Rory V. O'Connor
Jan Pries-Heje
Richard Messnarz (Eds.)

Communications in Computer and Information Science 172

Systems, Software and Services Process Improvement

18th European Conference, EuroSPI 2011
Roskilde, Denmark, June 2011
Proceedings

Springer

Communications
in Computer and Information Science 172

Rory V. O'Connor Jan Pries-Heje
Richard Messnarz (Eds.)

Systems, Software and Services Process Improvement

18th European Conference, EuroSPI 2011
Roskilde, Denmark, June 27-29, 2011
Proceedings

Springer

Volume Editors

Rory V. O'Connor
Dublin City University
Lero, The Irish Software Engineering Research Centre
Dublin, Ireland
E-mail: roconnor@computing.dcu.ie

Jan Pries-Heje
Roskilde University, Denmark
E-mail: janph@ruc.dk

Richard Messnarz
ISCN Ltd.
Graz, Austria
E-mail: rmess@iscn.com

ISSN 1865-0929 e-ISSN 1865-0937
ISBN 978-3-642-22205-4 e-ISBN 978-3-642-22206-1
DOI 10.1007/978-3-642-22206-1
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2011930517

CR Subject Classification (1998): D.2, K.6, H.4, F.3, C.2, D.3, H.5, J.1

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

Systems, Software and Services Process Improvement – 18th European Conference, EuroSPI[2] 2011, June 27–29, 2011

This textbook comprises the proceedings of the 18th EuroSPI Conference, held during June 27-29, 2011 in Roskilde, Denmark.

Since EuroSPI 2010 we extended the scope of the conference from software process improvement to systems, software and service-based process improvement. EMIRAcle is the institution for research in manufacturing and innovation, which came out as a result of the largest network of excellence for innovation in manufacturing in Europe. EMIRAcle key representatives joined the EuroSPI community, and papers as well as case studies for process improvement on systems and product level will be included in future.

Since 2008 EuroSPI partners have packaged SPI knowledge in job role training and established a European certification association (www.ecqa.org) to transport this knowledge Europe-wide using standardized certification and exam processes.

Another addition in to Roskilde 2011 conference was that results from the Danish SourceIT conference were presented in two sessions at the conference.

Conferences were held in Dublin (Ireland) in 1994, in Vienna (Austria) in 1995, in Budapest (Hungary) in 1997, in Gothenburg (Sweden) in 1998, in Pori (Finland) in 1999, in Copenhagen (Denmark) in 2000, in Limerick (Ireland) in 2001, in Nuremberg (Germany) in 2002, in Graz (Austria) in 2003, in Trondheim (Norway) in 2004, in Budapest (Hungary) in 2005, in Joensuu (Finland) in 2006, in Potsdam (Germany) in 2007, in Dublin (Ireland) in 2008, in Alcala (Spain) in 2009, and in Grenoble (France) in 2010.

EuroSPI is an initiative with the following major action lines http://www.eurospi.net:

- Establishing an annual EuroSPI conference supported by Software Process Improvement networks from different EU countries.
- Establishing an Internet-based knowledge library, newsletters, and a set of proceedings and recommended books.
- Establishing an effective team of national representatives (from each EU country) growing step by step into more countries of Europe.
- Establishing a European Qualification Framework for a pool of professions related with SPI and management. This is supported by European certificates and examination systems.

EuroSPI has established a newsletter series (newsletter.eurospi.net), the SPI Manifesto (SPI = Systems, Software and Services Process Improvement), an experience library (library.eurospi.net) to be continuously extended over the years and made available to all attendees, and a Europe-wide certification for qualifications in the SPI area (www.ecqa.org, European Certification and Qualification Association).

A typical characterization of EuroSPI is reflected in a statement made by a company: *"... the biggest value of EuroSPI lies in its function as a European knowledge and experience exchange mechanism for SPI and innovation.*"

Since its beginning in 1994 in Dublin, the EuroSPI initiative has outlined that there is no single silver bullet with which to solve SPI issues, but that you need to understand a combination of different SPI methods and approaches to achieve concrete benefits. Therefore each proceedings volume covers a variety of different topics, and at the conference we discuss potential synergies and the combined use of such methods and approaches. These proceedings contain selected research papers for six topics:

- Section I: SPI and Assessment
- Section II: SPI and Implementation
- Section III: SPI and Improvement Methods
- Section IV: SPI and Organization
- Section V: SPI and People/Teams
- Section VI: SPI and Reuse
- Section VII: Selected Key Notes for SPI Implementation

Section I presents studies on SPI and assessment. The authors provide different insights and additions into the assessment process. Mejia et al. present a multi-model workflow for assessing the solicitation and supplier agreement development process area of the CMMI-ACQ model. Mesquida and Pichaco look at best practices for IT service management. And Kasurinen et al. describe a self-assessment framework to be used with the new ISO/IEC 29119 test standard where the first results of use indicate that it is a very viable approach especially when combined with a maturity level- based approach.

Section II presents three papers on implementation issues in relation to SPI. First, Jäntti et al. use a case study to explain how to improve the deployment of IT service management processes. The explanation is that one should focus on understanding and training as well as dividing the implementation into smaller phases and milestones. Second, Kuhrman et al. provide insight into the usage style, ratings, and tempers of project managers working with a newer German government standard software development process for IT projects. Third, Sivakumar et al. present an approach improving verification and validation in the medical device domain.

Section III presents three papers more specifically dealing with implementation methods. Clarke and O'Connor look at the motivation for conducting SPI by providing further evidence of its positive impact. More specifically, they present a holistic scorecard (HSC) that can be used to examine business success systematically. Then Stettina and Heijstek propose a five-dimensional tool to foster self-reflection in agile software development teams. This paper also provides an account of using the tool with 79 individuals and 8 international Scrum teams showing that the tool is quite useful. Finally, Aysolmaz and Demirörs present an

SPI methodology with many details on life cycle, tasks, approaches, resources, tools, roles, participation of groups, and process assets, which has been tested in 10 organizations.

Section IV presents studies on SPI in relation to organizations. Lepmets et al. describe the results of an international survey showing that process assessment contributes positively to the internalization of process improvement goals. Neumann et al. investigate the importance of idea generation and idea sources in relation to innovation management for an organization. The investigation is based on a case study showing that more, and better, idea sources can boost innovation. Polgár and Biró describe the application of usability methodology for software process improvement and formulate specific ideas on how to adapt concrete usability improvement methods.

Section V presents studies on people and teams in relation to SPI. First, Basri and O'Connor explore the dynamics of software development teams – such as structure, process, communication, learning and sharing—and its impact on SPI. Second, Yilmaz and O'Connor use structural equation modelling for an empirical investigation of productivity enabling social factors in the software process. Third, Ringstad et al. argue for the use of diagnosis and action planning to improve teamwork in agile software development.

Section VI presents three papers on SPI and reuse. Valdes et al. describe a reusable process model called Tutelkan for enabling SPI in small settings. O'Leary and Richardson show a process model for product derivation coming out of several iterative development cycles and evaluated with both academic and industrial sources. Finally, Leitner and Kreiner investigate whether flexible product architecture conceptually is the same as flexible product line (PL) architecture. As a result they define what they call an 'agile continuum' emphasizing that there is no clear point in time when the product line process is finished and the product life cycle starts

Section VII presents selected key notes from EuroSPI workshops concerning the future of SPI. From 2010 onwards EuroSPI invites recognized key researchers to publish new future directions of SPI.

Four invited papers illustrate that SPI can beneficially be implemented in very small organizations. Caballero et al. discuss how SCRUM can be implemented in a small SME. O'Connor and Laporte illustrate how ISO/IEC 29110 can be used to support the improvement needs of VSEs, while Mas and Mesquida present a tool to manage SPI in SMEs. Finally, McCaffery et al. provide a practical case study from the medical device sector.

Further invited papers illustrate that SPI has a direct impact on the innovation competencies of an organization. SPI helps to create continuous learning organizations. Kishida describes how SPI can help form innovative software projects, Messnarz et al. describe a future vision of SPI and innovation networking strategies in Europe, and Riel aims at pinpointing new innovation management challenges that have evolved in product development and manufacturing industries.

Two invited papers discuss how the new functional safety standards influence the longer standing SPI initiatives and how the existing paradigms have to be extended to cover functional safety aspects as well. Ovi Bachmann et al. illustrate the implementation on a case study in automotive industry, and Messnarz et al. show step by step how an ISO 15504-based improvement program is extended towards covering functional safety concepts.

Recommended Further Reading

In [1] the proceedings of three EuroSPI^2 conferences were integrated into one book edited by 30 experts in Europe. The proceedings of EuroSPI^2 2005, 2006, 2007, 2008, and 2009 have been published by Springer in [2], [3], [4], [5], [6] and [7], respectively.

June 2011

Rory V. O'Connor
Jan Pries-Heje
Richard Messnarz

References

1. Messnarz, R., Tully, C. (eds.): Better Software Practice for Business Benefit – Principles and Experience, 409 pages. IEEE Computer Society Press, Los Alamitos (1999)
2. Richardson, I., Abrahamsson, P., Messnarz, R. (eds.): Software Process Improvement. LNCS, vol. 3792, p. 213. Springer, Heidelberg (2005)
3. Richardson, I., Runeson, P., Messnarz, R. (eds.): Software Process Improvement. LNCS, vol. 4257, pp. 11–13. Springer, Heidelberg (2006)
4. Abrahamsson, P., Baddoo, N., Margaria, T., Messnarz, R. (eds.): Software Process Improvement. LNCS, vol. 4764, pp. 1–6. Springer, Heidelberg (2007)
5. O'Connor, R.V., Baddoo, N., Smolander, K., Messnarz, R. (eds): Software Process Improvement. CCIS, vol. 16, Springer, Heidelberg (2008).
6. O'Connor, R.V., Baddoo, N., Gallego C., Rejas Muslera R., Smolander, K., Messnarz, R. (eds): Software Process Improvement. CCIS, vol. 42, Springer, Heidelberg (2009).
7. Riel A., O'Connor, R.V. Tichkiewitch S., Messnarz, R. (eds): Software, System, and Service Process Improvement. CCIS, vol. 99, Springer, Heidelberg (2010).

Organization

Board Members

EuroSPI Board Members represent centers or networks of SPI excellence having extensive experience with SPI. The board members collaborate with different European SPINS (Software Process Improvement Networks). The following six organizations have been members of the conference board for the last 10 years:

- 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

EuroSPI Scientific Program Committee

EuroSPI established an international committee of selected well-known experts in SPI who are willing to be mentioned in the program and to review a set of papers each year. The list below represents the Research Program Committee members. EuroSPI² also has a separate Industrial Program Committee responsible for the industry/experience contributions.

Ambriola Vincenzo	Università di Pisa, Italy
Biffl Stefan	University of Technology Vienna, Austria
Biró Miklós	Dennis Gabor College, Hungary
Buglione Luigi	Engineering Ingegneria Informatica S.p.A., Italy
Calvo-Manzano Villalon José Antonio	Universidad Politecnica de Madrid, Spain
Casey Valentine	Dundalk Institute of Technology, Ireland
Chua Bee Bee	University of Technology Sydney, Australia
Ciolkowski Marcus	Fraunhofer IESE, Germany
Clarke Paul	Lero, Irish Software Engineering Research Centre, Ireland
Dalcher Darren	Middlesex University, UK
De Amescua Seco Socio Antonio	Carlos III University of Madrid, Spain
Dingsøyr Torgeir	SINTEF ICT, Norway
García Rubio Felix Oscar	University of Castilla-La Mancha, Spain
García-Guzman Javier	Carlos III University of Madrid, Spain

Gorschek Tony	Blekinge Institute of Technology, Sweden
Gresse von Wangenheim Christiane	Federal University of Santa Catarina, Brazil
Kreiner Christian	Graz University of Technology, Austria
Landes Dieter	Fachhochschule Coburg, Germany
Lepmets Marion	CRP Henri Tudor, Luxembourg
Mäkinen Timo	Tampere University of Technology, Finland
Mas Pichaco Antonia	Universitat de les Illes Balears, Spain
McCaffery Fergal	Dundalk Institute of Technology, Ireland
McQuaid Patricia	California Polytechnic State University, USA
Münch Jürgen	Fraunhofer IESE, Germany
O'Connor Rory V.	Dublin City University, Ireland
Phalp Keith	Bournemouth University, UK
Richardson Ita	Lero, Irish Software Engineering Research Centre, Ireland
Riel Andreas	Grenoble Institute of Technology, Laboratory G-SCOP, France
Sillitti Alberto	Free University of Bolzano, Italy
Smolander Kari	Lappeenranta University of Technology, Finland
Stapel Kai	Leibniz Universität Hannover, Germany
Tichkiewitch Serge	Grenoble Institute of Technology, France
Vajde Horvat Romana	proHUMAN Ltd., Slovenia
Ventura Martins Paula	FCT University of Algarve, Portugal
Vondrak Ivo	VSB - Technical University of Ostrava, Czech Republic

General Chair

Richard Messnarz

Scientific Chairs

Rory V. O'Connor
Jan Pries-Heje

All three Chairs, the General and the Research Chairs, have quite a complementary and interesting profile. Dr. Messnarz works in close collaboration with Austrian research institutions (universities of applied sciences) and large German automotive companies.

Dr. Rory O'Connor is a senior lecturer in Dublin City University and a senior researcher with Lero, the Irish Software Engineering Centre. His main research interests center on software processes and SPI in relation to small and very small organizations. Jan Pries-Heje is Professor in Information Systems at Roskilde

University. He is past President of the Association of Information Systems in Scandinavia (IRIS). Jan serves as the Danish National Representative to IFIP Technical Committee 8 on Information Systems where he is also Vice-Chair. Jan is currently Associate Editor for *MIS Quarterly, Information Systems Journal, and Business and Information Systems*; three of the best journals in the field of IS.

The experience portfolio of the Chairs covers different market segments, different sizes of organizations, and different SPI approaches. This strengthens the fundamental principle of EuroSPI^2 to cover a variety of different markets, experiences, and approaches.

Table of Contents

SPI and Assessment

SPI and Implementation

SPI and Improvement Methods

SPI and Organization

SPI and People/Teams

SPI and Reuse

Selected Key Notes for SPI Implementation

SPI in SMEs

SPI and Innovation

SPI and Functional Safety

A Multi-model Workflow before Establishing an Acquisition Contract Based on CMMI-ACQ Model*

Jezreel Mejia[1], Mirna Muñoz[1], Jose A. Calvo-Manzano[2], Gonzalo Cuevas[2], and Tomás San Feliu[2]

[1] Instituto Tecnológico de Orizaba, av. Ote 9 no. 582, 94300 Veracruz, México
{jez.mejia,mirna.munozm}@gmail.com
[2] Universidad Politécnica de Madrid,
Facultad de Informática Campus de Montegancedo S/N, 28660 Madrid, Spain
{joseantonio.calvomanzano,gonzalo.cuevas,tomas.sanfeliu}@upm.es

Abstract. CMMI-ACQ is a model that provides guidance to organizations on the acquisition of software products and related services. This paper addresses an organizational proposal that establishes a multi-model workflow in order to implement specific practices of the Solicitation and Supplier Agreement Development (SSAD) process area of CMMI-ACQ model. To achieve this, the Organization of the Acquisition Processes (OAP) procedure is implemented. OAP have helped to determine the implementation sequences of the acquisition processes areas at maturity level 2. Once the sequence is obtained, the explicit dependencies and prioritization of the practices within the SSAD process area are analyzed, in order to establish the workflow and its multi-model environment.

Keywords: CMMI-ACQ, Solicitation and Supplier Agreement Development, Multi-model.

1 Introduction

Outsourcing is the organizational decision to turn over part or all the process related to the development, maintenance or exploitation of systems. The Information Technology (IT) services outsourcing market has grown rapidly every year. Outsourcing as a concept was accepted in the 1980s and is still used today to describe “a contractual relationship with a specialized outside service provider for work traditionally done in-house” [8]. Today it is growing rapidly worldwide and in recent years has gained much importance and this market continues to grow each year [3-8]. However, while the outsourcing is experienced a considerable growth, the number of reported cases of failure is also increasing [14]. According to a study [2, 3], 20 to 25 percent of large information technology (IT) acquisition projects fail within two years and 50 percent fail within five years. The case of Southern Pacific Transportation Co. described by [8] is an example.

* Authors are ordered by Organization.

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 1–13, 2011.

Mismanagement, the inability to articulate customer needs, poor requirements definition, inadequate provider selection and contracting processes, uncontrolled requirements changes and important gaps in the contracts are factors that contribute to project failure [2, 3].

The majority of project failures could be avoided if the acquirer learns how to properly prepare the contracts with providers [2, 3, 10, 11].

According to an analysis of 40 organizations from different sectors [12, 13], problems encountered with outsourcing contracts prior to renegotiation often stem from misunderstandings between the acquirer and the service supplier. In addition, the organizations cited that a major impediment to a higher degree of success in IT outsourcing projects is the lack of knowledge or expertise in the development and structuring of the initial acquisition or outsourcing agreement or contract with a supplier.

The CMMI for Acquisition model (CMMI-ACQ) provides a framework to facilitate the outsourcing strategies adoption, eliminating the existing barriers among the relevant stakeholders (service supplier, business departments, system areas, etc) [3, 14]. Moreover, it refers to the establishment of the agreement or contract with the supplier in the Solicitation and Supplier Agreement Development process area.

The purpose of this paper is to show the establishment of the multi-model workflow of Solicitation and Supplier Agreements Development (SSAD) process area. To achieve this, it is necessary to establish: 1) process maps that represent the existing dependencies among CMMI-ACQ processes area and, 2) multi-model environment to perform a best practices mapping.

This paper is organized as follows. Section 2 gives a brief description on CMMI-ACQ v1.3 model and SSAD process area, section 3 describes the procedure for establishing the implementation sequence related to the CMMI-ACQ process areas, section 4 addresses the establishment of the main workflow for the SSAD process area, in section 5 the multi-model environment is established and finally section 6 presents the conclusions.

2 CMMI for Acquisition Model

CMMI-ACQ is a model that provides guidance for acquisition organizations to initiate and manage the acquisition of software products and related services. This model focuses on acquirer processes and integrates bodies of knowledge that are essential for successful acquisitions. CMMI-ACQ provides an opportunity for acquisition organizations [3]:

- to prevent or eliminate barriers and problems in the acquisition process through improved operational efficiencies.
- to initiate and manage a process for acquiring products and services, including solicitations, supplier sourcing, supplier agreement development, and supplier capability management.
- to use a common language for both acquirers and suppliers so that quality solutions are delivered more quickly and at a lower cost using the most appropriate technology.

CMMI-ACQ supports two approaches: Continuous and Stage representations [2].

- The Continuous Representation has capability levels (CL) which enable organizations to improve an individual process area selected by the organization.
- The Stage Representation has maturity levels (ML) which enable organ izations to improve a set of related processes.

CMMI-ACQ contains 22 process areas, six process areas focus on practices specific to acquisition, addressing agreement management (AM, Maturity level (ML) 2), acquisition requirements development (ARD, ML 2), acquisition technical management (ATM, ML 3), acquisition validation (AVAL. ML 3), acquisition verification (AVER, ML 3), and solicitation and supplier agreement development (SSAD, ML 2). 16 are core process areas that cover Process Management, Project Management, and Support categories [3] configuration management (CM, ML 2), Measurement and analysis (MA, ML2), Project monitoring and control (PMC, ML2), Project Planning (PP, ML 2), Process and Product Quality Assurance (PPQA, ML 2), Requirements Management (REQM, ML 2), Decision Analysis and Resolution (DAR, ML 3), Integrated Project Management (IPM, ML 3), Organizational Process Definition (OPD, ML 3), Organizational Process Focus (OPF, ML 3), Organizational Training (OT, ML 3), Risk Management (RSKM, ML 3), Organizational Process Performance (OPP, ML 4), Quantitative Project Management (QPM, ML 4), Causal Analysis and Resolution (CAR, ML 5), Organizational Performance Management, (OPM, ML 5) [3].

Solicitation and Supplier Agreement Development Process Area

The purpose of Solicitation and Supplier Agreement Development (SSAD) is to prepare a solicitation package, select one or more suppliers to deliver the product or service, and establish and maintain the supplier agreement. The SSAD specific goal and its associated specific practices identify potential suppliers and develop and distribute the solicitation package, including evaluation criteria and the statement of work. The specific goals (SG) and its specific practice (SP) are:

- SG 1 Prepare for Solicitation and Supplier Agreement Development: SP 1.1 Identify Potential Suppliers, SP 1.2 Establish a Solicitation Package, SP 1.3 Review the Solicitation Package, SP 1.4 Distribute and Maintain the Solicitation Package.
- SG 2 Select Suppliers: SP 2.1 Evaluate Proposed Solutions, SP 2.2 Establish Negotiation Plans, SP 2.3 Select Suppliers
- SG 3 Establish Supplier Agreements: SP 3.1 Establish an Understanding of the Agreement, SP 3.2 Establish the Supplier Agreement.

3 OAP: Procedure for Organizing the CMMI-ACQ Process Areas

This section introduces to the OAP (Organization of the Acquisition Processes), used for analyzing the existing dependencies among CMMI-ACQ processes areas to propose an implementation sequence. OAP is divided into three stages [18].

- Identify dependencies: A matrix of dependencies among PAs is elaborated.
- Analyze dependencies: Verify the Strongly connected components (SCC). The cyclic and SCC group are selected.
- Determine the Implementation sequence: the formal implementation sequence is proposed.

It is considered important to describe briefly the three stages and how the proposal sequences are obtained in order to show SSAD dependencies and its prioritization.

