.

In large-scale NPD organizations, it is important to have a system-wide view of the whole organization [17]. Figure 1 is a high-level model of a large NPD company with different levels of operation. The (embedded) software project teams are connected to many other parts of the organization and even to parts outside the business unit. In embedded systems development, more efficient workflow can possibly be achieved for example simply by co-locating the related software and hardware developers [18]. This is an example of Lean Thinking in software development organization [19]. We have investigated the characteristic problems of NPD embedded software development projects elsewhere [11].



**Figure 1: Levels of NPD organization agility (adapted from [20]).**

Clearly, if we want the whole company to work in agile ways, it is not enough to focus on the team and project-level dimensions alone, as would a typical application of agile software methodologies. For example Sharifi and Zhang have proposed a methodology and framework for achieving agility in manufacturing enterprises [21]. The company has to first understand, how agile it currently is, and how agile it needs to become (drivers of agility). The necessary capabilities of agility (e.g., responsiveness, flexibility) can then be achieved with different providers stemming from the organization, technology, people, and innovation.

There is a need for a wider understanding of agility in software product development for successful

SPI. This leads to the following questions:

1. How does software project agility relate to NPD enterprise agility?
2. What are the implications for SPI (improving agility) in large-scale NPD organizations?

### **3 Agility of Large Software Projects in NPD**

#### **3.1 Agility Framework**

##### *3.1.1 Goals of Agility*

The first step for the organization to understand and then improve its software production agility is to really understand, what it is exactly that it wants to focus on. Agility is really a multidimensional concept. What kind of agility do we need? For example, do we want to

- decrease the Lead Time of producing new product features,
- release new product variants more often,
- introduce totally new product concepts more frequently,

and/or

- improve the cost efficiency of the software production?

The basic questions when studying the dimensions of agility in large organizations are thus

- What is the competitive edge of the company?
- Why agility (or lean) approach is taken in certain operations?
- How is it achieved (i.e., techniques to master, decisions to make, questions to answer)?

Figure 2 illustrates this multifaceted nature of enterprise agility. A strategic choice is to first select the kind of agility that the company needs. The principal goals are to be fast and responsive, to achieve high development productivity, and to create products with distinction and integrity [22/Ch. 1]. Different companies and even the same company under different circumstances could give weight ( $w_i$ ) to different facets of agility. Typically, there are then many possible ways to achieve the selected goals. The means are supported by different enabling factors. In a large NPD organization, the software product development is just one element contributing to the overall company agility.

To address the questions 'What' and 'Why', the company can – for example – benchmark the current products and processes against the main competitors. Products can be characterized with many attributes ranging from the measurable functional performance to subjective factors [23]. Key process attributes are the lead time and resource cost.

In practice it is not possible to maximize all those dimensions simultaneously (or the cost becomes prohibitive). The choices are many, and the answers may be different for different companies. Also, the choices may be different for the different products of the company. Is it better to invest in agility of the product by making an investment to the software platform, or is it better to make just a single product and vary that in agile ways? How much more could we sell our products, if we allowed better customizing? What would be the development cost of the extra features, and how would that change our place in the market?

Such basic questions must be answered before going into business. Furthermore, this is an evolutionary process. In turbulent environments, the strategy choices have to continuously be adjusted following the current business conditions.

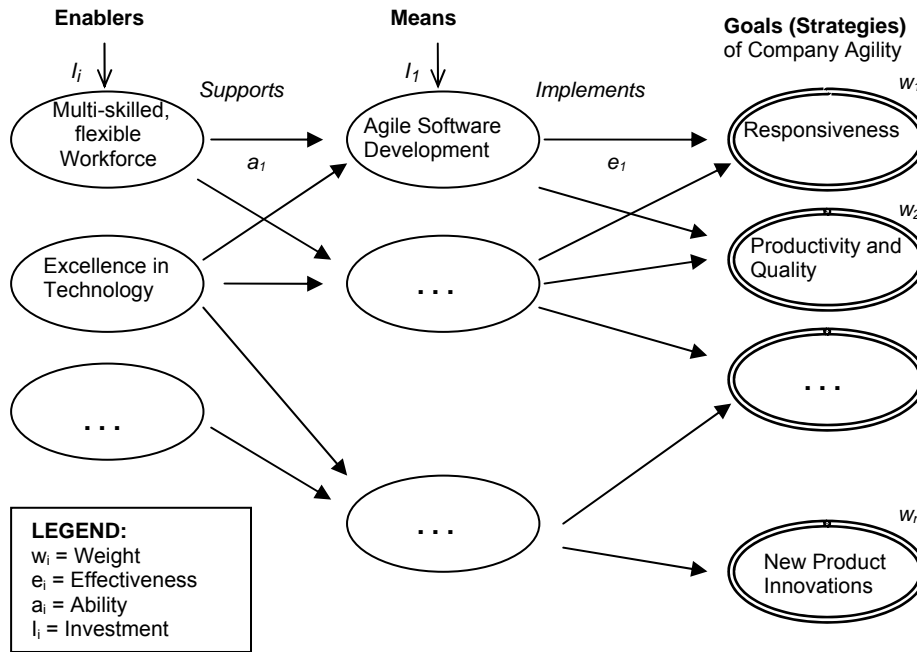


Figure 2: Some components of achieving NPD company agility.

### 3.1.2 Means of Agility

Considering the ‘How’ question from the software production point of view, we realize that the right choice to increase agility is business-dependent. The options presented may not, however, be a complete list. As the ultimate goal for each company may differ from one to another, also the means to achieve the goal may differ.

Figure 3 illustrates how agile software development is influenced by many different factors (c.f., Fig. 2). Notably current agile software methods cover only a subset of this space.

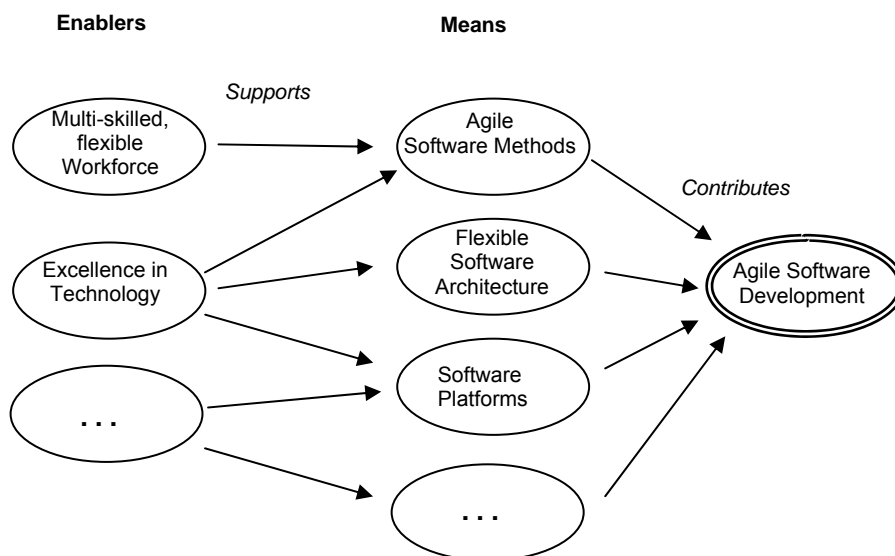


Figure 3: Some components of achieving software production agility.

### 3.1.3 Enablers of Agility

Looking again from a system-wide perspective (Fig. 1), we can see that in order to be able to really implement the goals of agile software production, many enabling factors and conditions, some external to the actual software development function, must be fulfilled. Without such abilities it is difficult (or ultimately, even impossible) to achieve sustainable agility.

For example, effective utilization of most agile software development methods relies on people factors. The project team should first be appropriately resourced, trained, and empowered. Only then can we expect to gain the full benefits of the agile methods.

In addition to the software production specific enablers, there are many common enabling factors in large NPD organizations. Excellence in the selected product technologies is a typical example of such factor (see Fig. 2 and 3). Organizational cultural factors are often overarching, yet more difficult to quantify.

Clearly, the different enabler factors are not disjoint, but are often mutually enforcing each other. On the other hand, even one weak or totally missing enabler capability may seriously disturb the agility of the company (e.g., inability to monitor and interpret critical market signals).