3.1 Identify Dependencies

The existing dependencies among process areas are identified by reviewing the new version of CMMI-ACQ v1.3 official bibliography [3], through the 22 process areas that the model includes. The analysis performed focused on the process areas related to maturity level 2, because it is the first maturity level introduced in an organization and SSAD process area is at maturity level 2.

The model components were analyzed to detect references in order to identify the existing dependencies [3, 18]. A matrix of dependencies is elaborated and updated from the dependencies found in the new version of CMMI-ACQ [3] (see Table 1). The rows represent the source process areas, the columns represent the destination process areas for the 9 process areas of maturity level 2 (AM, ARD, CM, MA, PMC, PP, PPQA, REQM and SSAD), the column TSDL2 is the Total Source Dependencies at Level 2 and is the sum of all dependencies for a process area.

Table 1. Matrix of dependencies

	Destination	AM	ARD	CM	MA	PMC	PP	PPQA	REQM	SSAD	TSDL2
Source	AM	0	0	1	1	1	1	1	0	0	5
	ARD	0	0	0	0	0	1	0	0	1	2
	CM	0	0	0	0	0	0	0	0	0	0
	MA	1	1	1	0	1	1	0	0	1	6
	PMC	1	0	1	0	0	1	0	0	1	4
	PP	1	1	1	0	1	0	0	0	1	5
	PPQA	0	0	0	0	0	0	0	0	0	0
	REQM	0	1	1	1	1	1	0	0	1	6
	SSAD	1	1	1	1	0	1	1	0	0	6

3.2 Analyze Dependencies

In the analysis stage, the strongly connected components (SCC) are verified and the cyclic and SCC groups are selected.

Verify the Strongly connected components

The dependencies in the digraph shown in Fig. 1 were evaluated using a mathematical software tool [10] in order to check the SCCs [1, 10]. After executing the mathematical software tool four SCC groups were obtained:

- Group1: {CM}, Group2: {PPQA}, Group3: {AM, MA, PMC, PP, ARD, SSAD} and Group4: {REQM}.

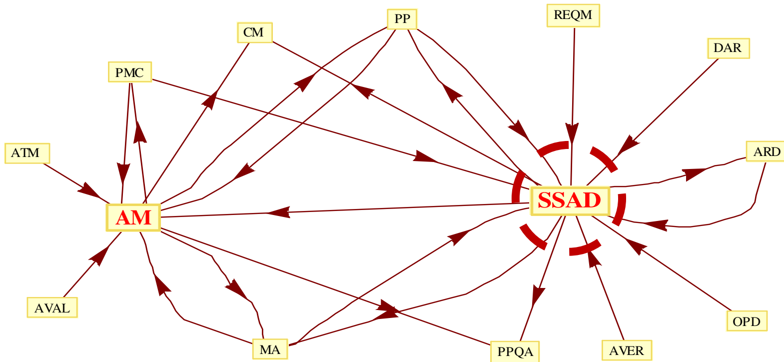


Fig. 1. Dependencies from SSAD Process Area

From the SCC groups obtained, the individual groups were discarded because of its triviality. Then, the SCC Group3 has been selected.

Generate combinations

The formula for combination is applied to the SCC Group 3 in order to get all the different combinations of 3 processes. As a result, 20 combinations were obtained (see Table 2).

Table 2. Combinations result

Group	Combinations AP	AP	AP	Group	Combinations AP	AP	AP
A	PP	ARD	AM	**K**	ARD	AM	MA
B	PP	ARD	MA	**L**	ARD	AM	PMC
C	PP	ARD	PMC	**M**	ARD	AM	SSAD
D	PP	ARD	SSAD	**N**	ARD	MA	PMC
E	PP	AM	MA	**Ñ**	ARD	MA	SSAD
F	PP	AM	PMC	**O**	ARD	PMC	SSAD
G	PP	AM	SSAD	**P**	AM	MA	PMC
H	PP	MA	PMC	**Q**	AM	MA	SSAD
I	PP	MA	SSAD	**R**	AM	PMC	SSAD
J	PP	PMC	SSAD	**S**	MA	PMC	SSAD

Cyclic groups

The cyclical groups are obtained (see Table 3) by applying the criteria in [18] to each combination from Table 2 [18]:

Table 3. Cyclical groups

Cyclical groups	Processes		
D	PP	ARD	SSAD
F	PP	AM	PMC

3.3 Implementation Sequences

According to the OAP, the implementation sequence cannot be implemented without the implementation of all processes that constitute the cyclical group [18]. In order to identify the processes implementation sequence, permutations for each cyclical groups has been generated from Table 3 (see Table 4). A permutation is sorted by higher to lower number of source dependencies, according to the TSDL2 column in Table 1.

Table 4. Implementation sequence alternatives

Permutation	Implementation sequence			Total Dependences by permutation
	1º	2º	3º	
F	AM (5)	PP (5)	PMC (4)	14
D	SSAD (6)	PP (5)	ARD (2)	13

Table 4 shows the two related implementation sequences alternatives, F and D permutations. Neither implementation sequences can be implemented without implementing all the processes that constitute the permutation [18].

- Permutation F: this implementation sequence is selected when there is an Outsourcing Agreement. The cyclical group F shows two possibilities, starting with the AM process area or the PP process area. Nevertheless, according to business goals, the implementation sequence would start with AM process area because Project Planning process area cannot be implemented without first updating the Agreement Management (AM). So, the implementation sequence would start with AM, then PP, and finally PMC.
- Permutation D: this implementation sequence is selected when there is no Outsourcing Agreement. The implementation sequence would start with SSAD, then PP, and finally ARD.

4 Establishing the Main Workflow from SSAD Process Area

Once the different implementation sequences are established, the two potential sequences are: 1) starting the implementation with SSAD process area or 2) starting with the AM process area according to the implementation sequences (see Table 4). This result confirmed the importance of the SSAD process area because this process area is necessary when there is no outsourcing agreement. Once this is confirmed, the workflow that includes the practices of SSAD process area and its dependences is established.

To establish the main workflow, the dependences from SSAD process area to the maturity level 2 process areas in Table 1 were analyzed, in order to, establish the main activities, the required activities and the support activities.

According to the SSAD process area dependencies in Table 1 and the digraph of Fig. 1 [15,16], Fig. 2 shows the activities sequence to be performed before establishing or developing the acquisition or outsourcing contract with the supplier, as well as the activities that are required before starting the main activities sequence and support activities that help to implement it:

Required Activities: the required activities showed in Fig. 2 indicate what the acquirer or client organization must establish or contemplate before starting the main activities sequence.

Main Activities: the main sequence of activities shows the activities that must be considered before signing the acquisition contract (see Fig. 2).

Support Activities: the support activities showed in Fig. 2 help to carry out activities in the main sequence.

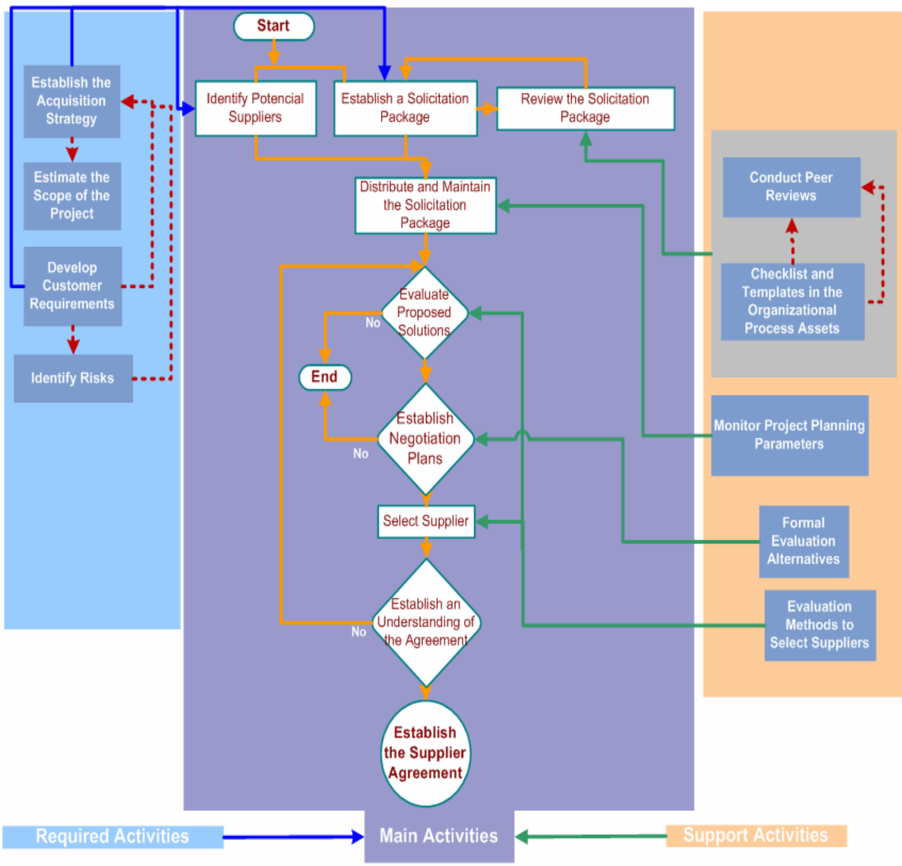


Fig. 2. Main workflow before signing the final agreement or contract

5 Mapping

This section introduces to the mapping used to establish the multimodal environment. The purpose of this mapping is to perform a best practices mapping using the outsourcing processes from different models and standards (IEEE 1062:1998 [19], PMBOK [20], ISO/IEC IEEE 12207:2008 [21], eSCM-CL [22]) to improve the previous main workflow (from section 4) and to establish a multi-model environment. To develop this mapping, the following steps were developed [17]: 1) Select the models and standards to be analyzed, 2) Choose the reference model, 3) Select the process to establish the study scope, 4) Establish the detail level, 5) Create a correspondence template, 6) Identify similarities among models, and 7) Show the obtained results.

As a result, the implementation of the 7 steps above, the CMMI-ACQ model was chosen as the reference model and the level of mapping at specific practice level was established because 1) a specific practice level can be found in all the models and standards analyzed, and 2) the specific practices help organizations in their process improvement.

A template was designed based on the CMMI-ACQ structure. The CMMI-ACQ structure includes example work products, subpractices and informative components. Other items, such as inputs, tools, and techniques, were added from the other models and standards. Then similarities among standards and models were identified.

The obtained results show templates of similarities among the activities in the main activities sequence with other models and standards. The templates contain basic information on how these activities could be strengthened by the other models and standards that have been analyzed. Tables 5 to 13 show the results for each main activity of the workflow in Fig. 2. These tables could be used as a best practices guide among models and standards.

Table 5. Template of "Identify Potential Suppliers"

Inputs	*Subpractices*	*Tools and techniques*	*Example work products*
• Internal organizational policy • Acquisition strategy • Project scope and requirements • Supplier criteria evaluation • Performance data of the historical contract supplier • Survey to users	• Develop a list of potential suppliers. • Communicate with potential suppliers concerning the forthcoming solicitation. • Verify participants who will evaluate supplier proposals. • Verify participants in supplier negotiations.	Not found.	• Develop a list of potential suppliers prepared to respond to the solicitation. • Software report. • Software developed or modified by suppliers. • User Surveys. • Evaluation criteria. • Statement of updated work.
Informative Components 1) Gather information on available software products, 2) Evaluate software during a demonstration, 3) Survey users of the supplier's software, 4) Review performance data from previous contracts, 5) Survey several suppliers offerings			

Table 6. Template of "Establish a Solicitation Package"

Inputs	*Subpractices*	*Tools and techniques*	*Example work products*
• Management acquisition plan. • Statements of Work. • Other plans. • Procurement documents. • List of qualified suppliers.	• Develop the statement of work for the supplier. • Specify the process, product, and service level measures for acceptance. • Develop supplier evaluation and proposal evaluation criteria. • Document the proposal content that suppliers must submit with their response. • Incorporate the acquirer's (standard) supplier agreement, terms and conditions, and additional information into the solicitation package.	• Use of standard formats. • Request of analysis and expert opinion in the area. • Advertising.	• Acquisition documents. • Evaluation criteria. • Updating of statements of work. • Evaluation criteria for proposals and suppliers. • Proposals.
Informative Components: Not found			

Table 7. Template of "Review the Solicitation Package"

Inputs	*Subpractices*	*Tools and techniques*	*Example work products*
• Solicitation package.	• Not found.	• Not found.	• Record of the reviews of the solicitation package.
Informative Components: Not found			

Table 8. Template of "Distribute and maintain the solicitation package"

Inputs	*Subpractices*	*Tools and techniques*	*Example work products*
• Solicitation package.	• Finalize a list of potential suppliers. • Distribute the solicitation package to potential suppliers. • Document and respond to supplier questions according to the instructions in the solicitation package. • Acknowledge the receipt of supplier proposals according to the schedule identified in the solicitation package. • Verify conformance to requirements and completeness of supplier responses. • Communicate with suppliers if the answer is incorrect or incomplete and that they can take corrective action. • Issue amendments to the solicitation package when changes are made to the solicitation.	• Not found.	• Responses to supplier questions. • Amendments to the solicitation package. • Supplier proposals. • Supplier questions and requests for clarification.
Informative Components: Not found			

Table 9. Template of "Evaluate proposed solutions"

Inputs	*Subpractices*	*Tools and techniques*	*Example work products*
• Supplier proposals. • Proposal evaluation standards. • Supplier qualification and selection process.	• Distribute supplier proposals to individuals identified by the acquirer to perform the evaluation. • Schedule an acquirer evaluation review of supplier proposals to consolidate questions, concerns, and issues. • Schedule supplier presentations. • Confirm the mutual understanding of the statement of work. • Evaluate supplier proposals and document findings. • Execute due diligence. • Document candidate supplier recommendations based on the proposal evaluation.	• Not found.	• Clarification correspondence between the acquirer and potential supplier. • Evaluation results and rationale. • List of candidate suppliers. • Proposal revisions based on clarification. • Supplier documentation of their approach to the project work, their capabilities, and a preliminary technical solution. • Evaluation of proposals. • Evaluation of suppliers. • Qualified suppliers list. • Supplier selection.
Informative Components 1) Evaluate supplier proposals, 2) Visit supplier facilities.			

Table 10. Template of "Establish Negotiation Plans"

Inputs	*Subpractices*	*Tools and techniques*	*Example work products*
• Evaluation of proposals. • Evaluation of suppliers.	• Not found.	• No found.	• Negotiation plan for each candidate supplier. • Negotiated contract.
Informative Components: Not found			

Table 11. Template of *"Select Supplier"*

Inputs	*Subpractices*	*Tools and techniques*	*Example work products*
• Visit supplier facilities. • User survey results. • Proposals. • Evaluation criteria.	• Evaluate supplier proposals. • Negotiate with suppliers to determine the best fit for the project. • Select a supplier to be awarded the supplier agreement. • Document the selection.	• Contract negotiation. • Independent estimates.	• Revisions due to negotiations. • Supplier selection decision. • Evaluation reports. • Contract.
Informative Components • Negotiate the contract.			

Table 12. Template of "Establish an Understanding of the Agreement"

Inputs	*Subpractices*	*Tools and techniques*	*Example work products*
• Organizational policies.	• Establish the supplier agreement. • Verify that all parties to the agreement understand and agree to all requirements by signing the supplier agreement. • Notify those suppliers not selected for the award. • "Communicate the supplier agreement in the organization as required". • Maintain the supplier agreement as required. • "Ensure that all records related to the supplier agreement are stored, managed, and controlled for future use".	• Not found.	• Criteria for evaluation and testing. • Linking payments to deliverables. • Review legal. • Prepared Contract.
Informative Components:1) Determine the quality of the work, 2) Determine how payment is to be made, 3) Determine nonperformance remedy, 4) Prepare contract provisions, 5) Review contract provisions with legal counsel.			

Table 13. Template of "Establish the Supplier Agreement"

Inputs	*Subpractices*	*Tools and techniques*	*Example work products*
• Responsibilities of the acquirer and the supplier. • Terms and conditions of the acquirer. • Terms of quality assurance. • Payment terms.	• Not found.	• Not found.	• Clear correspondence between the clauses of the agreements. • FAQs (for users, and other providers). • Supplier agreements (including terms and conditions). • Acceptance criteria. • Supplier performance criteria. • Established contract.
Informative Components: Not found			

6 Conclusions

The goal of this study was to establish a multi-model workflow for the Solicitation and Supplier Agreements Development (SSAD) process area of CMMI-ACQ model, in order to allow the acquirer to identify the activities that must be considered before establishing or formalizing an acquisition or outsourcing contract. In addition, the required and support activities to help implementing the main activities sequences in the workflow are established. Also, the multi-model workflow would help organizations to choose the best practices compliant with the available models and standards in the market.

Finally, the multi-model workflow validation has been carried out through meetings with professional expertise in outsourcing or acquisition projects through everis consultants, through which the activities sequence into the multi-model workflow has been detailed and refined, as well as the required and support activities.

In addition, according to the considerations of everis consultants, it is concluded that the obtained multi-model workflow shows a roadmap to implement the activities before establishing an acquisition or outsourcing contract, which is largely compatible within the business practice. This traceability is an evidence of the adequacy of the multi-model workflow.

Acknowledgements. This work is sponsored by everis Foundation and Universidad Politécnica de Madrid through the Software Process Improvement Research Group for Spain and Latin American Region.

References

1. Huai, J.: An Incentive Model of IS Outsourcing Contract. In: International Conference on Wireless Communications, Networking and Mobile Computing, WiCom (2007)
2. Gallagher, B., Phillips, M., Richter, K., Shrum, S.: CMMI for Acquisition, Version 1.2 (2007)
3. Gallagher, B., Phillips, M., Richter, K., Shrum, S.: CMMI for Acquisition, Version 1.3 (2011)
4. Huai, J.: An Incentive Model of IS Outsourcing Contract. In: International Conference on Wireless Communications, Networking and Mobile Computing, WiCom (2007)
5. Byrne, J., Byrne, P.J., Heavey, C., Liston, P.: Contract costing in outsourcing enterprises: Exploring the benefits of discrete-event simulation. International Journal of Production Economics (2007)
6. Gellings, C.: Outsourcing Relationships: The Contract as IT Governance Tool. In: 40th Annual Hawaii International Conference on System Sciences (2007)
7. Chun-ping, H., Pei, Z., Zhen-xiang, Z.: Study on Critical Success Factors for IT Outsourcing Lifecycle. In: International Conference on Wireless Communications, Networking and Mobile Computing (2007)
8. Dibbern, J., Goles, T., Hirschheim, R., Jayatilaka, B.: Information systems outsourcing: a survey and analysis of the literature. SIGMIS Database 35(4), 6–102 (2004)
9. Bartels, A.: US and Global IT Market Outlook: Q1 2010, Forrester Research (2010)
10. Babar, M.A., Verner, J.M., Nguyen, P.T.: Establishing and maintaining trust in software outsourcing relationships: An empirical investigation. Journal of Systems and Software (2007)
11. Jeanne, R., Beath, C.: Success and Failure in Outsourcing. CIO Newsletter (2006)
12. TPI, R. Restructuring outsourcing agreements: an indication of failure, or a tool to increase value (2007)
13. Consulting, J.R. The Outsourcing Contracts Dilemma: Best Practices for Contract Formation within the Outsourcing Lifecycle (2009)
14. Dodson, K.M., Hofmann, H.F., Ramani, G.S., Yedlin, D.K.: Adapting CMMI for Acquisition Organizations: A Preliminary Report. Software Engineering Institute (2006)
15. Wolfram Mathematica. Documentation Center (2008)
16. Bang-Jensen, J., Gutin, G.: Digraphs: theory, algorithms and applications. Springer, Heidelberg (2002)
17. Calvo Manzano Jose A., Gonzalo, C., Mirna, M., Tomás, S.F.: Process Similarity Study: Case Study on Project Planning Practices Based on CMMI-DEV v1.2. In: 15th EuroSPI (2008)

18. Calvo-Manzano Jose, A., Gonzalo, C., Gerzón, G., Jezreel, M., Mirna, M., Tomás, S.F., Ángel, S.: Processes Implementation Sequences to Achieve the Maturity Level 2 in CMMI-ACQ. In: 16th EuroSPI (2009)
19. Standard Committee IEEE, IEEE Std 1062: 1998 Recommended Practice for Software Acquisition. Software Engineering Standards Committee of the IEEE Computer Society 22 (1998)
20. Society, I.C., IEEE Guide Adoption of PMI Standard a Guide to the Project Management Body of Knowledge. IEEE Std 1490-2003 (2004)
21. Standard Committee IEEE, Systems and Software Engineering - Software Life Cycle Processes. IEEE STD 12207-2008 (2008)
22. ITSqc, The eSCM-CL v1.1: Model Overview (2006)

ISO/IEC 15504-5 Best Practices for IT Service Management

Antoni Lluís Mesquida and Antonia Mas

University of the Balearic Islands
Ctra. de Valldemossa, Km. 7.5. 07122 Palma de Mallorca, Spain
{antoni.mesquida,antonia.mas}@uib.es

Abstract. Software companies which have been involved in a process improvement programme according to ISO/IEC 15504 are showing an increasing interest in the ISO/IEC 20000 standard for Information Technology Service Management. With the intention of supporting these companies in the implementation of the ISO/IEC 20000 standard and in order to facilitate the simultaneous application of both standards, the existing relations between ISO/IEC 15504-5 base practices and ISO/IEC 20000-4 process outcomes have been analysed and the results are presented in this paper.

Keywords: ISO/IEC 15504 (SPICE), ISO/IEC 20000, IT Service Management (ITSM), Software Process Improvement (SPI).

1 Introduction

Information Technology Service Management (ITSM) is a process-oriented discipline which combines process management and industry best practices into a standard approach for optimizing IT services. ITSM provides specific processes, frameworks, methodologies and guidance to enable organizations to deliver quality IT services to meet business needs and adhere to service level agreements. Some of the ITSM reference frameworks and internationally accepted standards are ITIL (Information Technology Infrastructure Library), ISO/IEC 20000 and CMMI (Capability Maturity Model Integration) for Services (CMMI-SVC).

MiProSoft, our research group, has led several editions of the QuaSAR project, a SPI programme according to the ISO/IEC 15504 standard in software companies within our environment [1][2][3]. Many of these companies have recently shown a special interest in ITSM standards and, more concretely, they demand the implementation of ISO/IEC 20000 as ITSM framework. Heads of quality departments in these organizations have observed that some of the actions performed to implement ISO/IEC 15504 base practices could be very useful to implement the requirements of ISO/IEC 20000.

During the last years, some initiatives relating ITSM and SPI according to ISO/IEC 15504 have emerged. In that sense, several authors have considered the possibility of developing an ITSM extension for ISO/IEC 15504 [4][5]. Public Research Centre Henri Tudor has developed TIPA, a framework for the assessment of IT Service

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 14–24, 2011.

Management processes with the ISO/IEC 15504 standard assessment approach [6]. Moreover, ISO is currently working on the new standard ISO/IEC NP TR 15504-8 *Part 8: An exemplar process assessment model for IT service management*. This part will describe a Process Assessment Model for ITSM processes and will be closely related to the ITSM Process Reference Model described in ISO/IEC 20000-4.

In this paper we focus on the relationship between the processes of ISO/IEC 20000 and the base practices of ISO/IEC 15504 with a double objective:

- To facilitate the implementation of ISO/IEC 20000 in organizations which are already involved in an ISO/IEC 15504 SPI programme.
- To define a method to maximize the efficiency of a simultaneous application of ISO/IEC 15504 and ISO/IEC 20000 in software companies which have not implemented yet any of these standards.

This paper is structured as follows: section 2 presents a summary of the standards used in this work. Section 3 defines the process followed to perform the mapping between the standards. Section 4 analyses the detected relations. Section 5 details the results and opens discussions regarding them. Finally, in section 6 the conclusions are presented.

2 Background

This section offers a brief description of the standards that this research deals with.

2.1 ISO/IEC 20000 Information Technology - Service Management

ISO/IEC 20000 *Information technology - Service management* is an ITSM quality standard that promotes the adoption of an integrated process approach to effectively deliver managed services to meet the business and customer requirements. This standard consists of five parts. In this research, only the two following parts have been considered:

- ISO/IEC 20000-1:2005 *Part 1: Specification* [7] defines the requirements for a service provider to plan, establish, implement, operate, monitor, review, maintain and improve a Service Management System (SMS). The requirements include the design, transition, delivery and improvement of services to fulfil agreed service requirements.
- ISO/IEC TR 20000-4:2010 *Part 4: Process reference model* [8] offers a logical representation of the elements of the processes within service management, including the general SMS processes. The processes described in ISO/IEC 20000-4 are considered to be the minimum necessary to meet ISO/IEC 20000-1 requirements. This process reference model comprises a set of 26 processes which are structured in 6 categories. Each process is described in terms of a purpose and outcomes, which are observable results of the successful achievement of the process purpose. Table 1 shows the ISO/IEC 20000-4 six process categories and the number of processes and outcomes per category.

Table 1. Summary of ISO/IEC 20000-4 process categories

Process categories	Processes	Outcomes
SMS general processes	9	52
Design and transition of new or changed services processes	4	28
Service delivery processes	6	39
Control processes	3	19
Resolution processes	2	11
Relationship processes	2	15
Total:	26	164

2.2 ISO/IEC 15504-5 Part 5: An Exemplar Process Assessment Model

ISO/IEC 15504-5 [9] describes an exemplar Process Assessment Model for the particular case of the software lifecycle processes defined in ISO/IEC 12207/Amd 1&2 *Information technology - Software life cycle processes* [10]. The standard defines process performance indicators, also known as Base Practices (BP), for each one of the 48 software lifecycle processes which are structured in 9 process groups. Table 2 shows these nine process groups and the number of processes and BP per group.

Table 2. Summary of ISO/IEC 15504-5 process groups

Process groups	Processes	BP
Acquisition (ACQ)	5	23
Supply (SPL)	3	25
Engineering (ENG)	12	66
Operation (OPE)	2	11
Management (MAN)	6	52
Process Improvement (PIM)	3	23
Resource & Infrastructure (RIN)	4	29
Reuse (REU)	3	26
Support (SUP)	10	73
Total:	48	328

3 Research Method

The study of the relations between the two standards was done by following an iterative and evolving strategy in which the ISO/IEC 20000-4 process outcomes and the ISO/IEC 15504-5 base practices have been compared. This version of the mapping is the result of a successive refinement process performed in three stages as shown in Figure 1.

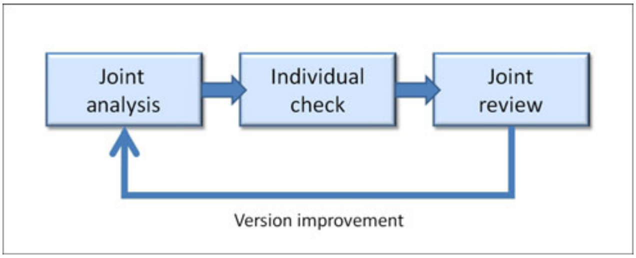


Fig. 1. The mapping process flow

With the objective of sharing the knowledge and the different points of view among the authors, during the joint analysis stage both standards were analysed in group. Since it was not possible to perform a complete mapping in only one session, different meetings were necessary in order to obtain a first preliminary version of the mapping. During each meeting five or six ISO/IEC 20000-4 processes were analysed.

More specifically, for each process, all its outcomes were examined in depth. It should be noted that the authors' knowledge of the ISO/IEC 15504 standard facilitated the initial selection of the set of ISO/IEC 15504-5 processes related to the ISO/IEC 20000-4 process outcome under consideration. After a detailed analysis of the base practices of the ISO/IEC 15504-5 selected processes, it was possible to determine the existence or not of a connection between a base practice and the ISO/IEC 20000-4 process outcome. Three types of correspondence were detected:

- Correspondence between an ISO/IEC 20000-4 process outcome and one or more base practices of an ISO/IEC 15504-5 process.
- Correspondence between an ISO/IEC 20000-4 process outcome and one or more base practices of different ISO/IEC 15504-5 processes.
- Nonexistence of a correspondence between an ISO/IEC 20000-4 process outcome and an ISO/IEC 15504-5 process.

With the intention of consolidating the results obtained after the meetings, these results were individually re-examined by each author to confirm the decisions reached or, on the contrary, to make some modifications to the initial version of the mapping.

Finally, during the joint review stage the individual proposals of each author were carefully discussed in order to reach a general consensus to accept or reject each proposal.

4 Analysis of the Relations

From the analysis of the ISO/IEC 20000-4 process outcomes and the ISO/IEC 15504-5 base practices, four different types of relations have been established:

1. **Full relation.** In this case, all the outcomes of an ISO/IEC 20000-4 process are covered by ISO/IEC 15504-5 base practices. Table 3 shows the twelve ISO/IEC 20000-4 processes with this type of relation. All these processes, except one, are covered by base practices of only one ISO/IEC 15504-5 process. The *Human resource management process* is covered by base practices of two different ISO/IEC 15504-5 processes.

Table 3. ISO/IEC 20000-4 processes fully covered by ISO/IEC 15504-5 base practices

ISO/IEC 20000-4 processes	ISO/IEC 15504-5 processes
Audit	SUP.5 Audit
Change management	SUP.10 Change request management
Configuration management	SUP.8 Configuration management
Human resource management	RIN.1 Human resource management RIN.2 Training
Improvement	PIM.3 Process improvement
Incident management and request fulfilment	SUP.9 Problem resolution management
Information item management	SUP.7 Documentation
Management review	SUP.4 Joint review
Measurement	MAN.6 Measurement process
Risk management	MAN.5 Risk management
Service planning and monitoring	MAN.3 Project management
Service reporting	SUP.7 Documentation

2. **Large relation.** In this case, not all but the majority of the outcomes of an ISO/IEC 20000-4 process are covered by ISO/IEC 15504-5 base practices. Table 4 shows the seven processes with this type of relation.

Table 4. ISO/IEC 20000-4 processes largely covered by ISO/IEC 15504-5 base practices

ISO/IEC 20000-4 processes	ISO/IEC 15504-5 processes
Budgeting and accounting for IT services	MAN.3 Project management
Information security management	RIN.4 Infrastructure MAN.5 Risk management
Organizational management	SPL.1 Supplier tendering SPL.2 Product release MAN.1 Organizational alignment MAN.2 Organizational management MAN.3 Project management MAN.5 Risk management ENG.1 Requirements elicitation ACQ.5 Customer acceptance SUP.4 Joint review
Problem management	SUP.9 Problem resolution management MAN.5 Risk management
Service requirements	ENG.1 Requirements elicitation SUP.3 Validation
SMS establishment and maintenance	PIM.1 Process establishment PIM.3 Process improvement
Supplier management	ACQ.2 Supplier selection ACQ.4 Supplier monitoring

3. **Partial relation.** In this type of relation only some outcomes of an ISO/IEC 20000-4 process are partially addressed by ISO/IEC 15504-5 base practices. This is the case of the processes *Business relationship management*, *Release and deployment management*, *Service level management* and *Service*

transition. Some of the outcomes of these processes are related to base practices of OPE.1 Operational use, OPE.2 Customer support and SPL.2 Product release processes.

4. **Nonexistence of relation.** In the last case, the ISO/IEC 20000-4 process is not related to any ISO/IEC 15504-5 base practice. This is the case of the processes *Capacity management*, *Service continuity and availability management* and *Service design*. Because of its particular nature, these processes are related to service provision activities which are not covered by ISO/IEC 15504-5.

4 Results and Discussion

In this section, an analysis of the results obtained from the study of the relations between both standards is performed. These results are analysed from different perspectives.

4.1 Summary of the Relations

Firstly, a high level view of the relations between the ISO/IEC 20000-4 process categories and the ISO/IEC 15504-5 process groups is shown in Table 5. Observing this table only an overall analysis can be performed. On the one hand, it can be seen that all the ISO/IEC 20000-4 process categories have been related, at least, to two ISO/IEC 15504-5 processes. On the other hand, it can be concluded that the processes of the Reuse process group (REU) cannot be used to facilitate ISO/IEC 20000 implementation.

Table 5. Relations between the ISO/IEC 20000-4 process categories and the ISO/IEC 15504-5 process groups

ISO/IEC 20000-4	ISO/IEC 15504-5 process groups								
process categories	ACQ	SPL	ENG	OPE	MAN	PIM	RIN	REU	SUP
SMS general processes	ACQ.5	SPL.1 SPL.2	ENG.1		MAN.1 MAN.2 MAN.3 MAN.5 MAN.6	PIM.1 PIM.3	RIN.1 RIN.2		SUP.4 SUP.5 SUP.7
Design and transition of new or changed services processes		SPL.2	ENG.1	OPE.1	MAN.3				SUP.3
Service delivery processes				OPE.2	MAN.3 MAN.5		RIN.4		SUP.7
Control processes		SPL.2							SUP.8 SUP.10
Resolution processes					MAN.5				SUP.9
Relationship processes	ACQ.2 ACQ.4			OPE.2					

Table 5 can be analysed from two different points of view. An analysis by rows gives information about the relations from the perspective of ISO/IEC 20000-4 process categories. An analysis by columns determines the relations from the perspective of ISO/IEC 15504-5 process groups. Both points of view are analysed in next sections.

4.2 Analysis from the ISO/IEC 20000-4 Perspective

In order to perform the analysis from this perspective we will focus on the first row of Table 5, which shows the relations of the SMS general processes. Looking at the first column, the Acquisition process group (ACQ), it could seem that the ACQ.5 Customer acceptance process is related to all the processes of the SMS general processes category. However, observing Table 6, which shows the relations of all the processes of the SMS general processes category, it can be seen that ACQ.5 is only related to the Organizational management process.

In Table 6, a strong correspondence between the SMS general processes and some ISO/IEC 15504-5 processes can be also observed. The majority of these processes, with the exception of the *Organizational management process*, are fully covered by only one (two in some cases) ISO/IEC 15504-5 process. The outcomes of the *Organizational management process* are related to base practices of nine different ISO/IEC 15504-5 processes.

Table 6. Relations between the ISO/IEC 20000-4 SMS general processes and the ISO/IEC 15504-5 processes

ISO/IEC 20000-4	ISO/IEC 15504-5 process groups								
SMS general processes	ACQ	SPL	ENG	OPE	MAN	PIM	RIN	REU	SUP
Audit									SUP.5
Human resource management							RIN.1 RIN.2		
Improvement						PIM.3			
Information item management									SUP.7
Management review									SUP.4
Measurement					MAN.6				
Organizational management	ACQ.5	SPL.1 SPL.2	ENG.1		MAN.1 MAN.2 MAN.3 MAN.5				SUP.4
Risk management					MAN.5				
SMS establishment and maintenance						PIM.1 PIM.3			

In order to observe an example of the maximum level of detail, Table 7 shows the mapping between the ISO/IEC 20000-4 Audit process outcomes and the ISO/IEC 15504-5 SUP.5 Audit process base practices. The *Audit process* outcomes are completely covered by the SUP.5 Audit process.

Table 7. Mapping of the ISO/IEC 20000-4 Audit process

ISO/IEC 20000-4 Audit process outcomes	ISO/IEC 15504-5 SUP.5 Audit process base practices
1. The scope and purpose of each audit is defined [and agreed]	SUP.5.BP1: Develop and implement an audit strategy.
2. The objectivity and impartiality of the conduct of audits and selection of auditors are assured	SUP.5.BP2: Select auditors.
3. Conformity of selected services, products and processes with requirements, plans and agreements is determined	SUP.5.BP3: Audit for conformance against the requirements.
4. Nonconformities are recorded	SUP.5.BP3: Audit for conformance against the requirements.
5. Nonconformities are communicated to those responsible for corrective action and resolution	SUP.5.BP4: Prepare and distribute an audit report. SUP.5.BP5: Take corrective action.
6. Corrective actions for nonconformities are verified	SUP.5.BP6: Track resolution.

4.3 Analysis from the ISO/IEC 15504-5 Perspective

In order to perform the analysis from this perspective we will focus on the columns of Table 5. As it has already been said in Section 4.1, the only process group without relation is the Reuse process group (REU). The purpose of the REU processes is to manage the life of reusable assets and to plan, establish, manage, control, and monitor an organization's reuse program to systematically exploit reuse opportunities. ISO/IEC 15504-5 considers as reusable assets the requirements, designs, code, test cases and libraries, that is, software and hardware components. Since services are not considered to be reusable assets, no evidences of relation with REU base practices have been identified.

Focusing on the Support process group (SUP) column, it can be observed that it is related to almost all the ISO/IEC 20000-4 process categories. The SUP process group consists of processes that support another process as an integral part with a distinct purpose and contributes to the success and quality of the software project. These processes are also employed and executed by several ITSM processes of ISO/IEC 20000-4.

Table 8 shows all the relations between the SUP processes and the ISO/IEC 20000-4 process categories. Seven of the ten processes of this group have been related to some process category. Only SUP.1, SUP.2 and SUP.6 have not been related to any process category. On the other hand, the *Relationship processes* category is the only category that has not been related to any SUP process.

Table 8. Relations between ISO/IEC 15504-5 Support processes and the ISO/IEC 20000-4 process categories

ISO/IEC 20000-4 process categories	ISO/IEC 15504-5 Support processes									
	SUP. 1	SUP. 2	SUP. 3	SUP. 4	SUP. 5	SUP. 6	SUP. 7	SUP. 8	SUP. 9	SUP. 10
SMS general processes				(2)	(3)		(4)			
Design & transition of new or changed services processes			(1)							
Service delivery processes							(5)			
Control processes								(6)		(8)
Resolution processes									(7)	
Relationship processes										

(1) *Service requirements process*
(2) *Management review process* and *Organizational management process*
(3) *Audit process*
(4) *Information item management process*
(5) *Service reporting process*
(6) *Configuration management process*
(7) *Incident management and request fulfilment process* and *Problem management process*
(8) *Change management process*

As before, in order to observe an example of the maximum level of detail from this perspective, the SUP.9 Problem resolution management process is deeply analysed. Table 9 details all the relations between SUP.9 base practices and the outcomes of *Incident management and request fulfilment* and *Problem management* processes.

Table 9. Mapping of the ISO/IEC 15504-5 SUP.9 Problem resolution management process

ISO/IEC 15504-5 SUP.9 Problem resolution management process	ISO/IEC 20000-4 process	
	Incident Management and request fulfilment	Problem Management
SUP.9.BP1: Develop problem resolution strategy. Determine the problem resolution strategy for ensuring that problems are described, recorded, analyzed, and corrected.	Outcome 1: incidents and service requests are recorded and classified.	
SUP.9.BP2: Identify and record the problem. Each problem is uniquely identified, and recorded.	Outcome 1: incidents and service requests are recorded and classified.	
SUP.9.BP3: Provide initial support and classification. Provide initial support and feedback on reported problems and classify problems according to the severity.	Outcome 2: incidents and service requests are prioritised and analysed.	
SUP.9.BP4: Investigate and diagnose the cause of the problem. Analyze problems in order to identify the cause of the problem.		Outcome 2: problems are prioritised and analysed.

Table 9. (*Continued*)

ISO/IEC 15504-5 SUP.9 Problem resolution management process	ISO/IEC 20000-4 process Incident Management and request fulfilment	Problem Management
SUP.9.BP5: Assess the impact of the problem to determine solution. Assess the impact of the problem to determine appropriate actions, and to determine and agree on a solution.	Outcome 2: incidents and service requests are prioritised and analysed.	Outcome 2: problems are prioritised and analysed.
SUP.9.BP6: Execute urgent resolution action, where necessary. If the problem warrants immediate resolution pending an actual change, it obtains authorization for immediate fix.	Outcome 3: incidents and service requests are resolved and closed.	
SUP.9.BP7: Raise alert notifications, where necessary. If the problem is of high severity and impacts other systems or users, an alert notification may need to be raised, pending a fix or change.	Outcome 5: information regarding the status and progress of reported incidents or service requests is communicated to interested parties.	
SUP.9.BP8: Implement problem resolution. Implement problem resolution actions to resolve the problem and review the implementation.	Outcome 3: incidents and service requests are resolved and closed.	Outcome 3: problems are resolved and closed.
SUP.9.BP9: Initiate change request. Initiate change request for diagnosed errors.		
SUP.9.BP10: Track problem status. Track to closure the status of identified problems.	Outcome 5: information regarding the status and progress of reported incidents or service requests is communicated to interested parties.	

5 Conclusion and Lessons Learned

In this paper, an analysis of the existing relations between the ISO/IEC 20000-4 outcomes and the ISO/IEC 15504-5 base practices has been presented. As it has been showed, ISO/IEC 15504-5 considers an important number of the ISO/IEC 20000-4 process outcomes. Consequently, software companies involved in a process improvement programme according to ISO/IEC 15504 have already performed some steps in order to implement the Service Management System of the ISO/IEC 20000 standard.

It has to be noted that, although some ISO/IEC 15504-5 processes can be easily adapted to cover some requirements of the ISO/IEC 20000, there is still a significant number of process outcomes that do not have any relation to ISO/IEC 15504-5 processes, and therefore, they must be implemented as indicated in the ISO/IEC 20000-4 and ISO/IEC 20000-1 standards.

The results presented in this paper represent one more step on the road to developing a method with the necessary guidelines for the implementation of both standards reducing the amount of effort. This method could be used to facilitate the implementation of ISO/IEC 20000 in software companies which are currently, or will be in the near future, involved in SPI programmes according to ISO/IEC 15504.

There is no proof of software companies in our environment that have obtained the ISO/IEC 20000 certification after being previously involved in SPI programmes according to ISO/IEC 15504. However, several software companies involved in SPI programmes according to ISO/IEC 15504 which are interested in implementing ISO/IEC 20000 will participate in the next edition of the QuaSAR project that our research group is currently preparing. From the results of this project, we expect to improve our method by considering the lessons learned from its application.

Moreover, we are also considering the possibility of developing a software tool to support the implementation of both standards.

Acknowledgments. This research has been supported by CICYT-TIN2010-20057-C03-03 "Simulación aplicada a la gestión de equipos, procesos y servicios", Sim4Gest.