## 3.2 Practical Measures

To make this more practical, we have started collecting a set of characteristics questions rather than a set of answers about how agility could be achieved in a larger organization. These questions cover some specific issues of the goals, means, and enablers in new software product development, but do not represent a comprehensive list. A detailed generic questionnaire would be too voluminous to be provided. The list can be found in Appendix 1.

It is possible to develop this question list further to a more structured agility evaluation matrix. Table 1 shows examples of defining evaluation scales for some of the question items in Appendix 1. Such a questionnaire matrix could be used as an assessment tool for profiling the levels of project and organizational agility. The profile can then be used to direct the SPI activities according to the strategic needs of the company. We have applied similar ideas elsewhere for software project problem (risk) profiling purposes [11].

KEY AREA, FACTOR	Level of Agility		
	LESS	↔	MORE
<b>Responsiveness to market changes (new features):</b>			
Does the customization happen before or after the product is made?	Early commitment (mass-production)	-	Late decision (tailoring)
...			
<b>Productivity and quality (operational profitability):</b>			
...			
How are the (embedded) software upgrades and licensing handled?	Fixed price	-	Feature-based selection
<b>New product innovations (competitive and desirable product):</b>			
...			
Is the product innovative or functional?	Functional	-	Innovative
...			
<b>Agile software methods (accommodating change):</b>			
...			
Do the teams have control over the process?	One organizational standard (one-size-	-	Project teams define their processes.

	fits-all)		
...			
<b>Flexible software architecture, software platforms (developmental flexibility):</b>			
Is the product architecture designed for flexibility?	Monolithic	-	Reconfigurable, modular
...			
<b>Multi-skilled, flexible workforce:</b>			
Do people have the right attitudes to work in a volatile environment?	Resistance to change	-	Comfortable with change
...			
<b>Excellence in technology (preconditions):</b>			
...			
Are we capable of identifying underlying trends to predict future changes (learning)?	Reactive	-	Innovative (proactive)
<b>Organizational flexibility:</b>			
...			
What is the 'amicability index' of the organization?	Only through hierarchies (need-to-know)	-	Free flow
...			

Table 1: NPD agility evaluation grid outline.

Note that we do not attempt to provide here detailed answers about how to actually implement agility improvements in any particular organization or product development project. For example, flexible software architecture design is a whole dedicated field of research.

#### 4 Example Case

At the time of this writing, we are not ready to publish detailed experience data about our propositions. However, the following characteristic product development example illustrates some points that in our experience can actually be observed.

Many telecommunications network element products have to support different transmission frequencies. Often new customers need new frequency support. In typical implementations the reaction to this means new hardware plug-in unit types with the associated embedded software support. Because of stringent time-to-market requirements the development of such new products (variants) is typically done with concurrent engineering of the new hardware and the embedded software functionality. Two different scenarios can be characterized:

- a) A non-agile product development organization does not realize the need for close co-operation between the related hardware and software projects. The different functions are “siloeed” with little collaboration. The hardware development proceeds independently to the software project. The software developers cannot influence the new hardware design, and the knowledge sharing between them is scarce. The software architecture has not been designed for accommodating new hardware unit types. Consequently, the new unit support requires extra design and testing efforts, and is error-prone. A Waterfall-based development model reveals major hardware-software integration problems at a late stage, while document-based milestones indicated illusory progress. All these factors cause time-to-market delays, leading to poor *Responsiveness* as well as lower *Productivity* (c.f., Fig. 2).
- b) A more agile way of doing this product development is to begin with a common systems engineering, which creates a modular, flexible product architecture (*Excellency in Technology* in Fig. 2 and 3). The need for different hardware plug-in unit variants is proactively anticipated from the beginning (possibly with *Software Platforms*). The software architecture is designed for easy



incorporating of new hardware unit types, based on standardized hardware/software interfaces (*Flexible Software Architecture*). The related hardware and software projects are managed in close co-operation, with daily face-to-face interactions with the hardware and software developers (*Multi-skilled, flexible Workforce*). The product integration proceeds smoothly in an iterative fashion, avoiding major breakage delays (*Agile Software Methods*). The net effect of all these contributing elements is good *Responsiveness* of the NPD company (Fig. 2).

Reflecting Table 1, we can see that many characteristics of those different approaches correspond to the *Level of Agility* spectrum.

### 5 Discussion

In Section 3 we investigated, what agility means for software product development overall, and showed some ways of achieving it in large-scale NPD contexts. The company has to first choose, how agile it needs to be. The software methodology choices should be based on the business the company is striving for [24]. The SPI activities should then be aligned with those considerations.

A product development organization may need different kind of agility in different phases of its business [25]. Also, the phase of the product evolution may affect the appropriate level of agility needed. In the innovative phase more comprehensive agility is usually needed. In later phases the product becomes typically more sustainable. Many commercial software products and other new products containing embedded software are by nature innovative. However, there are also many types of software productions which are not so – like more fixed end-products or some legally regulated software products. They might do better with cost optimized, i.e., lean systems. However, even in apparently stable market areas there may occasionally emerge disruptive changes, requiring fast and sometimes radical responses.

Conboy and Fitzgerald have investigated similar issues [5]. In their opinion, the Agile Manifesto is too informal to be a really useful concept base. Consequently, they formulate a systematic definition of agility, and show how software project teams ('ISD teams') could really be agile. Our purpose in this paper has been to go further beyond the software development team level, considering the entire NPD company.

Viewing the NPD company as a whole, we can see that there are forces supporting and inhibiting agility at the software project level (Table 1). Ideally all the elements of the company are aligned for agility. However, in practice there are often some parts of the large organization that are better set for agile ways of operation while some other parts may have different strengths. For example, if the overall company hiring process does not attract suitable people, it will be difficult to staff the software projects accordingly. Naturally such disabling factors should first be removed. It is very important to be aware of the factors that could limit the organization's agility. Clearly, some of the enabling factors are more significant while some lacking or weaker factors can often be compensated (with some cost). Like illustrated in Fig. 1, the magnitude and effect of the enablers/disablers varies according to the level of influence.

It is important to understand the interfaces between the agile software project teams and the rest of the NPD organization, both ways (c.f., Fig. 1). The business connection to the software development is essential. Also, the product development program management must understand the nature of the agile software methods, and govern the software projects accordingly [26]. For example Augustine has addressed such concerns ('adaptive IT enterprise') [27/Ch. 4]. The software project teams should adapt their agility to the surrounding organizational context, which may impose both enabling and inhibiting forces.

Remarkably, there can be seen a clear tension between large software organizations and the agile process model idea of self-organizing teams. A self-organizing team in this context has control over the business. It is typically easier in small organizations to make this happen than in larger organizations, because the business management is closer to the developers. For instance XP expects the onsite customer to be authorized and capable of making all the business decisions. In large-scale NPD projects it may be difficult to assign a customer (or even a proxy), and the business decisions are usually made by different stakeholders. We hypothesize that the reason why applying

agile methods in large organizations is so often difficult is this missing linkage between the business model and the software development, i.e. agility needs to be a strategic choice.

Scaling agility to large software projects is challenging, but there exist already some examples of how to do this [3, 28]. Scaling up agility in large embedded software development is even more challenging as the number of external dependencies increases. In large, established product development organizations successful adoption of agile software methods depends often more on software project external factors than software engineering technical issues [29].

Furthermore, in large product development environments every change has more wide-ranging impacts. Thus the span of agility needs to be longer, and the focus on changes moved from implementation to anticipation. Typically there are product families and long product evolution lifecycles to be taken into account. It is not enough to just adapt to the changes reactively when they come.

The implications of all this for successful SPI in large NPD organizations is that the perspective has to be expanded in order to avoid isolated local optimizations. The overall business and organizational factors have to be taken into account – i.e., SPI-in-the-large.

## 6 Conclusions

In this paper, we have investigated agile software development and SPI in large-scale NPD organizations. What could a future organization look like where all dimensions of agility are increased? It would have at least the following qualitative (Question 1, Section 2.2):

- The organization values multi-skilled people and intellectual capital assets.
- It consists of many virtual organizations, reconfigured dynamically, striving to combine the advantages of small entrepreneurial companies and large-scale production economics. Parts of the products would be created with collaborators and subcontractors, leveraging the core competencies and resources of each.
- It releases frequently new product features and varieties and also new breakthrough products following the prospective market changes.
- Its products would be based on flexible designs, be open, and customizable.
- Having open, flexible process frameworks, which product development teams could tailor themselves according to their current needs. Tools support the work.