References

1. Amengual, E., Mas, A.: Software Process Improvement in Small Companies: An Experience. In: 14th European Software Process Improvement Conference, Germany, pp. 11.11–11.18 (2007)
2. Mas, A., Fluxà, B., Amengual, E.: Lessons learned from an ISO/IEC 15504 SPI Programme in a Company. In: 16th European Systems & Software process Improvement and Innovation Conference, Spain, pp. 4.13–4.18 (2009)
3. Mas, A., Amengual, E., Mesquida, A.L.: Application of ISO/IEC 15504 in Very Small Enterprises. In: Riel, A., O'Connor, R., Tichkiewitch, S., Messnarz, R. (eds.) EuroSPI 2010. CCIS, vol. 99, pp. 290–301. Springer, Heidelberg (2010)
4. Malzahn, D.: A service extension for SPICE? In: International SPICE Conference on Process Assessment and Improvement, South Korea (2007)
5. Nehfort, A.: SPICE Assessments for IT Service Management according to ISO/IEC 20000-1. In: International SPICE days 2007, Germany (2007)
6. Public Research Centre Henri Tudor: ITSM Process Assessement Supporting ITIL®. Using TIPA to Assess and Improve your Processes with ISO 15504 and Prepare for ISO 20000 Certification. Van Haren Publishing, Zaltbommel (2009)
7. ISO/IEC: ISO/IEC 20000-1:2005 Information Technology - Service Management - Part 1: Specification (2005)
8. ISO/IEC: ISO/IEC TR 20000-4:2010 Information Technology - Service Management - Part 4: Process reference model (2010)
9. ISO/IEC: ISO/IEC 15504-5:2006 Information technology - Process Assessment - Part 5: An exemplar Process Assessment Model (2006)
10. ISO/IEC: ISO/IEC 12207:1995/Amd 1:2002/Amd 2:2004 Information technology - Software life cycle processes (1995)

A Self-assessment Framework for Finding Improvement Objectives with ISO/IEC 29119 Test Standard

Jussi Kasurinen[1], Per Runeson[2], Leah Riungu[1], and Kari Smolander[1]

[1] Software Engineering Laboratory, Lappeenranta University of Technology, P.O. Box 20, FI-53851 Lappeenranta, Finland
{jussi.kasurinen,leah.riungu,kari.smolander}@lut.fi
[2] Software Engineering Research Group, Lund University, P.O. Box 118, SE-22100 Lund, Sweden
per.runeson@cs.lth.se

Abstract. One of the latest additions in defining the test process is the upcoming ISO/IEC 29119 standard, which aims to define a universally applicable generic test process model. However, currently the standard does not offer any support for the adoption process of the model. In this paper, we present our framework, which aims to combine a maturity level-based approach with the standard process. Our objective was to create an easy-to-use framework for organizations to assess how closely their existing test process follows the standard, and give feedback on improvement objectives. Our results indicate that combining maturity levels with the standard is a viable approach to assess the implementation of the standard in practice.

Keywords: ISO/IEC 29119, self-assessment framework, test process maturity.

1 Introduction

In an ideal world, every time a new software system is produced, the test phase verifies every possible use case and scenario that can be done with the new system. However, the reality is that in the most cases, this is simply not possible, as comprehensive testing for a complex programs would take too long or consume too many resources [1]. In fact, even with realistic test resources, the test process usually ends up being the most expensive item in the software development project [2], taking as much as half of the development budget. Testing also has a large role in determining whether the end-product is commercially successful or not [3].

Because the test process is important for successful software development, it also means that the test process should be appropriate for the software organization and follow good practices. To help organizations to achieve this objective, the International Organization for Standardization (ISO) is developing a new standard focusing solely on software test process, called ISO/IEC 29119 Software Testing [4].

In this study, the current draft of the standard is examined from the viewpoint of its applicability in organizations. Our objective is to extend the standard with maturity

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 25–36, 2011.

levels, and to define a practical framework for organizations to self-assess their existing test process against the standard. The framework also aims to create a simple process assessment tool, which produces practical objectives on improving the test process and adopting practices as defined in the ISO/IEC 29119 standard.

The method to develop such a framework was to combine the processes defined in the standard test process model with an existing testing-related maturity model. In this study, the levels from Test Improvement Model (TIM) [5] were applied. This maturity model was selected based on the conceptual similarities between the key areas in the maturity model and the test processes defined in the ISO/IEC 29119 standard. The resulting framework was also assessed for its feasibility with a pilot study, in which the framework was used to evaluate four real-life case organizations' existing test processes. In three of these organizations, the profile was also assessed for accuracy and usability by the organization itself to gain feedback and enhancement proposals for further development.

The rest of the paper is constructed as follows: In Section 2, the existing test process maturity models and other related research is presented. In Section 3, the developed self-assessment framework, combining ISO/IEC 29119 processes and TIM levels is introduced. In Section 4, the pilot study with the framework and evaluation of the produced development objectives is discussed. Section 5 discusses the framework and pilot study, while Section 6 brings the paper to a conclusion.

2 Related Research

The ISO/IEC 29119 test standard [4] is by no means the first model for defining test processes. The Test Maturity Model integrated (TMMi) [6] defines a maturity model for assessing test processes by applying principles of Capability maturity model integrated (CMMi) [7] framework on a software testing context. Unlike the ISO/IEC 29119 standard, TMMi is structured as an assessment model, where TMMi practices are adopted in several maturity phases. The TMMi-model focuses on iterative improvement processes, where the process maturity level increases as the process improves. However, the application of the TMMi model in real-life organizations suffers from some practical limitations [8,9]. For example, the concept of moving all processes to one level before advancing to next is unrealistic to implement [9], and the level requirements are criticized for being counter-intuitive [8].

Another model, which introduces maturity levels to test processes, is the Test Improvement Model (TIM) [5]. TIM is based on CMM, a predecessor of CMMi [7], and it focuses on developing the test process improvement objectives from the existing state of the test process in the assessed organization. TIM also addresses the limitation of requiring all of the processes to reach one level before moving to the next; the test process is divided to key areas, such as *planning and tracking* or *testware*, which are all assessed individually. The key areas are assessed based on the existing practices, in levels which are *baseline*, *cost-effectiveness*, *risk-lowering* and *optimizing*, in the order of maturity. The model promotes a concept of balanced development, but does not enforce a strict parallel improvement of the key areas.

There are also previous attempts to develop a light-weight assessment framework for software testing. One such example is the model, which is especially geared towards small- to medium-sized [10] businesses, the Minimal Test Practice Framework (MTPF), designed by Karlström et al. [11]. This model elaborates on the concepts presented in the TMMi and TIM, simplifying the assessment process and enabling the framework to be applied in the smaller organizations, which may not have a need for extensive, large models. The MTPF model simplifies the test process into five categories and defines a three-step process for each category to mature towards the better testing practices, especially to align with the growth of the company. The model aims to simplify the process improvement process, offering concrete improvement objectives for small test organizations.

3 Self-assessment Framework

The draft for ISO/IEC 29119 standard model [4] aims to specify a generic test process model, which promotes the best practices of the discipline. However, the standard presents a process model, giving no input on how to assess the goals of process improvement or how to achieve conformance with the standard in a real-life organization. The objective of this study is to present a concept-level framework, with the intended purpose of enabling the evaluation of the existing test process against the practices defined in the standard.

3.1 The ISO/IEC 29119 Test Process Model

The current draft of the ISO/IEC 29119 [4] standard model is structured as several layers which have different processes. The topmost layer is the *organization level*, in which the organization-wide test policies and test strategy are defined. Below the organizational level is the *project-level* management, which defines the activities and responsibilities of the management in the individual projects. At this level, the organization-defined documents are used in definition of a test plan, and as a basis for deciding the test completion criteria. The project-level management also oversees the lowest level of the model, the *test execution level*, and reports test completion reports and feedback to the organizational management. At the test execution level the actual testing tasks are completed and reported to the management. The test execution level is further divided to two process types, static and dynamic. Static processes are the testing activities, which are carried out throughout the project, whereas dynamic processes are changing during the project advancement. In the model, these processes, which contain all of the activities, roles and responsibilities, are defined as follows:

- *Organizational test process* (OTP) develops and manages organizational test specifications, such as test policy and test strategy. It also is responsible for monitoring and controlling lower layers of the process.
- *Test management processes* (TMP) are the project-level management activities in the test process. TMP defines the test planning, test monitoring and control and test completion, and is also responsible for maintaining the test plans.
- *Test planning process* (TPP) is the process which is responsible for developing the test plan. Depending on the project phase, this may be project test plan, or test plan for a specific phase.

- *Test monitoring and control process* (TMCP) ensures that the testing is performed in line with the test plan and organizational test documents. It also is responsible for identifying updates necessary for the test plan.
- *Test completion process* (TCP) is the process which includes activities, which are done when testing is completed. It also ensures that the useful test assets are made available for later use.
- *Static test processes* (STP) describes how static testing activities, such as test preparation, test result review or test follow-up are done. These activities are the "general" activities, which are done to all test cases in all test phases of the project.
- *Dynamic test processes* (DTP) describe how dynamic test activities such as test implementation, test execution, test environment set-up and test incident reporting are done in the organization. These activities are the "practical" activities, which vary between different types of testing.

Some of the processes (like STP or TCP) also create output, which is used as an input in another process (like TMP or OTP). Some of the processes (for example TMP) are also the owners of the other processes, such as TPP or TCP. The relationships between different processes and model layers are illustrated in Figure 1.

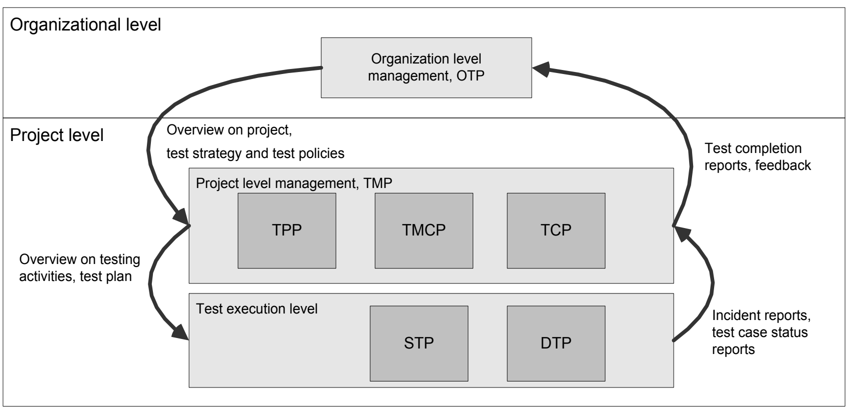


Fig. 1. ISO/IEC 29119 Test process model in a nutshell

3.2 Maturity Levels for Framework

The Test Improvement Model (TIM) [5] is a maturity model based on SEI's Capability Maturity Model (CMM) [see 7], focusing on test process improvement. The model applies key areas (KA), which are similar to CMM's key process areas (KPA), assessing important process factors in five levels of maturity. Unlike some improvement models such as TMMi [6], TIM encourages process improvements even in a single key area, as they are assessed separately and are rather independent. Even though it is possible to focus just on one or two process areas, TIM encourages balanced improvement. In a TIM model, level requirements are detailed for all key areas, which are *organization, planning and tracking, test cases, testware* and *reviews*. Even though the actual definitions for each level in different key areas vary, they can be generally defined as follows:

- *Level 0, Initial*: This represents the level of activity where the baseline is not measured or is non-measurable, meaning that the assessed activity does not exist in the organization, is not measurable, or it has not been measured.
- *Level 1, Baseline*: The organization has documented, generally agreed methods of doing testing, and it does basic testing functions, having dedicated people and resources for doing the testing tasks.
- *Level 2, Cost-effectiveness*: The organization has systematic efforts to become cost-effective and increase efficiency of product problem detection.
- *Level 3, Risk-lowering*: The organization is prepared to act on undesired effects. The organization applies measurements and is capable of early involvement and preventative actions to lower the risks of the project.
- *Level 4, Optimizing*: The organization is able to do Quality Assurance and the testing is fully integrated to the development project. Testing activities are continuously maintained and improved based on quality policies, needs and metrics.

In the TIM model, the key areas are assessed separately from each other, so that the organization has a better understanding on what test process areas mostly need improvements. Furthermore, the key areas of TIM maturity model are similar to ISO/IEC 29119 processes; the organization is conceptually close to organizational management process (OTP), planning and tracking to test management process (TMP) and TMCP, test cases to test plan process (TPP), testware to STP and DTP, and reviews to TCP. In this sense, TIM-based maturity levels seem feasible for the purposes of the framework.

4 Pilot Study

In this section we introduce the pilot study, which was used to assess the feasibility of the conceptual framework. This section is divided into three parts; in first part, the data collection method and the case organizations are introduced, and the framework is used to develop profiles for each of the organizations. In part two, the case profiles are examined to derive process improvement objectives, and in part three, these objectives are discussed and assessed based on their feasibility and feedback provided by some of the profiled organizations.

4.1 Data Collection and Analysis

The framework was tested with four different case organizations, on which sufficient data existed to enable the analysis of test process activities. These case organizations were selected from our population of organization units [12], on which empirical data was collected for our previous studies [for example 13, 14] in test process improvement. These organization units represent different sizes [10] and types of professional software development:

- MobileSoft is a large, internationally operating software company, producing embedded software for mobile platforms.
- SoftPortals is a small-sized nationally operating company, producing service solutions for customer organizations.
- DesignSoft is a large, internationally operating software company, producing a software product used for computer-assisted design.
- GridSystems is a medium-sized internationally operating company, producing embedded software for electrical networks and heavy industry.

The interview data was collected in four interview rounds between fall 2008 and spring 2010. In four interviews, the representatives from each case organization were interviewed for several testing-related themes. In all organizations, the interviewees were selected from the same organization unit [12], meaning that the interviewees were working on the same development projects. A summary of these interview rounds is listed in Table 1, and the original questionnaires, with the info package for the interviewees, can be accessed at *http://www2.it.lut.fi/ project/MASTO/.*

Table 1. Data collection rounds

Round: Type of interview	Interviewee role in the organization	Interview themes
1st round: Semi-structured interview	Software designer or people responsible for software design and architecture.	Design and development methods, Testing strategy and -methods, Agile methods, Standards, Outsourcing, Perceived quality
2nd round: Structured survey, additional semi-structured questions	Manager, test manager or project leader responsible for development project or test process of a product.	Test processes and tools, Customer participation, Quality and Customer, Software Quality, Testing methods and -resources
3rd round: Semi-structured interview	Tester or people responsible for doing testing in the development project.	Testing methods, Testing strategy and –resources, Agile methods, Standards, Outsourcing, Test automation and –services, Test tools, Perceived quality Customer in testing
4th round: Semi-structured interview	Manager, test manager or project leader responsible for development project or test process of a product.	Test policies, Test strategies, Test plans, Testing work, Software architecture, Delivery models, New software development concepts

The amount of data collected from the case organizations allowed the research to focus on observing how the different standard processes are implemented in these organizations. Based on the empirical data it appeared that all the standard-defined test processes seemed to at least exist, so technically the assessment was feasible. The observations, made on each standard-defined process, are summarized in Table 2.

For the pilot study with the framework, the ISO/IEC 29119 processes were combined with the TIM-maturity scale. The used scale was based on the general definitions of TIM levels, and only few changes were done to the general definitions. For example, the level 1 requirement for the documented test process was eased to accept also verbally agreed testing practices if they were done in a systematic manner. In addition, as the standard steers the test process activities towards continuous development and measurement, the level 4 was used to denote a self-optimizing process, which is in conformance with the standard. Overall, the levels for assessing the organizations were applied with the following rules:

- *Level 0, Initial*: The organization does not have defined methods for this activity.
- *Level 1, Baseline*: The organization does have documented or at least generally agreed guidelines for these process activities, the process is systematically done.
- *Level 2, Cost-effectiveness*: The organization tries to systematically promote cost-effectiveness or increase the efficiency of the process activities.
- *Level 3, Risk-lowering*: The organization has metrics or other methods to enable organization to do risk-lowering and preventative actions in process activities.
- *Level 4, Optimizing*: The organization has activities that aim to optimize the process; activities are done in a manner that is conceptually similar to the standard.

The resulting profiles of case organizations are presented in Figure 2. Based on the observations made in organizations, a draft for practical descriptions indicating the maturity levels of different test processes was also compiled. This draft and an example of full case profile can be accessed at *http://www2.it.lut.fi/project/MASTO/*.

Table 2. Process activity observations from case organizations

Process	MobileSoft	GridSystems	SoftPortals	DesignSoft
OTP	-Applies quality model, test strategy and policy exist. -Organizational management tends to underestimate testing.	-Test process defined as a "guideline", vague documentation on the topic.	-Test process defined as a "guideline", part of quality system.	-Policy and Strategy exist, high abstract level. -Organization applies quality model.
TMP	-Management decides what test activities are done. -Management defined, but passive.	-Management sets focus of test cases. -Test management can influence on release schedules.	-Management defined, but passive. -Roles and responsibilities in project level clear.	-Management lays test plan, weekly meetings. -Test management can influence on release schedules.
TPP	-Test plans are tailored to projects; checklists for required tests. -Plan is kept updated, new cases added based on found issues.	-Test plan based on found issues in previous projects.	-Plan is used as an overview for test objectives, little content updates during the project.	-Test plan follows abstract design, design sometimes updated.
TMCP	-Checklists to ensure test coverage -Error database to overview case completion. -No separate metrics.	-Daily SCRUM meetings. -Case status evaluation at the organizational level. -Test focus slipping.	-Customer-required functionalities are inspected closely. -Use case status, unit test coverage used as metrics.	-Problems with follow-up on found issues. -Weekly meetings and code reviews. -Use case status and used hours as metrics.
TCP	-Test process is evaluated after every project. -Continuous development.	-Test completion reports used in new plans. -Effort to increase usage or error reports.	-Test completion reports are done, but little effect to testing practices.	-Test process is evaluated after projects, some effect on future testing.
STP	-The amount of available test tools restricts testing. -Effort to increase amount of testers, tools.	-Amount of test resources sufficient for tasks. -New test cases created according to focus areas defined by management.	-The amount of available resources sometimes restricts testing work.	-Test resources, environments, are documented. -Amount of test resources sufficient for tasks.
DTP	-ISEB-certified testers, tools selected based on recommendations. -No technical limitations caused by testing tools. -Tests follow test plan closely.	-Tests follow test plan closely. -Large amounts of automation -Effort to increase amount of test resources like personnel.	-Some test cases are designed but not implemented. -Test cases are strictly followed and reported forward.	-Test plan and exploratory testing, results reported. -Sufficient tools, effort to automate. -Effort to introduce ISEB-certification.

4.2 Developed Organizational Profiles

The observations made from organizations made it possible to assess the maturity levels of each standard-defined process separately. This resulted in four profiles, each defining one case (illustrated in Figure 2). Based on these profiles, it was possible to generate two results, the general maturity level of the test process in case organization and the performance level of the processes in the organization. The general maturity level would indicate the average test process maturity and how closely the organization resembled the ISO/IEC 29119 test process model. The performance levels for different processes could be used to prioritize process development to focus, and correct, problems in the individual test activities.

In MobileSoft, the general results indicated that the test processes were generally in a good order. The most obvious difficulties in the organization were the organizational management, which commonly underestimated the test process resource needs. The test completion reports were sometimes not used and some testers complained that there were periodical resource shortages on testing tools like test platforms.

In DesignSoft, the profile also indicated rather mature test process, having only a few issues that should be attended. Based on the observations, in DesignSoft the biggest issue was in follow-up actions if problems were observed during testing. The problem was that discovered issues did not seem to affect the test plan, and follow-up on resolving newfound issues was left to the tester who originally found the problem.

In SoftPortals the test process is less mature than in the other case organizations. The profile suggests that the test management in projects should be improved to enhance the overall testing. In this organization, the management takes a rather passive role on the test process, leaving much of the test planning and control to the developers and testers. In addition, some interviewees complained that the projects usually did not have enough test resources, like dedicated testers or time.

In GridSystems, the test process was divided between extremely well-organized activities in the project level, and the organizational activities which needed improvement. The testing infrastructure employed heavily into test automation, having

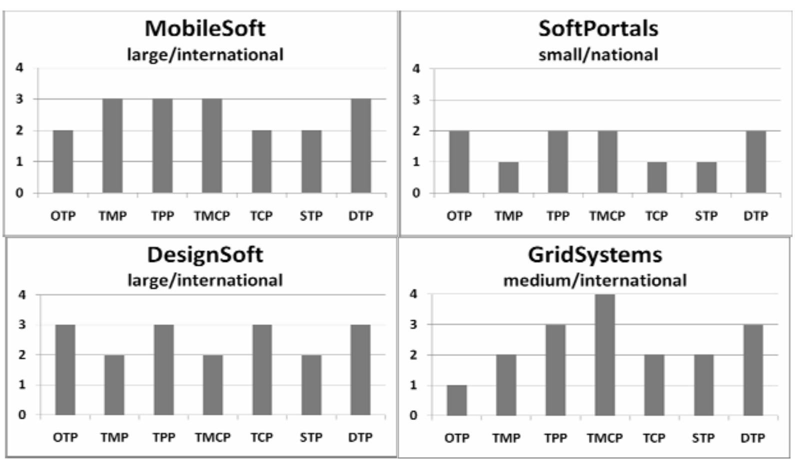


Fig. 2. Assessment results for case organizations using experimental levels

daily builds and Scrum-meetings. This enabled the organization to address test process issues within the same day the problems were observed. For these activities, the organization was considered to fully implement the concepts of the testing standard. However, a large issue was the organizational level management, as the OTP was only a distant decision maker, which provided only some vague guidelines.

4.3 Assessing Framework Results

The application of TIM maturity levels to the ISO/IEC 29119 standard was successful in a sense that the models are similar; the key factors from TIM and the test processes from ISO/IEC 29119 seem to map consistently to each other. We observed the case organizations using this framework and created an organizational profile based on the observed activities.

In the cases GridSystems and DesignSoft, the framework indicated clear process improvement objectives. In GridSystems, it seemed rather clear that the organization-level management needed to start paying attention to the test process activities. In DesignSoft, the framework indicated the need for developing test completion reports and follow-up activities on found issues during testing. In MobileSoft and SoftPortals the results based on the framework were more open to interpretation, but they did indicate some considerations for development objectives.

According to the profile, the case SoftPortals was the least mature organization in this study. However, based on the interviews, the organization itself does not consider their test process to be faulty or in need of major changes. Their test process seems to aim at cost-effectiveness and efficient application of the existing resources. In this sense, the framework indicates that the management should take a more active role in the process, and that test completion reports are not used efficiently. Should the organization start pursuing the risk-lowering strategy or optimizing test process, they are in need of a larger process improvement process.

In MobileSoft, the framework highlights some objectives for a better risk-lowering strategy. In this case organization, the test process in general is mature, and of the case organizations it is the best candidate to achieve the conformance with the ISO/IEC 29119 standard. The organization could capitalize the test completion reports and experience gained from completed projects more efficiently and the organizational management could be more active in the test process. The organization also follows an internal quality management system, which undergoes an assessment every few years. By making some improvements, it could be expected that they would reach the conformance with relatively minor changes to their test process.

In theory the framework seems plausible, but obviously the analysis based on pre-existing data offers only indications. To evaluate the feasibility of the framework further, three of the profiled case organizations were interviewed to assess the framework results. In the fourth case, GridSystems, the organization had changed so much that this further assessment was not possible. The overall attitude towards the framework and profiling results was positive, although further development needs were identified. Some of the feedback results are summarized in Table 4, where the sign "+ +" denotes that the organization was very positive towards the aspect of the framework, while "– –" denotes very negative feedback.

Table 4. Profiling feedback from the case organizations

	MobileSoft	DesignSoft	SoftPortals
Suitability of the framework	+; The applied approach is generally feasible.	++; Practical approach on quick and easy assessment of the level of different testing tasks.	+; Levels are too universal, but model itself seems to cover everything needed.
Suitability of the assessment levels	– –; In large organization, the levels overlap, unnecessary processes for some organizations.	+; Usable, although some processes do not need to be better than cost-effective.	– ; Levels in general are OK but the definitions should be less ambiguous.
Accuracy of the profile	– ; The profile should be more detailed.	+; The profile was accurate enough, although with some differences.	++; The profile represents the organization quite well.
Accuracy of the results	+; This type of feedback is always good for bringing out new ideas.	+; Results seemed usable.	++; Results same or similar to the internal discussions.
Framework development proposals	The assessment unit type and size should be clearly defined.	More definite descriptions for each framework level to reduce the overlap.	The assessment needs practical examples and more measurements.
Best profiler	An outsider from a third party, internal review is not accurate.	At least two manager-level employees; can be used internally.	A quality manager with a handpicked group of people, usable internally.

Based on the feedback, the framework is considered to be feasible with some criticism. In cases MobileSoft and SoftPortals the criticism focused on the ambiguity of the level definitions. The levels should have detailed metrics, or at least offer examples on what types of activities denote certain levels. MobileSoft also criticized the number of processes; in a software organization which is a part of a larger business unit, some of the activities were considered trivial. DesignSoft voiced a concern over the model levels; lowering risks may not always be a better objective than cost-effectiveness. As for the self-assessment, DesignSoft and SoftPortals considered the framework to be usable as a self-assessment tool, while MobileSoft voiced a concern over the accuracy of the self-assessment in general.

5 Discussion

Creation of a framework which combines maturity levels from one model and processes of an international standard is obviously open for criticism and discussion over its validity. It can be argued that the general requirements for any relevant construct should include at least that it is acceptable to the software development community and that it is based on agreed software engineering principles and practices [15]. The objective of the framework was to compare the existing test process against the ISO/IEC 29119 standard model. Based on the results of using the framework it was also possible to derive process improvement objectives, which would direct the organization towards practices defined in the standard. For this ability, an existing maturity model was fitted to the standard-defined processes. Both of the applied models, ISO/IEC 29119 and TIM, are well-known software engineering models, so theoretically the foundation for our framework should be sound.

The validity issues for developing frameworks have been addressed in several similar studies [8,11,15]. For example, Jung [8] developed a test process maturity model based on internal need, and validated the results via a case study and a survey. Similarly, with the MTPF-framework developed by Karlström et al. [11], the initial model was designed based on observations in real-life organizations, and further elaborated and validated with surveys and an empirical case study. If we compare these approaches to our framework, it is plausible to argue that the current results should be sufficient for a proof of concept: the results indicate that the framework could be developed into a usable tool, but obviously the framework needs further validation. The next obvious step is to address the observed difficulties and enhancement proposals. In general, more detailed qualitative studies and additional data are needed.

6 Conclusions

In this paper we presented a self-assessment framework that combines the ISO/IEC 29119 test process standard [4] with maturity levels from Test Improvement Model (TIM) [5]. The objective was to create a concept-level self-assessment tool to find development objectives for test process improvement and achieving conformance with the upcoming standard. The current limitation of the standard is that it does not offer support for adopting the standard in real-life software organizations, so the framework was defined to enable a maturity level-type self-assessment.

The self-assessment framework was developed and tested with pre-existing data from four organizations, and the results were confirmed with additional interviews with the profiled organizations. The results indicate that the framework could be developed to a useful self-assessment tool for organizations to compare their existing test processes against the standard and find process areas that could be improved. However, areas that require further development in the proof-of-concept version of the framework were also pointed out, and several concrete development proposals from case organizations were collected.

Overall, the results indicate that the framework proposed in this paper could become a feasible tool for defining process improvement objectives which promote the practices defined in the upcoming ISO/IEC 29119 standard. However, the current framework obviously needs further development and studies for validity. One possibility for the next logical step would be to conduct a qualitative study by applying the self-assessment framework in real-life organizations, and studying the applicability or relevance of the results the framework produces, when compared with other existing process development methods.

Acknowledgements. This study was partially supported by the ESPA-project (http://www.soberit.hut.fi/espa), funded by the Finnish Funding Agency for Technology and Innovation and by the companies mentioned in the project web site.

References

1. Myers, G.J.: The Art of Software Testing, 2nd edn. Wiley & Sons, Inc., New Jersey (2004)
2. Kit, E.: Software Testing in the Real World: Improving the Process. Addison-Wesley, Reading (1995)
3. Huang, L., Boehm, B.: How Much Software Quality Investment Is Enough: A Value-Based Approach. IEEE Software 23(5), 88–95 (2006), doi:10.1109/MS.2006.127
4. ISO/IEC: ISO/IEC WD 29119-2, Software and Systems Engineering - Software Testing - Part 2: Test Process (2010)
5. Ericson, T., Subotic, A., Ursing, S.: TIM - A Test Improvement Model. Software Testing, Verification & Reliability (STVR) 7(4), 229–246 (1997)
6. Test Maturity Model integration (TMMi), Version 3.1, TMMi Foundation, Ireland (2010)
7. CMMi Product Team: CMMI for Development, Version 1.3, Software Engineering Institute, Carnegie Mellon University, (2010), http://www.sei.cmu.edu/cmmi/
8. Jung, E.: A Test Process Improvement Model for Embedded Software Developments. In: Proc. of the 9th Internatinal Conference on Quality Software, Jeju, South Korea, August 24-25 (2009)
9. Oh, H., Choi, B., Han, H., Wong, W.E.: Optimizing Test Process Action Plans by Blending Testing Maturity Model and Design of Experiments. In: Proc. of the 8th International Conference on Quality Software, Oxford, UK, pp. 57–66 (2008), doi:10.1109/QSIC.2008.19 12.-13.8.2008
10. EU: SME Definition, European Commission (2003), http://www.ec.europa.eu/enterprise/policies/sme/facts-figures-analysis/sme-definition/index_en.htm
11. Karlström, D., Runeson, P., Nordén, S.: A minimal test practice framework for emerging software organizations. Software Testing, Verification and Reliability (STVR) 15(3), 145–166 (2005), doi:10.1002/stvr.317
12. ISO/IEC: ISO/IEC 15504-1, Information Technology - Process Assessment - Part 1: Concepts and Vocabulary (2002)
13. Kasurinen, J., Taipale, O., Smolander, K.: Software Test Automation in Practice: Empirical Observations. In: Advances in Software Engineering, Special Issue on Software Test Automation. Hindawi Publishing Co. (2010), doi: 10.1155/2010/620836
14. Kasurinen, J., Taipale, O., Smolander, K.: Test Case Selection and Prioritization: Risk-based or Design-based? In: Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM), Bolzano-Bozen, Italy, September 16-17 (2010), doi:10.1145/1852786.1852800
15. Burnstein, I., Suwanassart, T., Carlson, R.: Developing a testing maturity model for software test process evaluation and improvement. In: International Test Conference 1996 (ITC 1996) (1996), doi: /10.1109/TEST.1996.557106

Improving the Deployment of IT Service Management Processes: A Case Study

Marko Jäntti[1] and Julia Järvinen[2]

[1] University of Eastern Finland, School of Computing,
P.O Box 1627, 70211, Kuopio, Finland
marko.jantti@uef.fi
[2] SoftwareHouse Oy,
P.O Box 1188, 70211, Kuopio, Finland
julia.jarvinen@softwarehouse.fi

Abstract. Many IT service provider companies are interested in using the ISO/IEC 20000 service management standard to demonstrate their ability to provide high quality IT services to customers. However, the deployment of IT service management processes may cause several challenges. The research problem of this study is: How to improve the deployment of IT service management processes? The main contribution of this paper is to present experiences from a case study that focused on evaluating the deployment of an incident management process and a service desk tool.

Keywords: IT service management, deployment, ISO/IEC 20000, process.

1 Introduction

Thousands of IT organizations worldwide are struggling with the deployment of IT service management processes. It is relatively easy to model the future state of the IT service process and define goals, benefits, inputs, outputs, activities, metrics for the process. A major challenge is how to deploy the service management processes into daily IT operations and ensure that employees really follow the process activities.

A poorly planned process deployment can cause a chaos in the IT organization leading to massive resistance to change, decreased customer satisfaction and losing key developers. The number of organizations interested in the implementation of IT service management processes is rapidly growing. Thus, the deployment of IT service management processes is an important and actual research target. In order to improve IT service operations, IT organizations typically use some IT service management framework. The most widely used IT service management framework is IT Infrastructure Library (ITIL) that has gained a status of a de-facto standard.

In the ITIL version 2, IT service management is divided into two sections: service delivery processes and service support processes. ITIL version 3 was

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 37–48, 2011.

established to better address the service lifecycle thinking. The core books of the version 3 are Service Strategy [1], Service Design [2], Service Transition [3], Service Operation [4] and Continual Service Improvement [5]. We focus in this study on the deployment of service support processes, especially incident management.

The National Institute of Standards and Technology has estimated that software defects and problems annually cost around 60 billions the U.S. economy [6]. The rework in software projects (problem resolution and bug fixes) leads to higher software development and maintenance costs and and higher prices for IT services and products. This causes an enormous challenge for IT organizations' service support processes. In addition to the ITIL framework, there are several standards and frameworks that can be used to design, implement and deploy IT service management processes.

ITIL is a set of best practices for IT service management but is not a standard. There are two versions of ISO 20000 Part 1 that describes mandatory (shall) requirements for IT service management: ISO/IEC FDIS 20000-1:2010 Part 1: Service management system requirements [7] and ISO/IEC 20000-1:2005 Part 1: Specification for service management [8]. ISO/IEC 20000-1:2005 Part 2: Code of practice for service management [9] provides more detailed explanation of requirements.

ISO/IEC TR 20000-3:2009 Part 3 [10] can be used to define the scope of ISO/IEC 20000 certification, for example, IT organization may certify one IT service used by one customer. DTR 20000-4 Part 4: Process Reference Model defines name, context, purpose, outcomes and requirements traceability with ISO/IEC 2000 requirements for IT service management processes. COBIT [11] is an IT governance framework developed for IT process management with a strong focus on control.

Previous studies on IT service management have mainly focused on how to implement service management tools and service support processes. The main objective of *incident management* is to restore normal service operation as quickly as possible and minimise the adverse impact on business operations. However, few studies have dealt with IT service incident management. These studies have examined, for example, statistical prediction of incident management lifecycle [12] and requirements for an incident management tool [13].

Problem management is a process of managing problems and errors. Problem management aims to find the root cause of incidents and define a corrective solution or a process improvement [14]. Sandusky and Gasser [15] have explored the information coordination in the problem management process. Additionally, Kajko-Matsson [16] has presented a corrective maintenance maturity model for problem management.

The goal of *change management* is to ensure that standardised methods and procedures are used for efficient handling of changes [14]. Change management consists of the following activities: change logging and filtering, categorization, impact and resource assesment, approval, scheduling, building, testing, implementation and review. *Configuration management* aims to account for all the IT assets and configurations within the organisation and its services and provide accurate information on configuration items and their documentation to support all the other service management processes [14]. Ward et al. [17] discuss best-practice processes for change and configuration management developed by IBM for the CCMDB product.

Main goals of *release management* are to plan and oversee the successful rollout of software and related hardware [14]. Release management has been examined, for example, by Jansen and Bringkemper [18] who have investigated cost and value functions of release management. Kajko-Mattsson and Yulong [19] have defined a release management process model that integrates both vendor and acceptor sides. Sihvonen and Jäntti [20] present experiences from both patch management and release management.

Surprisingly few academic studies have examined the deployment of IT service mangement processes in IT service provider organizations. The study of Tan, Cater-Steel and Toleman [21] have identified six success factors in ITIL implementations: senior management support, project champion, relationships with tool vendors, change in corporate culture, project governance and execution and realisation of benefits. Similarly, a study of Pollard and Cater-Steel [22] considers top management support, training, virtual project teams, careful tool selection and use of external consults as key success factors in ITSM implementations.

1.1 Our Contribution

The main contribution of this paper is to

– explore the deployment of IT service management processes and tools in a Finnish IT service provider organization.
– provide lessons learnt from the deployment addressing challenges and problems.

The results of this study can be used by persons who are responsible for implementation or deployment of IT service management processes, IT quality managers, service managers, and customer service managers. The results can be used to support ITIL-based or ISO/IEC 20000-based process improvement work.

The rest of the paper is organized as follows. In Section 2, the research problem and methods are described. In Section 3, we describe the findings from interviews that focused on evaluating the ITSM process and tool improvement. Section 4 is the analysis of findings with lessons learnt. The discussion and the conclusions are given in Section 5.

2 Research Methods

The research problem of this study is: How to improve the deployment of IT service management processes? We used a case study research method with a single case organization to answer the research problem. According to Eisenhardt

[23], a case study is “a research strategy which focuses on understanding the dynamics present with single settings”.

Figure 1 describes the research settings of the case study. The unit of analysis in this study is an IT service provider organization that is implementing and deploying IT service management processes based on the ISO/IEC 20000 standard.

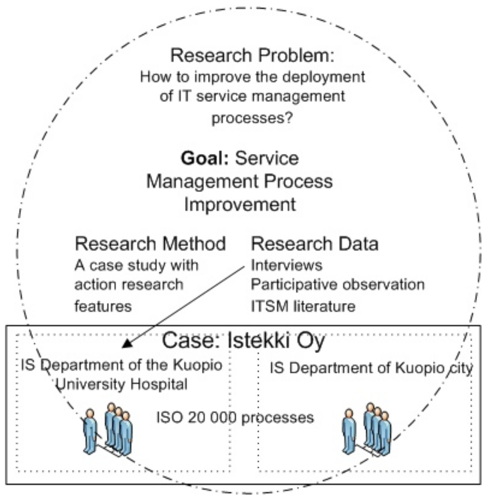


Fig. 1. The research settings of the case study

In addition to the ISO/IEC 20000 standard, we used the concepts of IT Infrastucture Library and Kotter’s 8-step Change Model [24] during the case study. The research problem was divided into 14 interview questions that followed the steps of 8-step Change Model.

2.1 The Case Organization and Data Collection Methods

Regarding the case selection, Istekki was a unique case because of ongoing ISO 20 000 certification process and service desk tool development process. Istekki Oy provides IT and medical technology services to two big customers: the city of Kuopio and Hospital District of Northern Savo. Istekki was founded in 2009 based on the merger of the IS department (Tekplus) of Hospital District of Northern Savo and the IS department of Kuopio city. Istekki started its operations in January 1st, 2010. The IT services of Istekki cover IT maintenance and support services, server and data center services, network and telecommunication services, ICT acquisition services, IT consulting, project and introduction services. Currently, Istekki has 160 full-time employees. The volume of support requests per year is around 35 000 in Tekplus. The case study work was carried out with Tekplus unit.

The case organization’s operations were divided into customer support (19 employees), application support (11 employees), technical support (21 employees), teletechnology (8 employees) and hardware acquisitions (2 employees). When this study began, there were three different ongoing improvement projects that have effects on customer support: an ISOQ project focusing on the ISO/IEC 20 000 certification, a Plussa project focusing on the improvement of the customer support processes and Feeniks project focusing on the service desk tool development.

In order to increase the quality of the case study, case study researchers can use three important principles of data collection [25]: 1) using multiple sources of evidence, 2) creating a case study database and 3) maintaining a chain of evidence. The following sources of evidence were used:

- Interviews focusing on the evaluation of process and tool improvement
- Participative observation (process improvement meetings in May-June)
- Documentation (ISOQ project plan, Plussa project plan, Feeniks system specification)
- Feeniks tool demo by an incident manager (30th June, 2009)
- Support process interfaces training for support process managers (25th August, 2009)
- Experiences on Feeniks project (seminar presentation on 9th September, 2009)

Interviews were conducted for 9 persons from 3 different units: customer support, application support, and technical support. Our goal was to interview around 10 persons. The resistance to change was expected to be highest in technical support where the deployment of incident management tool had started 8-9 months later than in customer support (January 2009). Interviewees were selected with random sampling. Some interviewees cancelled the interview by email and some refused to come without any notice. These persons were replaced with new interviewees.

2.2 Data Analysis

In the data analysis a within-case analysis technique [23] was used. The main purpose of within-case analysis is to examine case carefully as stand-alone entity before making any generalizations. The interview findings were collected to a summary report that described the experiences from the deployment process. A summary did not include interviewees' names and information how many persons were interviewed from each department. A summary report was submitted to an incident manager of the case organization. The summary was considered as a useful document that would help in the future process improvement work. Additionally, we used a research diary where the content of each meeting was recorded.

3 Improving the Deployment of IT Service Management Processes

KISMET (Keys to IT Service Management Excellence Technique) model was used as a roadmap for incident management process improvement work. The

KISMET model has been developed in the IT service management research projects of the University of Eastern Finland. The model consists of the following phases:

1. Create a process improvement infrastructure
2. Perform a process assessment
3. Plan process improvement actions
4. Improve/Implement the process based on ITSM practices
5. Deploy and introduce the process
6. Evaluate process improvement
7. Continuous process improvement

In this paper, the findings from the phase six (Evaluate process improvement) of the KISMET model are presented. The phases 1 to 5 are described in our previous research paper. Evaluation of process and tool improvement focused on service support processes, especially on incident management, and the deployment of the incident management tool, Feeniks. Next, the key findings are described.

3.1 Change Vision and Sense of Urgency

Questions: "When did you hear about ISO/IEC 20 000 project and Feeniks (Service Desk tool) project? Did you understand the connection between projects and their impact to your work tasks? Did you get enough information about projects? Did you get enough time to process information?"

Interviewees reported that they had heard about projects first time in 2008. The projects, especially ISO/IEC 20 000, were considered as distant. Some interviewees did not know how the ISO/IEC 20 000 project and Feeniks were related to each other. Many felt that the ISO/IEC 20 000 project does not affect to their jobs at all. The role of the service desk tool Feeniks was much clearer for interviewees.

According to our observations, interviewees were not very much interested in the ISO/IEC 20 000 standard and somebody had doubts on whether the standard is needed in the governance of Tekplus operations.

3.2 Coalition for Change

Questions: "Do you feel that the people who have carried out the change (management) have acted as a consistent group? Has the need for change and vision been clearly communicated to employees?"

The need for change and vision were understood quite well by interviewees but the ISO/IEC 20000 standard was considered distant. According to interviewees management had presented the change mainly as a unified group (employees had seen some managers having doubts on the benefits of change).

There was a general feeling that the old ticket system required changes and that the new system was built to correct the bottlenecks of the old system. People who were still waiting for deployment of Feenix had doubts whether the system results in any improvements to the workflow. Some were afraid of that the new system just disturbs the workflow.

3.3 Informing and Communication

Questions: "Has there been enough communication and informing between the change implementation team and employees? Do you know how to submit development ideas regarding the Feeniks tool?"

None of the interviewees reported that the communication and informing would not work. In general, the change was considered as "a thing that needed to be done". Regarding development ideas, interviewees knew to whom they should be submitted and also were confident that they will be analyzed and implemented.

3.4 Defining Milestones and Removing Barriers

Questions: "Have ISO/IEC 20000 project and Feeniks project been implemented in small steps enough? Are you aware of milestones? Have you been rewarded for achieving milestones?"

Interviewees considered the phased deployment of Feeniks as milestones. Some answers indicated the need for smaller milestones. However, part of the group were satisfied with the implementation. In team meetings, management had provided employees with positive feedback on implementing change-related tasks. New processes introduced in ISO/IEC 20 000 standard were considered challenging and interviewees stated that people might use old procedures instead of new procedures.

The incident manager of the case organization reported that the deployment of ISO/IEC 20000 processes was implemented in two main phases. The first phase included Service Management System, Service Level Management, Incident Management, Problem Management, Change Management, Configuration Management, Business Relationship Management and IT Financial Management (Phase 1.5). The second phase focused on Continuous Service Improvement, Management of New and Changed Services, Service Reporting, Release and Deployment Management, Capacity Management, Availability and Continuity Management, Security Management and Supplier Management.

3.5 Training

Questions: "Are you satisfied with the training (ITSM) you received? Would you have needed more specific or more general training? Do you feel that it is justified to change the names of existing business terms because the standard might use different types of terms? What do you think about the benefits of self-training tests for IT service management?"

Those interviewees who had received training were satisfied with the training. As expected, those who had not received any training on ITSM processes or Feeniks tool were not satisfied and were skeptic on the future trainings. Many interviewees mentioned the impact of summer holidays on learning. They felt that they forget the learnt issues during holidays. Some interviewees would have needed more interdepartment training (between customer support, application support, technical support departments) in addition to trainings related to employees' own tasks. These were considered useful for providing a clearer picture on how service requests move from one department to another and a "big" picture on the Feeniks tool. Additionally, the inter-department meetings were considered as a good method to share user experiences on Feeniks and process changes.