In such an organization, agile software project teams are naturally incorporated. They are appropriately resourced, empowered, and governed. The software project teams in turn understand all the time, how they serve the company business best, even under turbulence. The software process models used support both short-term flexibility as well as the longer-term proaction needs of the NPD organization.

Considering our Question 2 (Section 2.2), the implications of this study can be summarized as follows:

- The NPD should first make a conscious strategic choice of its targeted business operational model (e.g., product leadership). Agility is then a potential solution to certain business models (Section 3.1.1). Different companies need different levels of agility, depending on the market environment and product types. The investments in agility should be balanced based on how agile the company needs to be.
- There are many ways to achieve and improve agility in software product development (Section 3.1.2). Agile software methods are just one subset of such means.
- The agile software process models are based on certain profound (but often hidden) assumptions about the project teams, products, and customer business environments. The NPD organization must consider the validity in their case.

- The current agile software process models address only a local subset of the product development operation. They are, therefore, not enough to improve the agility of the entire NPD function. In particular, in large organizations developing complex products and/or wide product varieties, the perspective has to be extended from individual release projects to longer-term product evolution and portfolio management. Change creation and proactive capabilities have to be considered, but current agile software methodologies focus just on reaction.
- Achieving total and sustainable agility requires many enabling factors, some specific to software development but also many factors, that must be satisfied outside the software production function (Section 3.1.3).

Finally, this paper leaves room for further study:

1. Elaborating the frame of the agility influencing factors (Fig. 2 and 3).
2. Developing a systematic assessment tool for measuring the level of NPD company agility and its enabler/disabler factors, following the principles of Table 1.
3. Examining the correlation between the Goals, Means and Enabler factors and the question why adopting agile methodologies in software teams of large development organization sometimes fail.
4. Comparing different agile software methodologies with respect to how they fit in large product development organizations. What is the interface between the (embedded) software project teams and the rest of the NPD organization? What are the assumptions, prerequisites, and enabling factors?
5. Collecting more empirical evidence for validating our propositions. This could be done by conducting a project survey with the questionnaire list in Appendix 1 / Table 1.

## 7 Literature

- 1 Ronkainen J., Abrahamsson, P.: Software development under stringent hardware constraints: Do agile methods have a chance? In: Proc. 4th Int'l Conf. eXtreme Programming and Agile Processes in Software Engineering (XP) (2003) 73-79
- 2 Industrial XP: <http://industrialxp.org/> (accessed August 2005)
- 3 McMahon, P.: Extending Agile Methods: A Distributed Project and Organizational Improvement Perspective. CrossTalk 18(5) (2005) 16-19
- 4 Preiss, K.: Agility – the Origins, the Vision and the Reality. In: Proc. Int'l Conf. Agility (ICAM) (2005) 13-21
- 5 Conboy, K., Fitzgerald, B.: Toward a Conceptual Framework for Agile Methods: A Study of Agility in Different Disciplines. In: Proc. ACM workshop on Interdisciplinary software engineering research (WISER) (2004) 37-44
- 6 Gould, P.: What is agility? IEE Manufacturing Engineer 76(1) (1997) 28-31
- 7 Katayama, H., Bennett, D.: Agility, adaptability and leanness: A comparison of concepts and a study of practice. Int. J. Production Economics 60-61 (1999) 43-51
- 8 Yusuf, Y.Y., Sarhadi, M., Gunasekaran, A.: Agile manufacturing: The drivers, concepts and attributes. Int. J. Production Economics 62 (1999) 33-43
- 9 Kettunen, P., Laanti, M.: How to Steer an Embedded Software Project: Tactics for Selecting Agile Software Process Models. In: Proc. Int'l Conf. on Agility (ICAM) (2005) 241-257
- 10 Schwaber, K.: Will the Real Agile Processes Please Stand Up? Executive Report 2(8), <http://www.cutter.com/project/fulltext/reports/2001/08/index.html> (2001)
- 11 Kettunen, P.: Troubleshooting Large-Scale New Product Development Embedded Software Projects. In: Proc. 7<sup>th</sup> Int'l Conf. Product Focused Software Process Improvement (PROFES) (2006) 61-78
- 12 Kähkönen, T.: Agile Methods for Large Organizations – Building Communities of Practice. In: Proc. Agile Development Conf. (ADC) (2004) 2-10
- 13 Nerur, S., Mahapatra, R., Mangalaraj, G.: Challenges of Migrating to Agile Methodologies. Communications of the ACM 48(5) (2005) 73-78
- 14 Cockburn, A.: Two Case Studies Motivating Efficiency as a “Spendable” Quantity. In: Proc. Int'l Conf. Agility (ICAM) (2005) 1-5
- 15 Pikkarainen, M., Passoja, U.: An Approach for Assessing Suitability of Agile Solutions: A Case Study. In: Proc. Int'l Conf. eXtreme Programming and Agile Processes in Software Engineering (XP) (2005) 171-179
- 16 Mullins, J.W., Sutherland, D.J.: New Product Development in Rapidly Changing Markets: An Exploratory Study. Journal of Product Innovation Management 15 (1998) 224-236
- 17 Rand, C., Eckfeldt, B.: Aligning Strategic Planning with Agile Development: Extending Agile Thinking to Business Improvement. In: Proc. Agile Development Conf. (ADC) (2004) 78-82
- 18 Waters, M., Bevan, J.: Journey to Lean. IEE Engineering Management 15(4) (2005) 10-13
- 19 Poppendieck, M., Poppendieck, T.: Lean Software Development: An Agile Toolkit. Addison-Wesley (2004)
- 20 Yourdon, E.: Managing High-Intensity Internet Projects. Prentice Hall (2002)
- 21 Sharifi, H., Zhang, Z.: A methodology for achieving agility in manufacturing organizations: An introduction. Int. J. Production Economics 62 (1999) 7-22
- 22 Wheelwright, S.C., Clark, K.B.: Revolutionizing Product Development: Quantum Leaps in Speed, Efficiency, and Quality. The Free Press (1992)
- 23 Garvin, D.A.: Competing on the eight dimensions of quality. Harvard Business Review 65(6) (1987) 101-109
- 24 Anderson, D.J.: Agile Management for Software Engineering. Prentice Hall (2004)
- 25 Kontio, J.: Why Agile Now? The Agility Needed for Business Success. In: 3rd Agile Software Development Seminar (ASDS), <http://agile.vtt.fi> (2005)

## Session 12: SPI and Development Processes

- 26 Highsmith, J.: Agile for the Enterprise: From Agile Teams to Agile Organizations. Executive Report 6(1), <http://www.cutter.com/project/fulltext/reports/2005/01/index.html> (2005)
- 27 Augustine, S.: Managing Agile Projects. Prentice Hall (2005)
- 28 Eckstein, J.: Agile Software Development in the Large: Diving Into the Deep. Dorset House Publishing (2004)
- 29 Lindvall, M., et al.: Agile Software Development in Large Organizations. IEEE Computer 37(12) (2004) 26-34
- 30 Cockburn, A.: Crystal Clear: A Human-Powered Methodology for Small Teams. Addison-Wesley (2004)

### **8 Author CVs**

#### **Petri Kettunen**

Petri Kettunen is a R&D software engineering specialist with Nokia Corporation, Finland. He received his M.Sc. in Computer Science at Helsinki University, and Lic.Sc. (Tech.) at Helsinki University of Technology. He has been involved with industrial embedded software development for more than 15 years in various positions. His current research interests include new product development project management methods, and embedded software engineering process models.

#### **Maarit Laanti**

Maarit Laanti is a Senior Project Manager with Nokia Corporation, Finland. Her interest is in new product development, project management and leadership. She has been leading various size software development projects over ten years in Nokia, including a two-year assignment in Dallas, USA. She holds a M.Sc. in Data Transfer and Computer Sciences from Helsinki University of Technology.

### **Appendix 1: Software Product Development Agility Considerations**

The following is a list of example questions that a larger NPD software company could use to evaluate their level of agility. The list has been compiled by the authors by screening a large amount of questions collected from the referred and anonymous sources.

The main categorization reflects the major principles of agility: delivering value to the customer (goals), being ready for change (means), valuing human knowledge and skills, and forming virtual partnerships (enablers) [16]. For the enabler factors the answers characterize the abilities ( $a_i$  in Fig. 2), and for the means the effectiveness ( $e_i$ ), respectively. Some questions could be related to many factors. The list is by no means exhaustive.