Interviewees saw the the terminology changes surprizingly positively. They had understood that unified terminology enables people to communicate and work more effectively. Self-training tests were generally considered as a positive thing. Some interviewees doubted whether this type of tests are necessary and whether they have time to execute tests. Interviewees told that employees should decide by themselves whether to do the test or not. Self-learning material and Finnish ITIL tests could increase the employees ITIL knowhow in a motivating way. Test results should remain confidential and should not be submitted to management.

4 Analysis

In the analysis phase of this study, we summarized case study findings in the form of lesson learned. A source for each lesson is presented in parentheses (I= Interviews, O= Observation, D= Documentation, ST= Seminars and trainings organized by the research group).

Lesson 1: Clarify employees the role of the ISO/IEC 20000 standard in service management (I, O, ST). The typical challenge in any organization is that most employees are not interested in the quality frameworks and standards and are not aware of their content or benefits. The management of the IT organization must frequently inform employees of IT service management benefits and quick-wins and tell how a single person's work is related to the big picture of service management. We also noticed that employees seem to understand the improvement of the ITSM tool better than improvement of ITSM processes.

Lesson 2: Process managers play a key role how successful deployment of the ITSM processes shall be (O, I). Top management cannot execute organizational changes, such as moving from a product-oriented or project-oriented culture to service-oriented culture, without the help of active process managers. The process managers that we met during the case study were all highly motivated to process improvement work and actively participated in the ITSM trainings. According to observations and interviews, process managers and management acted as a unified and consistent group in implementing and marketing the change. It is not enough that management tells how IT employees should change their procedures and workflows. Management should also show that they are willing to change. A concrete example is that each process manager should pass the ITIL Foundation course.

Lesson 3: Informing and communication decreases resistance of change towards IT service management (I). It is important to use multiple communications channels to ensure effective informing and communication on IT service management issues. Potential communication channels are organisation's newsletters, intranet sites, posters, team meetings and seminars [26]. In our case, informing and communication seemed to work well although the first seminars on IT service management were considered as too abstract by some interviewees. Interviews indicated that longer the time between the first presentation of change and change actions, more sceptic employees become. Frequent informing and open communication may remarkably decrease the resistance of change.

Lesson 4: IT service management trainings should be organized in small groups and should emphasize process interfaces (ST, I). Smaller groups in IT service management training result in more questions and discussions. It is recommendable that employees should not skip Awareness level trainings and should not go directly to Foundation level. Otherwise, the training might be considered too difficult with a dozens of new concepts. Trainings create overall awareness and understanding of the ITSM framework and terminology [26]. The number of ITIL certificates and trainings can be used as a metric to measure the learning and the competence of IT service management. Another important thing regarding ITSM process trainings is that they should focus more on lifecycle thinking by explaining how processes interact with each other (e.g. how incidents may lead opening a problem, how a problem is solved by sending a Request for Change for change management). Process interfaces are the points where the most information gaps exist.

Lesson 5: Understanding the key IT service management concepts is very important (ST, O, I). Understanding the key concepts (service requests, incidents, problems, requests for change, release packages) plays very important role in any ITSM improvement project. ITSM trainings should provide participants with practical examples from real life. Fortunately, many IT service management companies organize company-specific ITSM trainings where one can freely discuss about the current challenges and problems and the relation between existing business concepts and ITSM concepts. To our surprise, interviews showed that the attitudes towards new terminology were very positive. It would be good to create memos from concept-related discussions to share the knowledge to other teams and units that are very likely discussing the same things but perhaps in the different service context.

Lesson 6: Processes introduced in ISO/IEC 20 000 standard were considered challenging (I). The ISO/IEC 20 000 IT service management standard describes auditable requirements for 12 key processes. For many organizations, a part of these processes are completely new, for example, problem management, and a part have existed but perhaps with a different name. One reason why ISO/IEC 20 000 standard is considered challenging is that solving a customer support request might require an activity chain where 7-10 different processes participate in. Employees shall very likely be confused when a consultant goes through the whole lifecycle of a support request. The role of the ISO/IEC 20 000 standard in relation to IT Infrastructure Library should be explained already in IT service management Awareness trainings. However, the most employees do not need formal ISO/IEC 20 000 Foundation training but a compact information package that also explains the benefits of using the standard.

Lesson 7: Use milestones and a phased approach in the deployment of ISO/IEC 20 000 standard and a service desk tool (D, I). The case organization used a phased approach in the deployment of ISO/IEC 20 000 standard and a service desk tool. The deployment of ISO/IEC 20 000 standard was divided into two main phases. It might be advisable to use more phases (3-4) because the process implementation and deployment may take years. The deployment of a service desk tool was phased according to 3 teams. Those teams that have to wait for a longer time to deployment need special care and frequent information on the deployment issues to decrease the number of rumours.

Described lessons learnt above are not presented in a priority order, nor in a chronological order. The interview findings were reported as a case study summary report describing the experiences from the deployment process. The summary report was later used in a Master's thesis work at the School of Computing [27]. A summary did not include interviewees' names and information how many persons were interviewed from each department. Thus, the management of case organization did not know who actually participated in interviews. A summary report was submitted to an incident manager of the case organization. The research team had worked before interviews with the incident manager to define the incident management process. She considered the summary report as a useful document that would support the future process improvement work by highlighting the issues that need to be taken account in the deployment of other ISO/IEC 20000 processes.

5 Discussion and Conclusions

This paper aimed to answer the following research problem: How to improve the deployment of IT service management processes? The main contribution of this study was to explore the deployment of an incident management process and a service desk tool in a IT service provider organization in Finland and to provide lessons learnt from the deployment. Interviews that collected feedback on the deployment were used as a main data collection method. Additionally, we used participative observation, documents, and seminars and trainings as data sources.

Our study resulted in the following key findings: 1) employees do not see clearly the benefits of the ISO/IEC 20000 standard, 2) IT service management trainings should focus more on the process interfaces and should be organized in smaller groups, 3) understanding a key ITSM concepts is crucial and 4) the deployment of ISO/IEC 20000 requires a phased approach with smaller milestones.

Every case study contains certain limitations. First, data were collected by using qualitative research methods from one case organization. A number of interviewees could have been larger and could have included also people from management side. Second, we cannot generalize the results of a single study to other IT service provider organizations but our study resulted in a valuable information on the deployment of ISO/IEC 20000 processes. Third, the case organization was not randomly selected but selected from the pool of the project's industrial partners. Thus, we had an easy access to the case.

To conclude, the implementing and defining of ISO/IEC 20000 processes seems to be much easier than the transition of the process to the daily work practices. More academic studies are needed to explore the deployment of the ISO/IEC 20000 standard in IT service provider organizations.

Acknowledgment. This paper is based on research in KISMET and MaISSI projects funded by the National Technology Agency TEKES, European Regional Development Fund (ERDF), and industrial partners.

References

1. Office of Government Commerce: ITIL Service Strategy. The Stationary Office, UK (2007)
2. OGC: ITIL Service Design. The Stationary Office, UK (2007)
3. Office of Government Commerce: ITIL Service Transition. The Stationary Office, UK (2007)
4. Office of Government Commerce: ITIL Service Operation. The Stationary Office, UK (2007)
5. Office of Government Commerce: ITIL Continual Service Improvement. The Stationary Office, UK (2007)
6. Geralds, J.: Software bugs cost billions. IT Week (2002), http://www.v3.co.uk/v3-uk/news/1973196/software-bugs-cost-billions (last access: April 21, 2011)
7. ISO/IEC: ISO/IEC FDIS 20000:1. ISO/IEC JTC 1 Secretariat: ANSI (2010)
8. ISO/IEC 20000: IT Service Management, Part 1: Specification for service management. ISO/IEC JTC1/SC7 Secretariat (2005)
9. ISO/IEC 20000: IT Service Management, Part 2: Code of practice for service management. ISO/IEC JTC1/SC7 Secretariat (2005)
10. ISO/IEC: ISO/IEC TR 20000-3 Information Technology - Service Management - Guidance on scope definition and applicability of ISO/IEC 20000-1. ISO/IEC JTC1/SC7 Secretariat (2010)
11. COBIT 4.1: Control Objectives for Information and related Technology: COBIT 4.1. IT Governance Institute (2007)
12. Caldeira, J., Brito e Abreu, F.B.: Influential factors on incident management: Lessons learned from a large sample of products in operation. In: Jedlitschka, A., Salo, O. (eds.) PROFES 2008. LNCS, vol. 5089, pp. 330–344. Springer, Heidelberg (2008)
13. Jäntti, M.: Defining requirements for an incident management system: A case study. In: Proceedings of the Fourth International Conference on Systems (ICONS 2009). IEEE Computer Society, Los Alamitos (2009)
14. OGC: ITIL Service Support. The Stationary Office, UK (2002)
15. Sandusky, R.J., Gasser, L.: Negotiation and the coordination of information and activity in distributed software problem management. In: GROUP 2005: Proceedings of the 2005 International ACM SIGGROUP Conference on Supporting Group Work, pp. 187–196. ACM Press, New York (2005)

16. Kajko-Mattsson, M.: Corrective maintenance maturity model: Problem management. In: ICSM 2002: Proceedings of the International Conference on Software Maintenance (ICSM 2002), p. 486. IEEE Computer Society, Washington, DC, USA (2002)
17. Ward, C., Aggarwal, V., Buco, M., Olsson, E., Weinberger, S.: Integrated change and configuration management. IBM Syst. J. 46, 459–478 (2006)
18. Jansen, S., Brinkkemper, S.: Ten misconceptions about product software release management explained using update cost/value functions. In: Proceedings of the International Workshop on Software Product Management, pp. 44–50. IEEE Computer Society, Washington, DC, USA (2006)
19. Kajko-Mattsson, M., Yulong, F.: Outlining a model of a release management process. J. Integr. Des. Process Sci. 9, 13–25 (2005)
20. Sihvonen, H.M., Jantti, M.: Improving release and patch management processes: An empirical case study on process challenges. In: Proceedings of the International Conference on Software Engineering Advances (ICSEA 2009), pp. 232–237 (2010)
21. Tan, W.G., Cater-Steel, A., Toleman, M.: Implementing it service management: A case study focussing on critical success factors. Journal of Computer Information Systems 50(2) (2009)
22. Pollard, C., Cater-Steel, A.: Justifications, strategies, and critical success factors in successful itil implementations in u.s. and australian companies: An exploratory study. Information Systems Management 26(2), 164–175 (2009)
23. Eisenhardt, K.: Building theories from case study research. Academy of Management Review 14, 532–550 (1989)
24. Kotter, J.: Leading Change. Harvard Business School Press, Boston (1996)
25. Yin, R.: Case Study Research: Design and Methods. Sage Publishing, Beverly Hills (1994)
26. Office of Government Commerce: Planning to Implement Service Management. The Stationary Office, UK (2002)
27. Järvinen, J.: Challenges and methods in the deployment of it service management processes: Case kys tekplus. Master's thesis, University of Eastern Finland, School of Computing (2011)

A Survey on the Application of the V-Modell XT in German Government Agencies

Marco Kuhrmann[1], Christian Lange[2], and André Schnackenburg[2]

[1] Technische Universität München, Institut für Informatik,
85748 Garching, Germany
kuhrmann@in.tum.de

[2] Federal Office of Administration, Office for Information Technology,
50735 Cologne, Germany
{Andre.Schnackenburg,Christian.Lange}@bva.bund.de

Abstract. The V-Modell XT is the standard software development process for IT-projects in the German government. For federal agencies, this process is mandatory to manage internal IT-projects, as well as to coordinate projects of third-party suppliers. The V-Modell XT is available since 2005 and already in use at several German agencies to organize and manage projects. In this paper, we present a survey that – 5 years after the release of the V-Modell XT – contributes an insight into the usage style, ratings, and tempers of project managers working with the new process. The survey consists of two stages: the first stage narrows down the application domain and allows for initial observations, followed by a second stage, which allows for quantified assertions. We also summarize the core requirements to extend the visibility of the process and to improve the quality of its application.

Keywords: Government Agencies, Process Application, V-Modell XT.

1 Introduction

The German government is one of the largest contracting authorities for software engineering projects in Germany. Almost 30% of the overall German IT-project budget is spent in governmental projects. Since the budget for IT-projects is provided by taxes, the government agencies are required to efficiently manage their budget in a law-conformant manner. Nonetheless, the German Federal Court of Auditors stated in their annual reports [3] that a vast number of projects were running out of budget or missing project deadlines. One reason is the complex project constellations in which governmental agencies have to setup IT-projects, either to act as client in tendering and procurement, or to develop complex IT-solutions for themselves. This requires an organized, legally sound, reproducible, and transparent process for tendering procurement and development.

As a consequence to this situation, the government seeks for the application of an adequate, i.e., flexible and structured IT development process. The V-Modell XT [4] was released in 2005 as the standard IT-development process

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 49–60, 2011.

for the German agencies. Its application is mandatory with respect to advertisement and procurement projects as for (internal) development projects. The process assists projects with a standardized organization, best practices, references to regulations or resolutions, and other standards. Suppliers are pledged by contract to setup their projects and to develop IT-systems with respect to the V-Modell XT. The process model itself contains several sub-processes. The sub-processes support the client during procurement, the supplier during the development, a project setting without contracting (e.g. for organization-internal development), and the improvement of development processes. To support the flexible customization of the comprehensive process to concrete project settings, the V-Modell XT offers a sophisticated and tool-supported tailoring model. This tailoring model is in particular realized on basis of a machine-readable metamodel [11], which is available as an XML-schema. Hence, the process can be supported by a various set of tools through its overall life cycle [6].

1.1 Research Objectives and Context

The Federal Office of Administration (Office for Information Technology, BIT[1]) is responsible for the coordination of the application and improvement of the process model as well as for the (generic) standard model and customized variants. The BIT also provides assistance for the application of the process.

In consequence, BIT as coordinating office is interested in learning which agencies already use the V-Modell XT, to what extent they use it, and how well the V-Modell XT fits their intended goals, what in turn formulates our particular research objective. The outcome of the survey gives BIT the opportunity to further improve the process and, thus, align further improvements with the particular needs of the agencies.

1.2 Related Work

The optimization of (development) processes is a frequently discussed topic with valuable contributions, e.g., CMMI [8] or Spice [7], used to determine the maturity of an organization. Regarding the analysis of development projects, international studies are available, such as the (heavily criticized) Chaos report [12] or corresponding studies that are specific for Germany, e.g. the Success study [2] or the IOSE survey [6]. All those contributions focus on organizations' maturity or the application of processes in general.

However, to the best of our knowledge there do not exist any reproducible studies, which investigate experiences made with development process models known as rich/structured or so-called "heavy-weight" processes. If anything, we can find reports that generally discuss advantages and experiences in the application of agile methods in comparison to rich process models, e.g., considering team structures, embedding the client into the project, or the handling of changes. Only few of those studies, such as [10], provide an empirical basis to

[1] German: BIT – Bundesstelle für Informationstechnik.

measure agile methods' efficiency. However, since agile methods do not meet the requirements of federal agencies in general, we close a gap in literature with the survey at hand.

Outline. The remainder of the paper is structured as follows. In Sect. 2, we discuss the study design. Section 3 presents our study results, before concluding our paper in Sect. 4.

2 Case Study Design

In this section, we present the design of the case study. The case study considers two stages: (1) a subject selection stage to develop and validate a set of questions, and (2) the elaboration of quantifiable and representative data gathered with a second questionnaire that considers a wider audience. The first stage of the study begun end of 2008 and was finished in spring 2009 [9]. Based on those results, the second stage was conducted in the year 2010. In the following, we present the case study's background and give insights into the used questionnaires. We describe how the particular questionnaires were designed, how they were structured, and how we identified appropriate subjects.

Background. As coordination office it is of special interest for the Federal Office of Administration to know: (1) Who implements the V-Modell XT in projects? (2) What is the implementation style? (3) Which experiences were made? The goal of the studies is, thus, to learn whether or not the V-Modell XT meets the government offices? requirements and what are possible improvements. Findings from the study will be considered during the further development of the process.

2.1 Survey Stage 1 – 2009 (Subject Selection)

The subject selection stage of the survey was designed as a qualitative study. The primary goals were to (1) identify the subjects and (2) to develop and validate the set of relevant questions. The first stage consists of three steps: (1) A survey by phone to identify considerable projects. (2) Answering of a questionnaire that contains 58 questions for a broad data collection. (3) Semi-structured interviews to get detailed information.

The questionnaire (step 2) was designed according to the Goal-Question-Metric model [1]. The relevant research questions for this stage are:

1. Who implements the V-Modell XT and how?
2. What was the motivation to apply the V-Modell XT?
3. What are the payoffs and which problems can be identified?

Subjects. To identify candidates for the questionnaire the following target groups in departments/offices that are involved in IT-projects were contacted:

- IT-project managers or consultants
- IT managers/coordinators of government agencies that develop IT systems
- Student lists from the internal advanced training center
- Servants who are certified according to the V-Modell XT certification system

Structure of the Questionnaire. The structure of the questionnaire provided to the subjects, is illustrated in Fig. 1. Each question is assigned to one of the areas depicted in Fig. 1. All question metrics were discretized (Yes/No, and on a Likert-scale from 1 to 5, which was represented to the participants from "worse": -2 to "better": 2). In addition to those closed questions, we defined open questions that gave the participating subjects the possibility to answer free text.

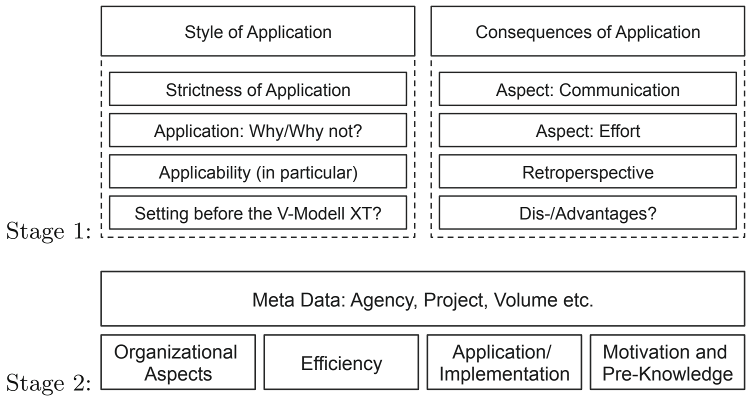


Fig. 1. Coarse structure of the questionnaires

2.2 Survey Stage 2 – 2010 (Data Collection)

The first stage was used to narrow down the envisioned domain and to identify concrete questions for the second survey stage. The resulting second questionnaire contains a set of 22 questions. The research questions were basically the same as for the first stage. We adjusted the focus of the study with respect to the findings of the first stage. Due to the findings of the first stage, we asked again for the opinions of the participants, and how the process model is visible to the organization.

Subjects. For the second questionnaire, there was no explicit selection of candidates. The questionnaire was promoted and distributed by the German IT-Council[2]. The IT directors of all agencies that are represented in that council internally promoted the survey. A memo, which was provided to the council, was forwarded to the responsible teams. Depending on the organization structure of corresponding agencies, the candidates were division managers, project managers or officials in general.

Structure of the Questionnaire. According to the results of the first stage, the second one was focused on specific aspects. Furthermore the second study was designed to be of quantitative nature. The structure of the questionnaire is shown in Fig. 1. Same as for the first questionnaire, all question metrics are discretized. For selected questions, there was the option also to answer as free text.

[2] The German IT-Council is a board that consists of all responsible managers of all resorts of the government agencies and ministries.

3 Results

This section, we present selected results of the surveys. We first describe the results of the two questionnaires before discussing those results in detail.

3.1 Results from Stage 1

The first step of stage 1 was the determination of the agencies – as well on the federal as on the state level – that should be taken into account. Telephone interviews were conducted to establish first contacts and to find out, whether or not a particular agency meets the criteria of maturity to participate in the survey. Table 1 shows the results of the interviews. As a whole, 30 agencies were contacted, leading to 27 who would provide us with information about their development activities. Half of the contacted agencies already applied the V-Modell XT. However, not all agencies could answer all questions. From the interviews, 6 agencies were selected for the questionnaire to go into depth.

Table 1. Participants – contacting and selection; first information was collected to determine whether or not the contact is qualified for the questionnaire

Criteria/Questions	Number	
Telephone interview with project managers from several agencies	30	
Federal agencies		*28*
State agencies		*2*
Projects that apply V-Modell XT	16	
V-Modell XT is not applied	11	
An individual process model is applied		*2*
Complete rejection of the V-Modell XT		*5*
Number of evaluation (overall)	8	
Selected for the questionnaire		***6***
Selected pilot projects from CIT		*2*

Questionnaire Results. The questionnaire contains questions related to the organization itself, the project types in general, and concrete project configurations. All considered projects had a team size of 4 to 10 people. The project duration was between 12 and 60 man months. The projects we covered were mainly medium-sized projects.

In the second part of the questionnaire, we collected information of the projects themselves and their presentation in the executing organization (see Table 2). The first outcome is that the teams were mostly trained. Control questions showed that most of the trained teams were self-motivated to a high degree and had organized trainings by themselves. The results show that the process was mainly used "as-is" and that project-specific deviations from the (generic) process were scarcely made. Whether or not the process was correctly applied was also barely monitored. In addition, 2 of 6 organizations have already established a responsible role contact for process-related questions.