#### **GOALS (Fig. 2)**

- Responsiveness to market changes (new features):
  - Is the product itself agile or not? How flexible/adaptive/customizable is the product? Does the customization happen before or after the product is made?
  - Which dimensions of the product are optimized? At which cost?
  - Are frequent client/user interviews and prototyping used (proaction for changes in demand) [5]?
  - What would be the development cost of the extra features, and how would that change our place in the market?
- Productivity and quality (operational profitability):
  - How much product is developed “in-house” and how much outsourced?
  - What are the most value-adding dimensions of the product placement?
  - How are the (embedded) software upgrades and licensing handled?
- New product innovations (competitive and desirable product):
  - Does the product belong to some product family, or exist by itself? What kind of product evolution is there? What is the expected life-time of the product?
  - Is the product innovative or functional? What is the value for the customer?
  - Is there active change of hardware, software to add more customer value (change creation) [5]?
  - Can the customer participate in defining the product, and how much? How much more could we sell our products, if we allowed better customizing?

#### **MEANS (Fig. 3)**

- Agile software methods (accommodating change):
  - Is the team oriented to customer-value [27]?
  - How much new technology is utilized, at what risk?
  - Do the teams have control over the process? Are people allowed to make a second implementation attempt, as the first one chosen is seldom the best? Are hand-drawn skeleton-type of designs allowed, even encouraged [19]?
  - What tools could be used to support the team process and encourage agile working practices? Can they effectively pass along the information? Do they have good automated regression tests [30]?

- Does the project team get together periodically for reflective improvement [30]?
- Is there a (secondary) project goal to set up for the next project [30]?
- Flexible software architecture, software platforms (developmental flexibility):
  - Is the product architecture designed for flexibility? How is the design done?<sup>1</sup>
  - Is/should there be a product platform? What kind of platform? Is it better to invest on agility of the product by making an investment to the software platform, or is it better to make just a single product and vary that in agile ways? If a software platform is used, is it more skeleton-type allowing loose connections?
  - What is the complexity of the product? Are there some frameworks or patterns?
  - Are there ready-made pieces that could be used (software reuse)?

### ENABLERS (Section 3.3)

- Multi-skilled, flexible workforce:
  - Do people have the right attitudes to work in a volatile environment? Do they communicate frequently and with good will [30]? Do the teams have sustainable self-discipline [27]? Can people easily detect something needs attention? Will they care enough to do something about it [30]?
  - Are they mostly generalists or specialists? Do people learn efficiently?
  - Are their competences sufficient [27]? What is their 'Cockburn level' [30]?
- Excellence in technology (preconditions):
  - Do we have the technology needed for the product? Are the project teams familiar with it? Do we have right (agile) production mechanisms and tools?
  - Are new technology experts used (reaction to change) [5]?
  - Do people have experience from / information on past technology adoption problems? Are we capable of identifying underlying trends to predict future changes (learning) [5]?
- Organizational flexibility:
  - Do the projects receive appropriate nourishment from Executive Sponsors (decisions, resources) [30]?
  - What dependencies do the product requirements have to external organizations? Is subcontracting used?
  - Are the team sizes small enough [27]? Have multi-project (program) management interdependencies been minimized [28]?
  - How much freedom do people have concerning their own work? Can people disagree and debate safely [30]? Do people have access to the code libraries of other projects [5]? Do people have enough quiet space to work?
  - Does the organization eliminate bureaucracy and let them design [30]? How to reduce the decision feedback time [27]? What can be left unwritten and communicated otherwise [3]? Do the teams have intense collaboration [27]?
  - What is the 'amicability index' of the organization, indicating how easily information passes from one part of the organization to another [30]? Are changes communicated only top-down or in both directions?
  - Is the organization optimized as a whole or just as integral parts [19]?

---

<sup>1</sup> For example, MobileD method suggests that top-down driven architectural and product line approaches are inflexible in fast-cyclic, end-user and business driven mobile software developments.





# Towards a { (Process Capability Profile)-Driven (Process Engineering) } as an Evolution of Software Process Improvement

*Clenio F. Salviano <sup>(1,2)</sup> and Mario Jino <sup>(2)</sup>*

*<sup>(1)</sup> Divisão de Melhoria de Processo de Software - Centro de Pesquisas Renato Archer  
Ministério da Ciência e Tecnologia (DMPS CenPRA MCT),  
Rodovia Dom Pedro I, km 143,6, CEP 13069-901, Campinas – SP – Brazil  
clenio.salviano@cenpra.gov.br*

*<sup>(2)</sup> Departamento de Engenharia de Computação e Automação Industrial  
Faculdade de Engenharia Elétrica e de Computação - Universidade Estadual de Campinas  
(DCA FEEC UNICAMP)  
Cidade Universitária "Zeferino Vaz", CEP 13081-970, Campinas, SP – Brazil  
jino@dca.fee.unicamp.br*

## **Abstract**

This article presents an innovative proposal towards a process engineering, driven by process capability profile, for software and any other knowledge intensive human work as an evolution of the current software process improvement. The proposed process engineering is based on a critical view on the current state of the practice and state of the art presented as seven issues and opportunities. An exemplar approach, named PRO2PI, for the proposed process engineering is also presented. The PRO2PI approach has been constructed using an industry-as-laboratory approach in 200 process improvement projects since 1999. The proposed process engineering and the PRO2PI approach aim to guide the software intensive industry, specially small software intensive organizations, in the establishment of relevant, feasible, opportunistic, systemic, representative, traceable, specific and dynamic process capability profiles to drive more innovative and successful business oriented process improvements cycles, as an evolution of the current process improvement approaches composed by the implementation of the fixed "one size fits all" maturity levels of SW-CMM and CMMI models.

## **Keywords**

process improvement, process engineering, process capability profile, process capability model, ISO/IEC 15504, small software organizations

### 1 Introduction

The software industry has been using software process improvement approaches, based on the maturity levels of SW-CMM and CMMI staged models, to improve its business. However, in practice, most organizations are, usually in an informal basis, doing more than just implement the maturity levels. They are using multiple models and other references for process improvement, and they are using additional process areas. They are also using process improvement in other areas related to software processes. This article presents a proposal towards a Process Capability Profile Driven Process Engineering (PCDE) as an evolution of the current Software Process Improvement based on Process Capability (and Maturity) Models. This proposal has been constructed since 1999 using experiences from many process improvement projects in software industry. An industry-as-laboratory approach [Potts 1993] was used, instead of the traditional research-then-transfer approach. This proposal aims to formalize what software intensive organizations are already doing, in an informal way.

The article is organized in six sections. This introduction is the first section. The second section presents a critical view on the current scenario as seven issues and opportunities that oriented this research. The third section presents a proposal for a process engineering as an evolution of the current software process improvement. The fourth section presents an approach for the proposed process engineering. The fifth section presents experiences in software industry. Finally, the sixth section presents a conclusion.

### 2 Seven issues and opportunities

As a synthesis of a critical view on the issues and opportunities from the current state of the art and state of the practice of software process improvement area and related areas, seven views are presented.

- **Fixed maturity levels:** The software process improvement area was established based on the fixed “one size fits all” maturity levels of SW-CMM model [Paulk et al. 1994] from around 1993 until its retirement around 2002 and of CMMI staged representation models [Chrissis et al. 2003] since 2000. These maturity levels guided the software industry on the improvement of the process for software development projects. As all good pioneer work, the maturity levels established an area based on “one size fits all” approach, as, for example, Henry Ford established the car industry with his “T model” black car. After the establishment, there is a need for more flexibility to cope with the diversity. Most of the problems of software process improvement identified by the community, as, for example, the ones from Conradi and Fuggetta [2002] and Rifkin [2002], are due to the fixed maturity levels of SW-CMM and CMMI models.
- **Flexibility of continuous architecture:** The ISO/IEC 15504, also known as SPICE (Software Process Improvement and Capability dEtermination) proposed the continuous architecture as a more flexible alternative to the fixed maturity levels of staged models. The continuous architecture defines two dimensions for process capability models: one with a set of processes and another with process capability levels. An organization can choose a process capability profile, which is a subset of processes, each one in at a process capability level, to guide the improvement of its more relevant processes. Examples of continuous models are the ISO/IEC 15504-5 [2006], FAA iCMM model [Ibrahim 2000] and the continuous representation of CMMI models [Chrissis et al. 2003]. In order to be more used by software industry, the continuous models need an evolution of the current approaches for software process improvement. There a need for methodological support to use the flexibility to define process capability profiles.
- **Using multiple models:** Many organizations are using simultaneously elements of multiple models as reference for process improvement. Some of these models are capability maturity models, other are certification models, other generic process models and any other kind of model

that represents best practices. Maybe the most common combination nowadays is the combined usage of CMMI-SE/SW staged representation, ISO 9001, RUP and PMBOK.

- **Define or use models:** Nowadays there is a clear cut separation between groups that define process capability models and the ones that only uses these models. Examples of groups that defines models include SEI groups, in the development of SW-CMM and CMMI models, ISO/IEC JTC1 SC7 WG10 group, in the development of ISO/IEC 15504-5 model, FAA groups, in the development of iCMM model, and MPS-BR group, in the development of MR-MPS model [Weber et al. 2005]. Examples of groups that only use one of these models are the SEPG groups of software intensive organizations.
- **More specific models:** There is a tendency for more specific process capability models for technological segment or domain under the ISO/IEC 15504-2 framework. ISO/IEC 15504-2 [2003] defines six process capability levels as a measurement framework, the requirements for process reference models and process assessment process, and other orientations and requirements. In addition to the ISO/IEC 15504-5 model, CMMI models, iCMM model, MR-MPS model, there already other models under ISO/IEC 15504-2 framework, including the OOSPICE model for component-based software engineering [Stallinger et al. 2002] and automotive SPICE model for car's software supplies for the automobile industry [Automotive SIG 2005].
- **Model-driven engineering:** Model-driven engineering is an emergent area that put model in the center of an engineering. Favre concludes a model-driven engineering overview with the definition of two terms. "Model engineering is the disciplined and rationalized production of models. Model-driven engineering is a subset of system engineering in which the process heavily relies on the use of models and model engineering" [Favre 2005, p. 26].
- **Fundamental concepts:** David Card observed that the process improvement approaches are "all based on very similar concepts and techniques". However, because these approaches "have evolved or been adapted to software engineering largely without the participation of the academic research community", "they are considered competitors". "The packaging obscures the underlying principles. Eliciting and refining underlying principles is the role of science" [Card 2004].

An evolution of the current software process improvement should extend the fixed maturity levels using the flexibility of continuous models, supporting the usage of multiple models, including the development of more specific models, allowing an organization to define and use models as in model-driven engineering, by eliciting and refining the fundamental concepts and underlying principles of the current state of the practice and state of the art of software process improvement.

### ***3 The basis and a proposal for a process engineering***

Process can be considered as virtually any granularity. Given an organization unit, the complete view about what people do in that organizational unit can be seen as a process, in this case, the organization unit process. Any subset of the organizational unit process can be considered also as process, as, for example, the process for unit tests. An appropriate granularity for process is related what the ISO/IEC 15504-5 defines as "process" and what the CMMI models define as "process area". In spite of the difference in name, both concepts are similar. ISO/IEC 15504-5:2006 defines 48 processes and CMMI-SE/SW v1.1 defines 22 process areas. The term "process area" is used in this research to mean both concepts. Therefore, a process is a set of actions that people do that could be represented by a process area, from any model.

The six process capability levels as defined by ISO/IEC 15504-2 are the best candidate for the fundamental law of process improvement. The performance of any process at an organization unit can be estimated by a characterization of the process in one of the six capability levels from level 0, incomplete, to level 5, optimizing, given that the process is abstracted as a process (area).

Process capability profile as a combination process capability level and process area, is the best candidate for the basic unified concept of a reference for a process improvement. Each element of a process capability profile represents a process, in the granularity of, and represented by, a process area, at a process capability level. Therefore a process capability profile represents a process at a granularity of the aggregation of the processes in each element. The Figure 1 illustrates a candidate

for the basic relationship for process engineering.

Using the M0-M1 hierarchy [Bézivin 2003, slides 62 and 66], a process is a part of the world (M0) represented by a model, in the modeling space (M1). In this case the model is a process capability profile that represents the process, under the process capability aspect. Note that a process description also can represent a process, in this case under the process description aspect. A process capability profile can be a prescriptive or descriptive model. As a prescriptive model a process capability profile drives the improvement of a process towards a better process using the requirements and orientations from the process capability profile. As a descriptive model the process is represented by a process capability profile that is a result from a process assessment process.

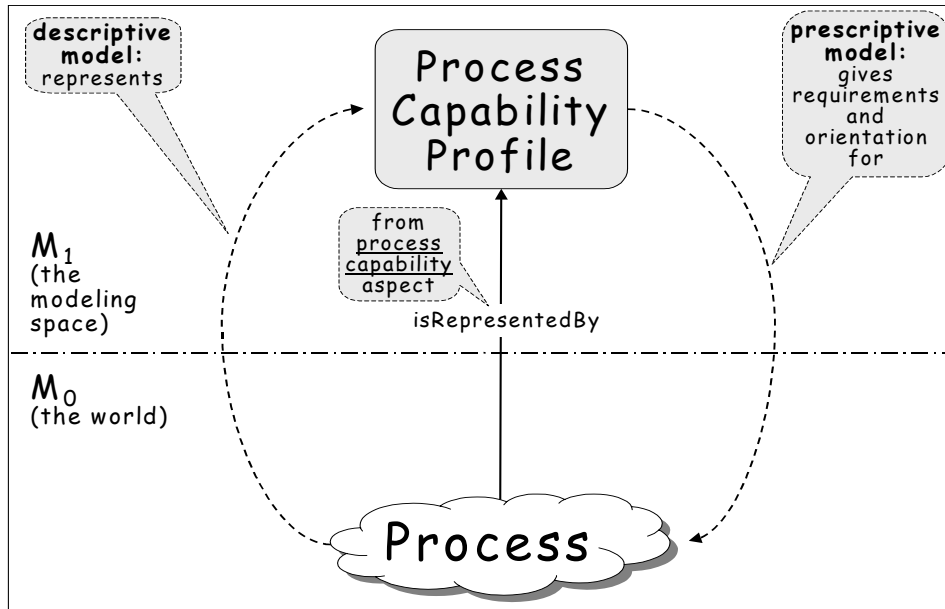


Figure 1 – Process and process capability profile

The short term “{(Process Capability Profile)-Driven (Process Engineering)}” and a long term “{(Process Capability Profile)-Driven [Software and any other Knowledge Intensive Human Work] (Process Engineering)}”, both with the same meaning and with the same initials (PCDE), are proposed as an evolution of the current software process improvement. A more complete definition for the proposed process engineering is presented at Figure 2.

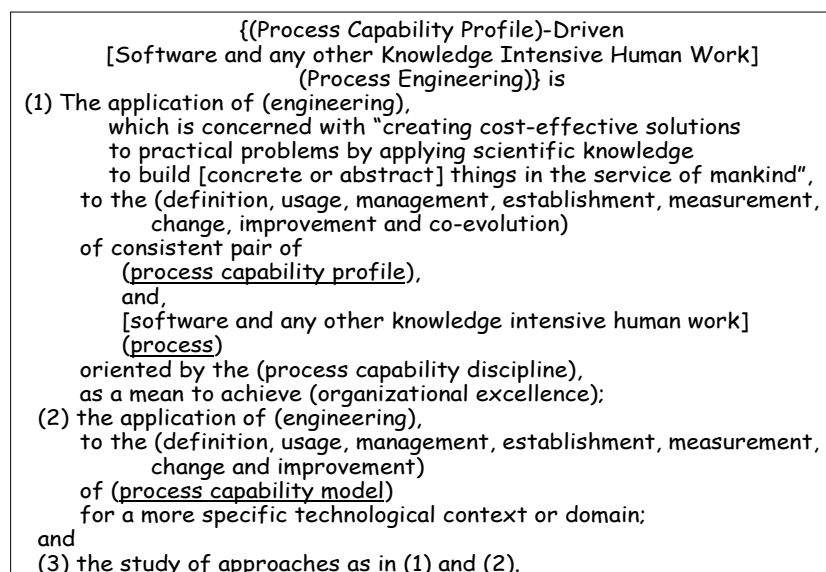


Figure 2 – A proposed definition for the proposed process engineering

Process improvement is not anymore only for software processes. The current versions of SEI CMMI and ISO/IEC 15504 use the term “system” as a more generic boundary that includes software. A boundary line related with the term “knowledge worker” seems to be more appropriate. This term was used by Peter Drucker in his 1959 book, Landmarks of Tomorrow as “a knowledge worker is anyone who works for a living at the tasks of developing or using knowledge”. “For example, a knowledge worker might be someone who works at any of the tasks of planning, acquiring, searching, analyzing, organizing, storing, programming, distributing, marketing, or otherwise contributing to the transformation and commerce of information and those (often the same people) who work at using the knowledge so produced” [searchCRM 2006]. The knowledge worker includes those in the information technology fields, such as programmers, system analysts, technical writers, academic professionals, researchers, and so forth”. Knowledge workers include people outside of information technology, such as lawyers, doctors, diplomats, lawmakers, marketers, managers, bankers, teachers, scientists of all kinds and students of all kinds.