The last considered aspect was the (subjective) expert opinion of the project managers when applying the new process (Table 3). Hence, we are aware that we cannot get empirical sound data, whereby we captured the data in the Likert-scale and calculated the mean value to estimate resulting trends. Especially the additional effort was seen critical. Basically, the manager's opinion could be considered as "neutral", but they stated a slightly trend to increased effort. With extra control questions, we could find out that documenting the project results and reporting did not cause the additional effort.

Table 2. Application context – did the project team members have pre-knowledge, how were the projects operated and what is the process's position in the organization

Criteria/Questions	Yes/Total
Did the project team have knowledge about the V-Modell XT?	4/6
Was the V-Modell XT tailored according to project parameters?	1/6
Was the "correct" application of the process monitored?	1/6
Does the organization have a responsible for the V-Modell XT?	2/6

Table 3. Range of opinions according to the application in concrete projects

Criteria/Questions	Evaluation
Delivery quality and functionality is:	slightly better
Compared to the projects without V-Modell XT the aspect (*) is:	
Communication	better
Additional effort	neutral
Flexibility	neutral
Extent of regulations in the V-Modell XT is:	worse

Tentative Conclusion. From this first stage, we can draw the following conclusion. In 2009, the application of the V-Modell XT was still in early stages. We found out that the (correct) application without expert coaching usually leads to additional efforts. Self-studying is not enough, even for professionals. In addition, detailed knowledge about the V-Modell XT was only sporadically available in the agencies.

3.2 Results from Stage 2

The second questionnaire was conducted less than a year after the first one. With respect to the kind of application and the temper of the project teams, we asked questions with respect to training and individual evaluations to be able to infer trends. The German IT-Council – that is a central committee for all government agencies – promoted the questionnaire. As a whole, 201 participants begun filling in the questionnaire. Finally, we could get 29 complete data sets from 16 agencies, which is a share of 14.4% of valid data sets.

Questionnaire Results. Same as for the first stage, we asked the participants for some meta-data (optional), e.g., the project size. Table 4 shows the classification of the projects. While mainly small and medium-sized projects were considered in the first stage, more than half of the projects under consideration were categorized as "Large".

Table 4. Project size summarized – 58.6% of the projects are categorized as large and very large projects (project duration up to 2,000 man months)

Project size	Volume (in man months)	Number
Small	1 – 6	4
Medium	7 – 20	6
Large	21 – 100	9
Very large	$\geq$ 101 man months	8
(no information)		2

Table 5. Application of the V-Modell XT – How is the process's position in the organization? Is the application monitored? (detailed view: project and organization level)

Criteria/Questions	Yes/Total (projects)	Yes/Total (organization)
Is the V-Modell XT mandatory to apply for you?	24/29	11/16
Was the "correct" application of the process monitored?	10/29	–
Does the organization have a responsible for the V-Modell XT?	10/29	4/16

The V-Modell XT was stated to be the organizations' standard development process in 24 of 29 projects (83%). Taking into account that an agency could participate with more than one project, we evaluated the engagement of the application on the organizational level (Table 5). The result show that for 11 of 16 (68.75%) of the participating agencies, the V-Modell XT is the mandatory process. Considering the consequences, i.e., the need for a responsible person for the process, the organizations are not that mature. Only 4 of the participating 16 agencies (25%) still established a process owner. Finally, 10 of 29 (34.5%) of the projects monitor the "correct" application.

From the first stage of the study, we learned about the subjective opinion of the project managers and teams, and their satisfaction with the new process. For the second stage, we asked again the corresponding questions (using the same metrics). The results are summarized in Table 6. Hence, since we have to consider projects with different objectives and different volumes, we refined this evaluation accordingly.

Exemplarily, we present the refined results for software development projects and the project roles (client, contractor, internal development) the corresponding projects were in.

Table 6. Range of opinions according to the application in concrete projects

Criteria/Questions	Evaluation
Compared to the projects without V-Modell XT the aspect (*) is:	
Communication	slightly better (0.35)
Additional effort	slightly worse (-0.52)
Flexibility	neutral (0.04)

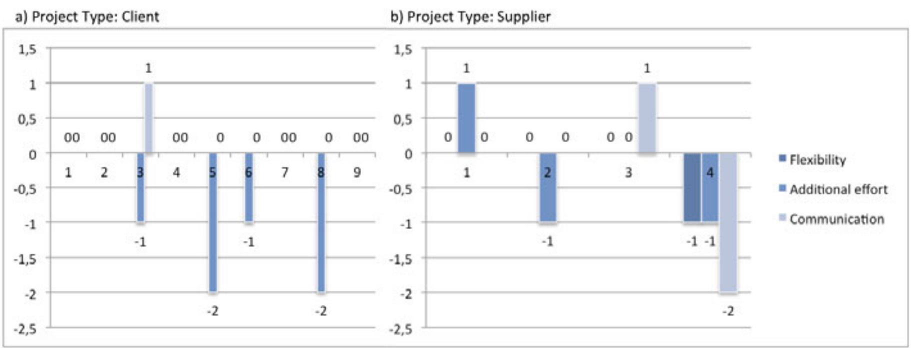


Fig. 2. Application in projects of type: a) client (objective: procurement of software systems, contracting, supplier/contractor management) and b) supplier/contractor (objective: bidding, software/system development, system maintenance)

The evaluation of the client projects is shown in Fig. 2a. We observe that the client projects stated additional effort (4 of 9), while the other criteria are evaluated as neutral. The evaluation of the contractor/supplier projects (objective: software development) is shown Fig. 2b. The results are diverse, so we cannot make sound conclusions, not even speculations.

The evaluation of the internal development projects[3], which is shown in Fig. 3, shows the most interesting results. In that setting we observe improvements with regards to communication and additional effort. Two of the projects stated additional effort and one project stated loss of flexibility.

The last aspect to present is the opinion of the participants, whether or not the new process fits their requirements. 20 of 29 (68.97%) participants judged that the V-Modell XT was appropriate for their projects. 23 of 29 (79.31%) of the participants concluded that the V-Modell XT is capable to implement the preferred development process (Table 7).

Tentative Conclusion. In 2010, the application of the V-Modell XT was established. We can state the first effects of the application, especially with respect to internal development projects. We observe a trend to establish the V-Modell

[3] The V-Modell XT supports internal development projects without contracting between operating/special departments and (internal) IT service vendors.

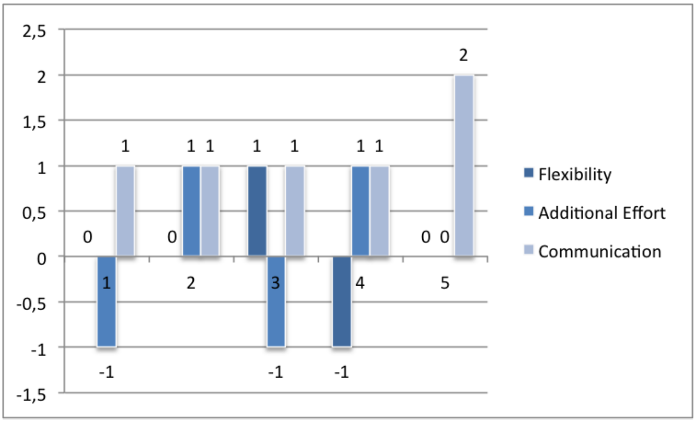


Fig. 3. Application in internal projects (objective: software development, communication with operating/special departments)

Table 7. Evaluation of the participants – Fits the V-Modell XT the requirements? The V-Modell XT is appropriate to capture all relevant project settings in the government context

Criteria/Questions	Yes/Total
Was the V-Modell XT appropriate to describe and assist the project?	20/29
Is the V-Modell XT capable to implement your preferred process?	23/29

XT on the organizational level and to also establish process owners who are responsible for the process. The V-Modell XT is visible.

3.3 Interpretation

The studies lead to results, which might be unexpected on first sight, but which also require background knowledge to understand them. In the following, we interpret the results and give further background information necessary for a deeper understanding of the results. Since the German government is one of the largest contracting authorities for IT-projects, the first aspect to be considered is the difficulty of finding appropriate projects for the survey. The difficulties come from the kind of how projects are founded and organized. Usually, projects are accompanying day-to-day work, except for large and strategic projects that over-span several agencies. The ordinary project is staffed with part-time teams, which is the reason for 10 people in a project with durations of 20-25 months and a volume of 15-20 man months. Another aspect that needs to be considered is that projects are often influenced by political situations and developments. In contrast to development projects found in pure industrial contexts, this can lead to projects being paused and restarted, redefined, or completely canceled if the circumstances require re-prioritization.

Additional Effort 1: To efficiently manage such projects, the Federal Court of Auditors requires the agencies to organize their projects according to a structured process model. The IT-Council defined the V-Modell XT as the standard process. This is the possible explanation for the increased number of projects that use the V-Modell XT and that recognize it as the mandatory process (83%). On the other hand, the data show that only 25% of the agencies have established a process owner, who is responsible for the process. Furthermore, the data shows that only 34.5% of the projects monitor the "correct" application. With respect to the first stage, where most of the participants stated they selected the process by self-motivation, we interpret the data as follows: Based on the decisions of the Federal Court of Auditors and the IT-Council many agencies have passed a resolution to introduce the process. The resulting consequences (the need for process owners, or process/project monitoring) were not considered adequately. This is one possible explanation of the additional effort that is reported by the studies' participants. The day-to-day work is interrupted by additional work that is required to fulfill the resolution.

Additional Effort 2: We observe that the client projects stated additional effort (4 of 9, ref. Fig. 2a), while the other criteria are evaluated as neutral. Having background knowledge of the way government agencies usually work in the client role, a possible explanation could be as follows: Additional effort occurs, because the V-Modell XT shifts requirements engineering back to the client (basic idea: 'Only the client knows what he needs.'). We assume this to be the reason for the reported additional effort. This has, however, to be further examined in the future.

Improvement of Communication: Projects, which we categorized as internal development projects, reported improved communication. Important to know is that the internal development was more or less structured. Even a telephone call was an established procedure to propose a bug or to request a new feature. This often resulted in ad-hoc developments, non-documented changes, and software of poor quality or – in the worst case – non-working software. Same as for client projects, the V-Modell XT regulates the interaction in such project settings. There is an interaction style established for project management as well as for the definition of responsibilities. We assume that those communication improvements result from the defined structures that first of all state who has to provide which information to whom.

Trends. We can observe a significant trend regarding the "visibility" of the process. The V-Modell XT is introduced as the standard development process by resolution. The consequences are an increased demand for training and coaching. Since more and more agencies establish the V-Modell XT, the number of trainings grows. Besides the trainings, we observe another trend: The number of agency-specific V-Modell XT variants also grows. During the last three years, at least 5 agencies developed a specialized variant of the process. Those variants cover the particular needs of an agency by meeting the overall regulations. At the same time, the specialized process is closer to the habitual style of work (terminology, templates etc.) of the agencies.

Threats to Validity. The following aspects we consider critical with regards to validity: The results are based on a sample. Furthermore, the information collected in that survey is based on self-disclosures. Nevertheless, the data can be generalized. The introduction of the V-Modell XT is mainly done by resolutions. This procedure is basically the same for all governmental agencies in Germany. Thus, we did not ask for the efficiency of concrete development methods but for the projects at a whole, we also assume similar outcomes for different process models. The reasons are mainly the organizational aspects of the process introduction, such as management commitment, trainings etc.

4 Conclusion and Future Work

The V-Modell XT is the standard development process model for IT-projects in the German governmental agencies. It contains best practices, and references to regulations and standards that support governmental agencies in tendering and developing IT-systems. The V-Modell XT is a generic process model that supports different project settings and needs to be tailored according to a specific scenario.

In the presented survey, we conducted studies according to the application of the V-Modell XT in the governmental agencies and the satisfaction of the project teams. The survey was performed stepwise over a period of two years. The first stage was the subject selection, where we selected the subjects, and developed and validated the questions. In the second stage we collected the data of 29 projects from 16 agencies.

The findings show that the V-Modell XT has received increasing attention at governmental offices during the last years. The number of projects that use this process grew and, accordingly, the number of trained people and agency-specific V-Modell XT variants. The paper at hands focuses on the attitudes of the project teams. We observe that there are improvements in the projects, especially in project types that were not regulated before. So, for example projects of type "internal", which means there is no third-party supplier involved in a software development project, benefit from the V-Modell XT. Communication, as one of the key factors for successful projects, improved as the process introduces interaction styles. Another aspect that became obvious during the studies is that comprehensive process models are not a matter of self-studying. Extensive coaching and training is required to qualify the IT-personnel. Training programs are established since 2005 and we notice the demand by the wait lists and rising number of extra trainings.

The initial questions of the survey have been answered here. However, a few of the presented conclusions could not have been drawn purely on the collected data but rather on the extensive background knowledge of the authors on governmental institutions. We have learned who is using the V-Modell XT and in what manner. We have also gathered some new hypotheses for future studies from our participants, e.g., reasons for the extra time and effort needed in procurement and contracting projects. Thus, new questions have been generated for a successive survey. To this end, the data basis has to be extended and refined. Subsequently, details about specific aspects have to be raised in further interviews. The results of such a survey would be an important contribution to the enhancement of the government-tailored "V-Modell XT Bund" [5].

References

1. Basili, V.R., Rombach, H.D.: The TAME Project. Towards Improvement-oriented Software Environments. IEEE Transactions on Software Engineering 14(6) (1998)
2. Buschermöhle, R., Eekhoff, H., Josko, B.: SUCCESS 2006 — **SUCC**ess and failur**E** of hard- and **S**oftware project**S** (Erfolgs- und Misserfolgsfaktoren bei der Durchführung von Hard- und Software-Entwicklungsprojekten in Deutschland). BIS-Verlag (2006)
3. Federal Court of Auditors. Annual Report on Federal Financial Management – further audit findings. Technical report, Federal Court of Auditors (2010)
4. Federal Ministry of the Interior. V-Modell XT. Current release: 1.3 (English version available), http://www.v-modell-xt.de
5. Federal Office for Administrative Affairs. V-Modell XT Bund. Current release: 1.0 (basis V-Modell XT 1.3), http://www.bva.bund.de/v-modell-xt
6. Fritzsche, M., Keil, P.: Kategorisierung etablierter Vorgehensmodelle und ihre Verbreitung in der deutschen Software-Industrie. Technical report, Technische Universität München (2007)
7. ISO. Information technology – Process Assessment — Part 5: An exemplar Process Assessment Model. ISO/IEC 15504-5:2006 (Spice). Technical report, International Organisation for Standardization (2006)
8. Kneuper, R.: Improving Software and Systems Development Processes Using Capability Maturity Model Integration (CMMI-Dev). Rooky Nook (2008)
9. Lange, C., Kuhrmann, M., Schnackenburg, A.: Einsatz des V-Modell XT in der Behördenpraxis — Analyse, Auswertung, Verbesserung. In: Proceedings of Software & Systems Engineering Essentials (2009)
10. Mah, M.: How Agile Projects Measure Up, and What This Means to You. Technical report, Cutter Consortium Business-IT Strategies (2010)
11. Ternité, T., Kuhrmann, M.: Das V-Modell XT Metamodell. Technical report, Technische Universität München (2009)
12. The Standish Group. Chaos Reports, http://www.standishgroup.com

Improving Verification & Validation in the Medical Device Domain

M.S. Sivakumar, Valentine Casey, Fergal McCaffery, and Gerry Coleman

Regulated Software Research Group,
Dundalk Institute of Technology & Lero, Ireland
{Sivakumar.Madhavakuruppu,Val.Casey,Fergal.McCaffery,
Gerry.Coleman}@dkit.ie

Abstract. The benefits of effective verification and validation activities in the medical device domain include increased usability and reliability, decreased failure rate and recalls and reduced risks to patients and users. Though there is guidance on verification and validation in multiple standards in the medical device domain, these are difficult for the manufacturer to implement, as there is no consolidated information on how they can be successfully achieved. The paper is intended to highlight three major areas for improvement in the medical device software development domain. This research is based on an analysis of available literature in the field of verification and validation in generic software development, safety-critical and medical device software domains. Additionally, we also performed a review of the standards and process improvement models available in these domains.

Keywords: Medical device standards, Medical device software verification and validation, V&V, Medical device software process assessment and improvement, Medi SPICE.

1 Introduction

Verification and Validation (V&V) are amongst the most important activities in the software development lifecycle and consume up to 50% of project development time [1], [2] and up to 50% of the total cost [3].

The concepts of V&V emerged in the late 1960s and 1970s as the use of software in military and nuclear power systems increased. Initially, individual projects began to develop their own standards to address the need for V&V. Then government and industry began to develop V&V standards so they would have a specified approach for contract procurement and for monitoring the technical performance of V&V efforts. V&V standards and guidelines serve large, heterogeneous communities and are applicable to many types of software. These include:

1. The US Air Force's AFSCIAFLC 600-5, Software Independent Verification and Validation [4]

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 61–71, 2011.

2. The American Nuclear Society's ANS 10.4, Guidelines for the Verification and Validation of Scientific and Engineering Computer Programs for the Nuclear Industry [4]
3. The NASA Jet Propulsion Laboratory's JPL D-576, Independent Verification and Validation of Computer Software: Methodology [4]
4. Food and Drug Administration (FDA) Center for Devices and Radiological Health (2002) General Principles of Software Validation (GPSV) [5] for medical devices

While both V&V play a key role in software development, there is a level of ambiguity in the use of these terms. This is evident in the literature as the following definitions demonstrate:

- Berling, and Thelin argue that verification aims at checking that the system as a whole works according to its specifications and validation aims at checking that the system behaves according to the customers' intentions [6]
- Arthur, Groner, Hayhurst, and Holloway cite in one of their papers that verification refers to the process of examining each development phase to ensure that the output of a particular phase satisfies all the pertinent requirements of the previous phase, is internally acceptable, and can support the development effort in the next phase. Validation, on the other hand, is an activity primarily concerned with software testing. During validation you execute the system and compare the test results to the requirements [7]
- Wallace and Fujii state that verification involves evaluating software during each lifecycle phase to ensure that it meets the requirements set forth in the previous phase and validation involves testing software or its specification at the end of the development effort to ensure that it meets its requirements (that it does what it is supposed to). They go on to say that while 'verification" and "validation" have separate definitions, you can derive the maximum benefit by using them synergistically and treating "V&V" as an integrated definition. [4]

As these definitions highlight, the terms verification and validation are often used interchangeably whereas in the strict sense the objectives of these two processes are different. While they may both employ the same techniques or environment while performing these processes.

The main activities of verification and validation are reviews and testing. Software testing cannot be conducted until the software product is built. Since faults need to be found early, software inspections are conducted before the product has been implemented [6]. The difference between validation and verification can be explained by looking at the purpose of the tests performed. The use of prototypes to test if requirements can be addressed is an example of a verification practice. The evaluation of a prototype by the users to test if the product fulfills their needs is an example of a validation practice. In other words, we can say that the verification should ask "Are you meeting the specified requirements?" and "Are you building the product right?" In the same way we can say that the validation should ask "Are you meeting the operational need?" "Does this product meet its intended use in the intended environment?" and "Are you building the right product?" [8]. ANSI/IEEE Std 1012 [9], Standard for

Software Verification and Validation Plans provides a set of minimum V&V tasks for each of the lifecycle phases in a software project.

2 Research Approach

To undertake this study, our research was performed through the following phases:

- The V&V processes were reviewed in detail to understand how they were addressed in the context of generic software development. The review included relevant literature, standards and process improvement models such as Capability Maturity Model® Integrated (CMMI®) [10], and ISO/IEC 15504-5 [11].
- Research was undertaken in the domain of safety-critical software development and V&V in this context. This included conducting a review of how the V&V processes were addressed by safety-critical software development standards such as DO-178B [12] and Automotive SPICE [13]
- Medical device software V&V was extensively researched. To this end a detailed analysis of the medical device standards was undertaken which included ISO/IEC 13485 [14], ISO/IEC 62304 [15], ISO/IEC 14971 [16] and the Food and Drug Administration (FDA) Guidelines for Software Validation (FDA GPSV).
- From the comparative analysis performed, we arrived at a set of findings which suggest areas for improving V&V practices in the medical device software domain

3 V&V in Generic Software Development

Two important reference models which are widely used in the context of software process improvement are the Capability Maturity Model® Integrated (CMMI®) and ISO/IEC 15504-5. They address the software verification and validation processes in the following ways: CMMI® in line with ANSI/IEEE Std 1012-1986 recommends a lifecycle view for verification and validation activities. It defines verification as "Confirmation that work products properly reflect the requirements specified for them". In other words, verification ensures that 'you built it right' and validation as "Confirmation that the product, as provided (or as it will be provided), will fulfill its intended use". Therefore, validation ensures that 'you built the right product'. The verification and validation processes are part of the engineering processes category and both are level 3 process areas in the staged model.

The verification process area in CMMI® is used by many of the other process areas irrespective of the category they fall into. These processes include Project Planning, Measurement & Analysis and other support process areas. The verification process is used to verify the work products created from the performance of these processes. Verification is also used extensively in the context of the engineering process areas which includes Requirements Management, Requirements Development, Technical Solution and Product Integration. The model also provides guidance in terms of examples of methods like peer reviews, statement coverage testing, and branch coverage testing that could be used in this context.

The validation process area incrementally validates products against the customer's needs. Validation may be performed in the operational environment or simulated operational environment. Coordination with the customer on the validation requirements is an important element of this process area. The scope of the validation process area includes validation of products, product components, selected intermediate work products, and processes. These validated elements may often require re-verification and revalidation.

Validation activities can be applied to all aspects of the product in any of its intended environments, such as operation, training, manufacturing, maintenance, and support services. Like the verification process area, validation is also performed during the course of the product development as it moves from each phase of the lifecycle. For example in the requirements phase, the model suggests analysis, simulations, prototyping and demonstrations as possible techniques for validation. Both validation and verification activities often run concurrently and may use portions of the same environment.