This PCDE definition is based on a combination of definitions for engineering, software engineering, software process engineering, software process improvement, process capability model engineering and model-driven engineering.

#### 4 An exemplar approach for the proposed process engineering

The proposed process engineering was conceived during the many cycles of exploration, application and consolidation of a six years research following the industry-as-laboratory approach. During these six years an exemplar approach has been developed for the proposed process engineering. This approach, presented for the first time in Salviano et al. [2004], is named PRO2PI (Process Capability Profile to Process Improvement). Figure 3 illustrates an overview of PRO2PI approach.

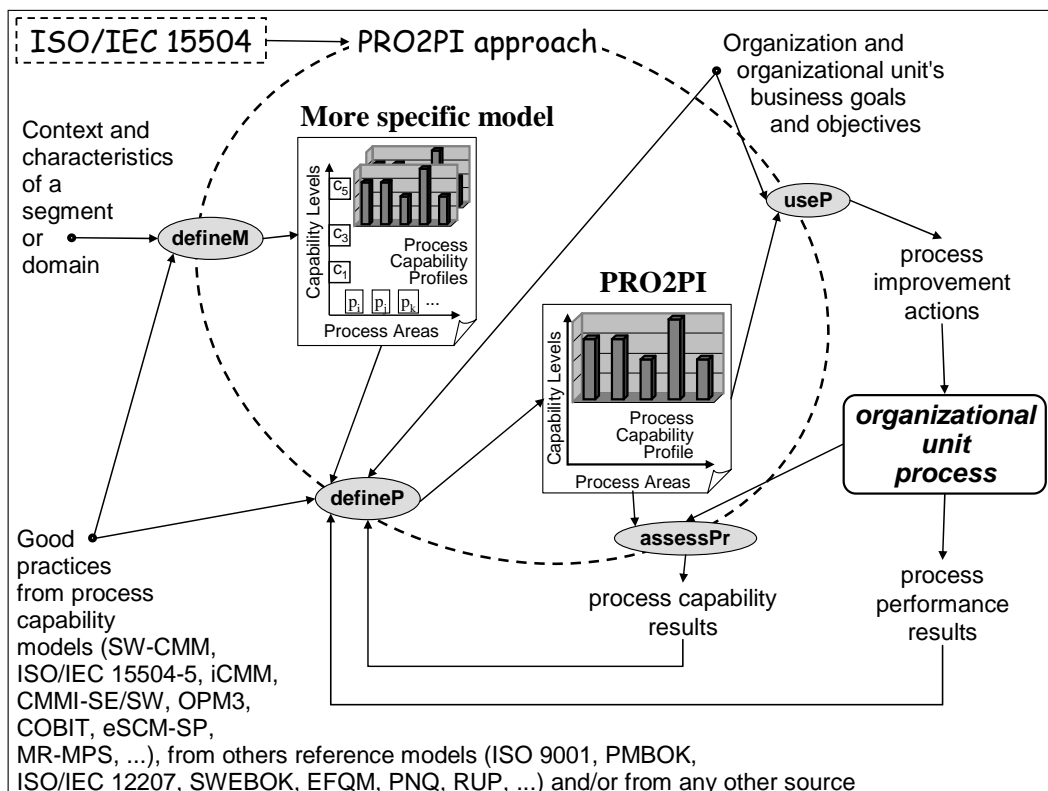


Figure 3 – Overview of PRO2PI approach

A more specific model can be defined for a specific context or using characteristics of a segment or domain. This more specific model uses selected good practices from more generic process capability models (SW-CMM, ISO/IEC 15504-5, iCMM, CMMI-SE/SW, OPM3, COBIT, eSCM-SP, MR-MPS, ...),

from other types of reference models (ISO 9001, PMBOK, ISO/IEC 12207, SWEBOK, EFQM, PNQ, RUP, ...) and/or from any other source. This definition, which can be also an update, is represented by the *defineM* (*define or update model*) function in Figure 3. This more specific model can be a staged model, with a hierarchy of process capability profiles, or a continuous model, with a set of process areas and process capability levels.

A process capability profile to process improvement, also named as PRO2PI, can be defined in alignment with the organization and organizational unit's business goals, using selected good practices from a more specific model, from process capability models (SW-CMM, ISO/IEC 15504-5, iCMM, CMMI-SE/SW, OPM3, COBIT, eSCM-SP, MR-MPS, ...), from other types of reference models (ISO 9001, PMBOK, ISO/IEC 12207, SWEBOK, EFQM, PNQ, RUP, ...) and/or from any other source. The definition of this PRO2PI can use also analyses from the process capability results of a process assessment and from the process performance results of the current process. This definition, which can be also an update, does not need to be done at once. Rather it is better to do it in an incremental way. This definition or update is represented by the *defineP* (*define or update a PRO2PI*) function in Figure 3.

A process improvement cycle uses a PRO2PI, again in alignment with the organization and organizational unit's business goals, to plan and realize process improvement actions to change the organization unit process towards a process driven by the PRO2PI. This usage of PRO2PI is represented by the *useP* (*use PRO2PI*) function in Figure 3

The organizational unit process can be examined using a process assessment oriented by a PRO2PI. This process assessment produces a process capability results. This assessment is represented by the *assessPr* (*assess process*) function in Figure 3. These four functions (*defineM*, *defineP*, *useP* and *assessPr*) represent an overview of the PRO2PI approach. The ISO/IEC 15504 is the conceptual base for PRO2PI approach.

PRO2PI approach has been used with the support of four elements: a set of eight properties for a good PRO2PI (relevant, feasible, opportunistic, systemic, representative, traceable, specific and dynamic), a model that unified the elements from the most relevant process capability models, a set of measures and phases for a process improvement cycles including a function to define, update or use a PRO2PI. The initial phases of this cycle are support by a method for a workshop to establish a process capability profile to process improvement (PRO2PI-WORK). The current versions of these elements are defined in Salviano [2006] and previous versions of PRO2PI are presented in Salviano and Jino [2004] and Salviano et al. [2004].

The PRO2PI-WORK method is composed by activities to achieve the following objectives:

- to capacitate members of the organization in process engineering fundamentals and the most relevant process capability models;
- to consolidate information about the organization and the organization unit;
- to consolidate information about the business goals for a process improvement;
- to model the current relevant processes of the organizational unit;
- to define a process capability profile to process improvement;
- to understand the current capability of the organizational unit process;
- to define orientations for process improvement actions; and
- to reinforce the motivation and commitment to process improvement.

The definition of the process capability is based on an analyses of the results of a set of activities including: an identification of major current problems, a view on the strengths, weakness, opportunities and threats of the organizational unit; an identification of relevant for the organization for a each selected process area; an identification of the current process capability for each selected process area; an identification of the risk for the business in case of each selected process area still be performed at the current process capability.

## 5 PRO2PI experiences in software intensive organizations

The Table 1 summarizes experiences with PRO2PI utilizations from 1999 to 2005.

**Table 1 – Summary of PRO2PI utilizations**

Characteristic and year interval of the utilization	# of projects	# of uses
Software process improvement cycle in an organization [1999-2002]	1	3
Software process improvement cycle in another organization [2002-2003]	1	1
Establishment of process capability profiles to improvement [2000-2005]	9	12
15504MPE Project [2003-2004]	1	5
Process improvement in groups of organizations [2004-2005]	2	8
Development of more specific process capability models [2004-2005]	7	7
Students projects in professional SPI courses [2004-2005]	10	164
TOTAL	32	200

The fundamentals and elements of PRO2PI has been used from 1999 to 2005 in at least 32 (thirty-two) projects. In these projects, the defineP (define or update a PRO2PI) function has been used least 200 (two hundred) times [Salviano 2006], as summarized in Table 1. In 2006, about 20 (twenty) new utilizations are on the way.

The first utilization in Table 1 is a process improvement project, based on ISO/IEC TR 15504-5 and ISO 9001:2000 models in a low maturity product-oriented organization, without previous experience in software process improvement, performed from 1999 to 2002 [Nicoletti and Salviano 2003]. In this cycle a process capability profile with five process areas (CUS.4.2 Customer Support, SUP.3 Quality Assurance, MAN.2 Project Management, ORG.1 Organizational Alignment and ORG.2.1 Process Establishment), at capability level 2, was established after a careful study of the organization context and business goals. The engineering processes were considered as in a capability level 1 due to 10 years history of software development with four products in 100 of clients. A formal ISO/IEC 15504 conformant process assessment was performed in 1999, before the improvement. The result was a capability level 0, with process attribute PA1.1 partially achieved and PA2.1 and PA2.2 not achieved. A process improvement plan was established. The process improvement actions were performed by the organization, without external consultancy. In the middle of the project, the profile was updated to include the requirements of ISO 9001:2000. In 2002, after the improvement cycle, another formal ISO/IEC 15504 conformant assessment were performed for an extended process capability profile, with the inclusion of ENG.1.2 Software Requirement Analysis and ORG.5 Measurement processes. The result was a capability level 2 for most of the processes. An ISO 9001 certification was also achieved by the organization. An initial version of PRO2PI was used for this process improvement cycle. The major result from the organization perspective was the establishment of a management system to control the relevant processes. At the time of the second assessment, twelve sets of data related with software development, customization and maintenance processes and four sets related with customer support has been collected, analyzed and used to manage the organization.

The second utilization in Table 1 is a process improvement project based on ISO/IEC TR 15504-5 model in a very small project-oriented software organization [Silva et al. 2003]. After an elicitation and analysis of the organization business context, strategy and goals, the improvement focus was driven to the organization software factory process, composed by five phases: prospect, contract, development, deliver and close. Five 15504-5 process were selected as a process capability profile to guide the process improvement: CUS.2 Supply process (at capability level 2), to cover a macro view of the whole organization software factory process, ORG.5 Measurement process (at level 3), to consolidate measurement as the support given that the organization already have an unusual good foundation on using measurement, and a set with CUS.3 Requirement Elicitation process (at level 2), MAN.2 Project Management process (at level 2) and ENG.1.6 Software test process (at level 3) to cover a more detailed view of the organization software factory process. In addition to ISO/IEC TR 15504-5 model, three more references were selected for the improvement: RUP generic process,



because the organization already uses it, ProGer generic software project management process model for small organization [Roullier 2001], and the IEEE Std 829 [1998] for software test documentation. A formal ISO/IEC 15504 conformant process assessment was performed before the improvement actions. The result was a capability level 2 for measurement process and a strong capability level 1 for the other four processes. A process improvement plan was established. The process improvement actions were performed by the organization with external consultancy.

The third utilization in Table 1 is a set of nine projects to establish process capability profile for process improvement in nine different organizations, from 2000 to 2005. For each organization, a specific process capability profile, based on the specific business goals and context of each organization, was established using elements of four reference models: SW-CMM (projects 2 and 5), ISO/IEC TR 15504-5 (projects 1, 2, 3, 5, 6, 7, 8 and 9), CMMI-SE/SW (projects 4 and 9) and ISO 9001 (project 4). In five of these projects, the author of PRO2PI method participated in the establishment, and in four of them, the author did not participated. These experiences are described in Salviano [2006].

The fourth utilization in Table 1 is the development of 15504MPE Project from 2003 to 2004 [von Wangenheim et al. 2006]. In 15504MPE, a process capability model and a process assessment method were developed to using ISO/IEC 15504 for process improvement in small software organization. Based on PRO2PI, a contextualization phase is performed to study the overall situation of the organization's business and improvement goals and its software processes. As a result of this phase, a process capability profile to process improvement is defined. From 2003 to 2004, five projects, in five different organizations, were performed, with the establishment of five different profiles.

The fifth utilization in Table 1 is in process improvement in groups of organizations from 2004 to 2005. In two different projects, eight organizations shared activities for process improvement actions oriented by CMMI-SE/SW's maturity level 2. PRO2PI-WORK was used to study the eighteen process areas from maturity levels 2 and 3 as a mean to introduce them to each organization and to prioritize the process improvement actions from an analysis of the importance and risk for each process area to each organization business.

The sixth utilization in Table 1 is the development of seven more specific process capability models from 2004 to 2005. One project was the development of a process capability maturity model for Brazilian small and medium software organization in the MPS-BR initiative [Weber et al. 2005]. In addition to the main idea to developed a more specific model, the utilization of PRO2PI oriented three suggestions: (i) to use the maturity levels of CMMI as the basis for the model, (ii) to include additional maturity levels between level 1 and 2 and between levels 2 and 3 in the model, and (ii) to define the model as an ISO/IEC 15504-conformant model. The first suggestion was based on the dissemination and domination of SW-CMM and CMMI staged models in Brazilian software industry, the second suggestion was based on the opportunity to give smaller steps for small and medium organizations, and the third one was to be aligned with an international framework for the development of process capability models and process assessment processes.

The seventh utilization in Table 1 is the development of 164 articles, each one describing the establishment of a process capability profile for a specific organizational unit, as a course project in 10 editions of 40 hours of a software process improvement course by 261 students from 2004 to 2005. Each group of one to three students used the PRO2PI-WORK method to guide the development of an article covering the identification a an organizational unit and a suggestion of a process capability profile to process improvement, using references from CMMI-SE/SW, ISO/IEC 15504-5 and/or MPS.BR models.

## 6 Conclusion

This article presented a proposal towards a Process Capability Profile Driven Process Engineering (PCDE) as an evolution of the current Software Process Improved based on Process Capability (and Maturity) Models and an exemplar approach for this process engineering. Many versions of what is now the PRO2PI approach has been used in 200 projects since 1999 [Salviano 2006]. These

experiences are related with full process improvement cycle, only with the initiation and preparation of an improvement cycle, and the definition of more specific models. PCDE and PRO2PI have been formalized recently [Salviano 2006].

From the best of our knowledge the proposed process engineering PCDE and PRO2PI approach are innovative work. The closest related works to PRO2PI are the methodology SEI Toolkit for using CMMI continuous in small setting [Garcia et al. 2004], the Olson orientation for using staged and continuous representation of CMMI models [Olson 2003] and the “constagedeous” approach [Kasse 2004] to use a combination of staged and continuous CMMI models. PRO2PI goes beyond that using references from any model and supporting a process capability profile as a model for a process.

The proposed process engineering and the PRO2PI approach aim to guide the software intensive industry, specially small software intensive organizations, in the establishment of relevant, feasible, opportunistic, systemic, representative, traceable, specific and dynamic process capability profiles to drive more innovative and successful business oriented process improvements cycles, as an evolution of the current process improvement approaches composed by the implementation of the fixed “one size fits all” maturity levels of SW-CMM and CMMI models.

### **Acknowledgments**

The authors would like to thank all the people from the SPI projects reported in this article and from CenPRA for their comments and suggestions. Early work in this research has been supported by a grant from CNPq, an entity of the Brazilian Government directed to scientific and technological development.

### **Literature**

- [Automotive SIG 2005] Automotive SIG, Automotive SPICE Process Assessment Model, Final Release, v2.2, 144 pages, © The SPICE User Group, 21 August 2005 (available from <http://www.automotivespice.com>, last accessed 09/11/2005).
- [Bézivin 2003] Jean Bézivin, MDA™ : From Hype to Hope and Reality, Slides from a guess talk presentation at UML'2003, San Francisco, 96 slides, 2003 (available at: [www.sciences.univ-nantes.fr/info/perso/permanents/bezivin/UML.2003/UML.SF.JB.GT.ppt](http://www.sciences.univ-nantes.fr/info/perso/permanents/bezivin/UML.2003/UML.SF.JB.GT.ppt), last accessed in 27/07/2004)
- [Card 2004] David N. Card, Research Directions in Software Process Improvement, Proceedings of 28th International Computer Software and Applications Conference (COMPSAC 2004), Hong Kong, China, IEEE Computer Society, p. 238, September 27-30, 2004.
- [Chrissis et al. 2003] Mary Beth Chrissis, Mike Konrad and Sandy Shrum, CMMI: Guidelines for Process Integration and Product Improvement, Addison-Wesley Pub Co, 2003.
- [Conradi and Fuggetta 2002] Reidar Conradi and Alfonso Fuggetta, Improving Software Process Improvement, IEEE Software, 19(4), p. 92-99, July/August 2002.
- [Favre 2005] Jean-Marie Favre, Foundations of Model (Driven) (Reverse) Engineering: Models, Episode I: Stories of The Fidus Papyrus and of The Solarus, 31 pages, Pilot of the series “From Ancient Egypt to Model Driven Engineering”, available at <http://www-adele.imag.fr/mda>, 2005. (last accessed in 20/03/2006)
- [Garcia et al. 2004] Suzane Garcia, Sandra Cepeda, Gene Miluk and Mary Jo Staley, Adopting CMMI for Small Organizations, slides from presentation at Fourth Annual CMMI Technology Conference and Users Group, Denver, USA, November 2004. (available at <http://www.dtic.mil/ndia/2004/CMMIT2Mon/110504Cepeda.pdf>, last accessed in 17/02/2005)
- [Ibrahim 2000] Linda Ibrahim, Using an Integrated Capability Maturity Model® – The FAA Experience, in Proceedings of the Tenth Annual International Symposium of the International Council on Systems Engineering (INCOSE), Minneapolis, Minnesota, pp. 643-648, July 2000.
- [ISO/IEC 15504-2 2003] The International Organization for Standardization and the International Electrotechnical Commission, ISO/IEC 15504 - Information Technology - Process Assessment – Part 2, 2003.

## Session 12: SPI and Development Processes

- [IEEE STD 839 1998] The Institute of Electrical and Electronics Engineers, IEEE STD 829: Standard for Software Test Documentation, IEEE Computer Society, September 1998.
- [ISO/IEC 15504-5 2006] The International Organization for Standardization and the International Electrotechnical Commission, ISO/IEC 15504 - Information Technology - Process Assessment – Part 5: An exemplar Process Assessment Model, 2006.
- [Kasse 2004] Tim Kasse, Practical Insight into CMMI, Artech House Publishers, 281 pages, 2004.
- [Nicoletti and Salviano 2003] Adilson S. Nicoletti and Clênio F. Salviano, An Experience using ISO/IEC TR 15504 and ISO 9000:2000 for Software Process Improvement, Proceedings of SPICE 2003 The Third International SPICE Conference, Nordwick, Netherlands, pp. 141-142, March 2003.
- [Olson 2003] Timothy G. Olson, Staged or Continuous: Which Model Should I Choose?, slides from presentation at Third Annual CMMI Technology Conference and Users Group, USA, November 2003.
- [Paulk et al. 1994] Mark C. Paulk, Charles V. Weber, Bill Curtis and Mary Beth Chrissis, The Capability Maturity Model - Guidelines for Improving the Software Process, CMU-SEI, Addison-Wesley, 441 pages, 1994.
- [Potts 1993] Colin Potts, "Software-Engineering Research Revised", IEEE Software, Volume 10, Number 5, pages 19-28, September 1993.
- [Rifkin 2002] Stan Rifkin, Is process improvement irrelevant to produce new era software? in Software Quality - ECSQ 2002, Lecture Notes in Computer Science 2349, ed. by Jyrki Kontio and Reidar Conradi, pp. 13-16, Springer-Verlag, 2002.
- [Salviano and Jino 2004] Clênio F. Salviano and Mario Jino, Using Continuous Modes as "Dynamic and Specific Staged Models", slides from presentation at Fourth Annual CMMI Technology Conference and Users Group, Denver, USA, November 2004. (available at [http://www.dtic.mil/ndia/2004/CMMIT1Tue/1114ClenioSalviano\\_new.pdf](http://www.dtic.mil/ndia/2004/CMMIT1Tue/1114ClenioSalviano_new.pdf), last accessed in 12/10/2005)
- [Salviano et al. 2004] Clênio F. Salviano, Mario Jino and Manuel de Jesus Mendes, Towards an ISO/IEC 15504-Based Process Capability Profile Methodology for Process Improvement (PRO2PI), in Proceedings of SPICE 2004 The Fourth International SPICE Conference, Lisbon, Portugal, p. 77-84, April 28-29, 2004.
- [Salviano 2006] Clênio F. Salviano, Uma proposta orientada a perfis de capacidade de processo para evolução da melhoria de processo de software (in Portuguese), PhD thesis, Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas (FEEC-Unicamp), 2006.
- [Roullier 2001] Ana Cristina Roullier, Gerenciamento de Projetos de Software para Empresas de Pequeno Porte (in Portuguese), PhD thesis, Universidade Federal de Pernambuco (UFPE), 2001.
- [searchCRM 2006] searchCRM, The searchCRM.com website, 2006. (available at <http://searchcrm.techtarget.com>, last accessed in 30/03/2006)
- [Silva et al. 2003] Odair J. da Silva, Carlos A. Borges, Clênio F. Salviano, Ana L. Sampaio, Adalberto N. Crespo and Ana C. Roullier, An ISO/IEC 15504-Based Software Process Improvement Project in a Small Brazilian Software Organization, in Proceedings of SPICE 2003, The Third International SPICE Conference, 2003.
- [Stallinger et al. 2002] F. Stallinger, A. Dorling, T. Rout, B. Henderson-Sellers and B. Lefever Software Process Improvement for Component-Based Software Engineering: An Introduction to the OOSPICE Project, Proceedings of the 28th EUROMICRO Conference, Germany, IEEE Computer Society, 2002.
- [von Wangenheim et al. 2005] Christiane G. von Wangenheim, Timo Varkoi and Clenio F. Salviano, Performing ISO/IEC 15504 Conformant Software Process Assessments in Small Software Companies, experience report at EUROSPI, Budapest, Hungria, 2005.
- [von Wangenheim et al. 2006] Christiane G. von Wangenheim, Alessandra Anacleto and Clenio F. Salviano. "Helping Small Companies Assess Software Processes," IEEE Software, vol. 23, no. 1, pp. 91-98, January/February, 2006.
- [Weber et al. 2005] Kival C. Weber, Eratóstenes E. R. Araújo, Ana Regina C. da Rocha, Cristina A. F. Machado, Danilo Scalet and Clênio F. Salviano, Brazilian Software Process Reference Model and Assessment Method, in Proceedings of The 20th International Symposium on Computer and Information Sciences - ISCIS'05, October 26-28th, Instambul, Turkey, 2005.

### **Author CVs**

#### **Clenio F. Salviano**

Clenio F. Salviano is a researcher at the “Renato Archer” Research Center (CenPRA) in Campinas, Brazil since 1984 and a professor in software process improvement in professional courses at UFLA, SENAC and UNIMEP Universities. At CenPRA, he coordinates a software process improvement division, doing technological research and consultancy in software industry. He is co-editor of ISO/IEC 15504-5, the exemplar process assessment model for ISO/IEC 15504. His research interests include software process capability models, process improvement, process assessment and software patterns. He received his PhD at Unicamp (State University of Campinas) in 2006 with a thesis on a process capability profile driven proposal for software process improvement. Contact him at CenPRA, Rodovia D. Pedro I (SP/65), Km 143,6, CEP 13069-901, Campinas, SP, Brazil or by e-mail [clenio.salviano@cenpra.gov.br](mailto:clenio.salviano@cenpra.gov.br).

#### **Mario Jino**

Mario Jino is a professor at the Department of Computer Engineering and Industrial Automation (DCA) of School of Electrical and Computer Engineering (FEEC) at State University of Campinas (Unicamp). Contact him at Unicamp, Cidade Universitária “Zeferino Vaz”, CEP 13081-970, Campinas, SP – Brazil or by e-mail [jino@dca.fee.unicamp.br](mailto:jino@dca.fee.unicamp.br).