Similar to CMMI®, verification and validation have two distinct processes in ISO/IEC 15504-5. Verification and validation are under the supporting life cycle process category and the source for both these processes are ISO/IEC 12207 AMD1 [17].

In ISO/IEC 15504-5, the purpose of the verification process is to confirm that each software work product and/or service of a process or project properly reflects the specified requirements. The tasks pertaining to verification include: development of a verification strategy, development of criteria for verification, performing the activity of verification, determination of actions based on verification results and making the results available to the stakeholders.

As per ISO/IEC 15504-5, the purpose of the validation process is to confirm that the requirements for a specific intended use of the software work product are fulfilled. Validation aims to confirm by examination and provision of objective evidence that software or system specifications conform to user needs and intended uses, and the particular requirements implemented by the software product can be consistently fulfilled. The tasks pertaining to validation include: the development of a validation strategy and criteria for validation, performing the validation activities and the identification of problems, providing the validation data and making the results available to customers and other stakeholders.

The state of adoption of formal process management for V&V varies across organizations depending on the size and nature of business/products being developed. A study on the state of practice of V&V in industry [18] reveals the following: the process is more emphasized in large organizations while smaller organizations have identified inconsistencies in the documentation standard, but did not consider non-adherence to processes as being too harmful.

Though CMMI®, pays attention to V&V, it is still rather modest in its focus on these areas compared to other elements of the development processes [19]. As stated, industry-wide experiences indicate that validation and verification activities typically consume about 30-50% of development budgets [20]. In May of 2007, a working group was formed by ISO/IEC to produce a new software testing standard (the ISO/IEC 29119). The development of the standard is expected to be completed by May - October 2012 [21].

Our analysis points to the fact that the general software development standards and process improvement models need to be adapted to meet the specific requirements of medical device software development, which is highly regulated and safety-critical in nature.

4 V&V in Safety-Critical Domains

In order to review the V&V processes for software development in the safety-critical medical device domain, a relevant approach was to determine how verification and validation is addressed in the safety-critical software development domains of aerospace, avionics, space, nuclear and automotive. As with medical devices, software in these areas is a critical element of complex, potentially dangerous problems and failures in these products can result in loss of life, significant environmental damage, and major financial loss [22].

In the development of safety-critical systems, the speed of technology and rising demand for improved productivity has created a host of new challenges. These include issues such as, known safety techniques are not applied, ability to demonstrate (certify) that safety requirements have been met is inadequate, and automated code generators, automated testing tools etc. which are used to improve productivity may have associated problems which may be poorly understood [23]. In addition, designed products are required to meet a very high-level of reliability, security, and performance in safety-critical areas as many of them could pose serious threat to users/consumers or the general public. Therefore, ensuring that such systems meet their predefined requirements and that they perform as expected is a challenging issue [24].

It has been found that there is a relationship between the increasing occurrence of system accidents and the increasing usage of software [25]. Within the safety-critical software arena, different standards/certifications are available for different industries. These include the DO-178B for Aerospace, and Automotive SPICE and ISO 26262 [26] in the Automotive industry. IEC 60880 [27] describes the European standards for certification of nuclear power generating software. IEC 61508 [28] describes a general-purpose hierarchy of safety-critical development methodologies that has been applied to a variety of domains ranging from medical instrumentation to electronic switching of passenger railways. Though these standards address verification and validation in sufficient detail, their role is not to address process improvement. In addition, there are some authors [29] who consider that V&V assessment in CMMI® is not adequate when dealing with safety-critical software, and they proposed a new framework for V&V assessment, focused on safety-criticality. This framework is defined using safety standards together with the CMMI® V&V process areas and the ISO 9001 standard [30].

Some of the key differences between safety-critical standards and generic-software development standards/models highlighted in our research were:

a. Risk Management

Risk Management involves the identification and management of risk. Risk assessment is a function of impact and the probability of occurrence. A risk based approach to safety allows the hazards associated with a system to be identified and prioritized. The risk assessment involves calculation of level of risk associated with a hazard.

b. Integrity levels
Several standards use the concept of integrity levels to define the application of effort and rigour appropriate to the criticality of a component. In DO-178B, there are software levels starting from A to E (A being highest and E being lowest). ISO 26262 assigns the value of the ASILs (Automotive Safety Integrity Levels) from D (most critical), to A (least critical), based on IEC 61508

c. Stress on Independence
Independence in verification and validation is important because the developer of the software product may have a biased opinion about their own product and therefore possibly miss performing certain verification/validation activities.

d. Qualification of tools [31]
Section 12.2 of DO-178B, specifies that the objective qualification of tools used in development (which includes verification and validation tools) is to ensure that the tool provides confidence at least equivalent to that of the processes eliminated, reduced, or automated.

We examined these factors for the purpose of identifying a set of improvements for the medical device software domain.

5 Improvement in V&V in the Medical Device Software Domain

The applicable standards in the problem domain of medical device software development are ISO 13485, ISO/IEC 62304, ISO/IEC 14971 and FDA GPSV. We analysed how the areas of verification and validation are addressed by these standards in detail as well as in the relevant literature. We then compared our findings to standards and process improvement models in general software development and other safety-critical domains, which included the aerospace and automotive industry. As a result of this analysis we identified three key areas which need to be addressed for improvement in the context of verification and validation for medical device software development. These are as follows:

5.1 Having Distinct Processes for Verification & Validation

ISO 13485 represents the requirements for a comprehensive management system for the design and manufacture of medical devices. As per ISO 13485, verification is performed to ensure that the design and development outputs have met the design and development input requirements. Design and development validation is carried out to ensure that the resulting product is capable of meeting the requirements for the specified application as defined in relation to its intended use. Validation should be completed prior to the delivery or implementation of the product. However, the level of detail available for verification and validation in ISO 13485 is not sufficient for an effective implementation of these processes in the context of medical device software.

To address this, manufacturers rely on ISO/IEC 62304, which provides a framework of life cycle processes with activities and tasks necessary for the development and maintenance of medical device software. In this context the standard defines the software development lifecycle as a conceptual structure spanning the life of the software from the definition of its requirements to its release and maintenance. In ISO/IEC 62304, verification is confirmed through the provision of objective evidence that the specified requirements have been fulfilled. ISO/IEC 62304 identifies the milestones at which the completeness of specified deliverables are verified thus stressing the importance of verification. ISO/IEC 62304 expects that verification will be covered as part of each lifecycle activity from requirements to software release. Though the standard does address software verification, it considers validation to be a system level process and is therefore outside its scope even when the system is entirely software.

The FDA's GPSV outlines the general verification and validation principles that the FDA considers applicable to the validation of medical device software or the validation of software used to design, develop, or manufacture medical devices. As per the FDA GPSV, software verification provides objective evidence that the design outputs of a particular phase of the software development lifecycle meets all of the specified requirements for that phase. Software verification looks for consistency, completeness, and correctness of the software and its supporting documentation, as it is being developed, and provides support for a subsequent conclusion that software is validated. Even though validation is defined in Section 3.1.2 of the FDA GPSV, it is not covered thereafter in the guidelines.

ISO 14971 is a standard for performing risk based activities in the medical device domain. ISO 14971 states that verification is "confirmation, through the provision of objective evidence, that specified requirements have been fulfilled". Two distinct types of verifications are required by the standard: (1) Verification to confirm that the risk control measure has been implemented in the final design. (2) Verification to ensure that the measure as implemented actually reduces the risk. In some instances, a validation study can be used for verifying the effectiveness of the risk control measure.

5.1.1 Improving V&V through Separation of These Two Processes

Since ISO 14971 does not provide guidance on the verification and validation activities that should be performed manufacturers rely on ISO 13485 and ISO/IEC 62304 for implementation of verification and validation. As a result it can be difficult for the software manufacturers to meet the requirements of the risk management standard effectively.

As the processes of V&V are critical to software development in the medical devices domain, there needs to be a mechanism by which manufacturers could assess where they stand with respect to these processes and identify possible improvement opportunities. Manufacturers would also require a benchmark in terms of verification and validation processes to evaluate a potential vendor's capability in these processes. Hence it becomes necessary that a set of distinct processes be defined for verification and validation. Further, manufacturers would greatly benefit from the definition of a set of practices for verification and validation which relate to each of the engineering processes.

5.2 Independence in V&V

The IEEE Standards for Software Verification and Validation state that classical Independent Verification and Validation (IV&V) is generally required for the development of software systems deemed “critical” in nature, i.e., those which can result in loss of life, mission or significant social or financial loss [32]. The results of a study [7] highlight the difference in fault detection capabilities between two methods – non-independent V&V and Independent V&V (IV&V). The results indicate that IV&V provided a significant value-added component to the software development process. Independence is an important factor addressed by DO-178B and gives specific guidance on the subject.

ISO/IEC15504-5 and Automotive SPICE state that “degrees of independence” is something each project has to plan as part of its verification and validation strategy. The FDA GPSV addresses independence in Section 4.9, but leaves it to the discretion of device manufacturers on how this is to be achieved. ISO/IEC 62304 does not mandate independence. Independence is addressed in ISO 13485 in section 5.5.1 - Responsibility and Authority, where it states: “Top management shall establish the interrelation of all personnel who manage, perform and verify work affecting quality, and shall ensure the independence and authority necessary to perform these tasks”.

5.2.1 Improving V&V by Making Them Independent of Development

Section F.3 of ISO 14971 states that the risk management plan should identify the personnel with responsibility for the execution of specific risk management activities, for example, reviewer(s), expert(s), independent verification specialist(s), individual(s) with approval authority (see 3.2). This assignment can be included in a resource allocation matrix defined for the design project. From the statement, we can assume that the standard recognizes the need for independent verification specialist(s). Since ISO 14971 relies on other standards like the ISO 13485 for implementation of verification and validation, the manufacturers may have to look for additional guidance for implementation. As we have seen in the case of ISO 13485 and the FDA GPSV, though independence in verification and validation is addressed implicitly, there needs to be further guidance on which phases/engineering activities would require independence in verification and validation.

5.3 Qualification of Tools

The tools to be used in software development should be evaluated to ensure they adequately address the requirements for which they are being procured. In a safety-critical domain like Aerospace, the software verification process objectives for software development tools are described in the DO-178B standard. Tools are classified into software development tools and software verification tools. The qualification criteria for software verification tools are specifically addressed in the standard. ISO 13485 and ISO/IEC 62304 do not address this requirement, which is essential for safety-critical software development.

In terms of qualification of tools in software development and verification and validation activities, the FDA GPSV states that most of the automated equipment and systems used by device manufacturers are supplied by third party vendors and are purchased Off-The-Shelf (OTS). It gives guidance on validation of OTS software and automated equipment in section 6.3. The standard states that the vendor's life cycle documentation, such as testing protocols and results, source code, design specification, and requirements specification can be useful in establishing that the software has been validated. The standard however states that documentation may not be available from the product vendors as they might not be willing to share proprietary information. The standard suggests a number of practices which could be adopted by the device manufacturer given the lack of available documentation from the vendors.

5.3.1 Improving V&V by Using Appropriate Tools

Given what the standards propose, qualification of tools in the software development and V&V context needs to be addressed consistently in the medical device domain.

Due to delivery schedule pressures and shortfall in properly trained software engineers, the development of medical device software and systems has not kept pace with software assurance techniques practiced in other safety-critical domains such as avionics [33]. Apart from this, the lack of standard architectures for medical device software and the lack of integrated standard principled engineering tool support for analyzing software have created challenges to be dealt with [34].

Thus, there is a need for the development of an integrated model, which incorporates best practice and regulatory compliance, which is domain specific to medical device software development. To address this Medi SPICE [35], comprising of a process reference model and a process assessment model for software development in the medical devices domain is being developed by the authors. Medi SPICE brings together the best practice from 15504-5 and the regulatory requirements of the medical device standards which include ISO 13485, ISO/IEC 62304, and ISO 14971and FDA GPSV. Medi SPICE will also incorporate and address the findings from this research. The authors believe that with the publication of Medi SPICE more specific guidance will be available for process design, assessment, and improvement in the medical device software industry [35].

6 Conclusions

The identified areas for improvement will provide guidance for the definition of the V&V processes in Medi SPICE. Further to the definition of a set of processes and the associated practices related to V&V, the processes should be piloted in organizations within the medical device software development industry. Based on the results observed and the feedback from the medical device software development industry, the processes should be evaluated and continuously improved.

Acknowledgments

This research is supported by the Science Foundation Ireland (SFI) Stokes Lectureship Programme, grant number 07/SK/I1299, the SFI Principal Investigator Programme, grant number 08/IN.1/I2030 (the funding of this project was awarded by Science Foundation Ireland under a co-funding initiative by the Irish Government and European Regional Development Fund), and supported in part by Lero - the Irish Software Engineering Research Centre (http://www.lero.ie) grant 03/CE2/I303_1.

References

1. Bernard, E., Legeard, B., Luck, X., Peureux, F.: Generation of test sequences from formal specifications: GSM 11-11 standard case study. Software: Practice and Experience 34(10), 915–948 (2004)
2. Rayadurgam, S., Heimdahl, M.P.E.: Test-sequence generation from formal requirement models. In: Proc. of Sixth IEEE International Symposium on High Assurance Systems Engineering, 2001, pp. 23–31 (2001)
3. Myers, G., Sandler, C., Badgett, T., Thomas, T.: The Art of Software Testing, 2nd edn. Wiley, Chichester (2004)
4. Wallace, D.R., Fujii, R.U.: Software verification and validation: an overview. IEEE Software 6(3), 10–17 (1989)
5. US FDA Centre for Devices and Radiological Health, General Principles of Software Validation; Final Guidance for Industry and FDA Staff. CDRH, Rockville (2002)
6. Berling, T., Thelin, T.: An industrial case study of the verification and validation activities. In: Proceedings of Ninth International Symposium on Software Metrics, 2003, September 3-5, pp. 226–238 (2003)
7. Arthur, J.D., Groner, M.K., Hayhurst, K.J., Holloway, C.M.: Evaluating the effectiveness of independent verification and validation. Computer 32(10), 79–83 (1999)
8. Monteiro, P., Machado, R.J., Kazman, R.: Inception of Software Validation and Verification Practices within CMMI Level 2. In: 2009 Fourth International Conference on Software Engineering Advances, icsea, pp. 536–541 (2009)
9. IEEE Standard for Software Verification and Validation Plans, IEEE Std 1012-1986, p. I (1986)
10. CMMI Product Team, Capability Maturity Model® Integration for Development Version 1.2. Software Engineering Institute (2006)
11. ISO/IEC 15504-5:2006. Information technology — Process Assessment — Part 5: An Exemplar Process Assessment Model. ISO, Geneva, Switzerland (2006)
12. RTCA, RTCA DO-178B, Software Considerations in Airborne Systems and Equipment Certification (1992)
13. Automotive SPICE Process Assessment Model 2.2, Automotive SIG (August 21, 2005)
14. ISO 13485:2003. Medical devices — Quality management systems — Requirements for regulatory purposes. Second edn. ISO, Geneva, Switzerland (2003)
15. ANSI/AAMI/IEC 62304:2006. Medical device software—Software life cycle processes. AAMI, Arlington (2006)
16. ISO 14971:2007. Medical Devices — Application of risk management to medical devices. Second edn. ISO, Geneva (2007)
17. ISO/IEC 12207:1995/Amd.1. Information Technology — Software life Cycle Processes Amendment 1. ISO, Geneva, Switzerland (2002)
18. Andersson, C., Runeson, P.: Verification and validation in industry - a qualitative survey on the state of practice. In: Proceedings of 2002 International Symposium on Empirical Software Engineering, 2002, pp. 37–47 (2002)
19. Jacobs, J.C., Trienekens, J.J.M.: Improving verification and validation in hardware/software environments. Introduction to the workshop. In: Proceedings of 10th International Workshop on Software Technology and Engineering Practice, STEP 2002., pp. 121–122, October 6-8 (2002)
20. Kit, E.: Software Testing in the Real World. Addison-Wesley, Reading (1995)
21. ISO/IEC 29119 Software Testing, `http://www.softwaretestingstandard.org/projecttimeline.php` (accessed on April 18, 2011)

22. Why Software Requirements Traceability Remains a Challenge, Cross Talk (July/August 2009) Issue, http://www.crosstalkonline.org/storage/issue-archives/2009/200907/200907-Kannenberg.pdf (accessed on April 18, 2011)
23. Heimdahl, M.P.E.: Safety and Software Intensive Systems: Challenges Old and New. In: Future of Software Engineering, FOSE 2007, May 23-25, page 1, pp. 137–152 (2007)
24. Alawneh, L., Debbabi, M., Hassaine, F., Jarraya, Y., Soeanu, A.: A unified approach for verification and validation of systems and software engineering models. In: 13th Annual IEEE International Symposium and Workshop on Engineering of Computer Based Systems, ECBS 2006, March 27-30, vol. 10, p. 418 (2006)
25. Leveson, N.G.: Paper on System Safety in Computer-Controlled Automotive Systems. MIT, Cambridge (2011), http://www.sunnyday.mit.edu/papers/sae.pdf (accessed on April 18, 2011)
26. ISO/DIS 26262 Road vehicles - Functional safety. ISO, Geneva, Switzerland
27. IEC 60880:2006. Nuclear power plants - Instrumentation and control systems important to safety - Software aspects for computer-based systems performing category A functions. IEC, Geneva, Switzerland (2006)
28. IEC/TR 61508:2005. Functional safety of electrical/electronic/ programmable electronic safety related systems. BSI, London (2005)
29. Kyung, A.Y., Seung-Hun, P., Doo-Hwan, B., Hoon-Seon, C., Jae-Cheon, J.: A Framework for the V&V Capability Assessment Focused on the Safety-Criticality. In: Proc. 13th IEEE International Workshop on Software Technology and Engineering Practice, 2005, pp. 17–24 (2005)
30. ISO 9001:2000 - Quality management systems - Requirements (2000)
31. Eastaughffe, K.A., Cant, A., Ozols, M.A.: A Framework for Assessing Standards for Safety-critical Computer-Based Systems, isess. In: Fourth IEEE International Symposium and Forum on Software Engineering Standards, p. 33 (1999)
32. IEEE, IEEE Standards for Software Verification and Validation, Page 58 (1998)
33. Lee, I., Pappas, G.J., Cleaveland, R., Hatcliff, J., Krogh, B.H., Lee, P., Rubin, H., Sha, L.: High-Confidence Medical Device Software and Systems. Computer 39(4), 33–38 (2006)
34. Jetley, R., Iyer, S.P., Jones, P.L., Spees, W.: A Formal Approach to Pre-Market Review for Medical Device Software. In: 30th Annual International on Computer Software and Applications Conference, COMPSAC 2006, September 17-21, vol. 1, pp. 169–177 (2006)
35. Caffery, F.M., Dorling, A., Casey, V.: Medi SPICE: "An Update". In: The 10th International Spice Conference (SPICE 2010), Pisa, Italy, May 18-20 (2010)

The Meaning of Success for Software SMEs: An Holistic Scorecard Based Approach

Paul Clarke[1] and Rory V. O'Connor [2,3]

[1] Lero Graduate School in Software Engineering, Dublin City University, Ireland
pclarke@computing.dcu.ie
[2] Dublin City University, Ireland
[3] Lero, the Irish Software Engineering Research Centre
roconnor@computing.dcu.ie

Abstract. Software processes support the work of software development and software process improvement (SPI) is concerned with improving the operation of the software process. One of the primary reasons for conducting SPI is to increase the success of a software development company [1], [2]. While evidence of the benefits of SPI exists, project/senior managers report that their motivation for conducting SPI would be strengthened by the provision of further evidence of the positive impact of SPI on business success [3]. This paper proposes a new approach that utilises the Holistic Scorecard (HSC) [4] to systematically examine business success in software development companies. Furthermore, we relate the experience of applying this new approach to software small to medium sized enterprises (SMEs). This novel approach to examining success in software development companies provides a suitable mechanism for SPI researchers and practitioners seeking to establish evidence of the business benefits of SPI.

Keywords: Software Process Improvement, Business Success, Software SMEs.

1 Introduction

Owing to the diverse and dynamic nature of software development settings, software development teams and managers conduct SPI so as to "*create more effective and efficient performance of software development and maintenance through structuring and optimising of processes*" [5]. While there can be many motivations for conducting SPI [6], one of the important considerations is the maximisation of business success [1], [2]. However, there are different views in relation to business success [7]. Consequently, the authors have investigated the different views of business success and identified a reference framework, the HSC [4], which is appropriate for the examination of business success in software development companies. We believe that it is important for SPI studies to have a reliable, systematic and comprehensive method for making determinations in relation to business success and consequently, we have transformed the HSC framework into a survey instrument suited to the task of identifying the business objectives of software development companies. The survey instrument is deployed over time: initially, the instrument is utilised to determine the

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 72–83, 2011.

business objectives for the forthcoming period; subsequently, the instrument is deployed in order to determine the extent to which the original objectives have been achieved. This twin approach to determining business success improves the quality of the business success data by reducing the uncertainty associated with biased and false recollection. The initial component of the business success survey instrument has been deployed to the software SMEs sector, where lessons have been learned regarding the suitability of the HSC framework for use in smaller software development settings. Along with outlining the approach to identifying the business success parameters for software companies, the results of the initial application of the approach to software SMEs are presented.

The remainder of this paper is structured as follows: Section two presents details regarding different views of business success. Section three outlines the approach for establishing the extent to which a software development organisation is being successful, while section four relates the experience of applying this approach to software SMEs. Section five discusses the relevance of the HSC reference framework for software SMEs, and finally, section six presents a discussion and conclusion.

2 Different Views of Business Success

Many studies have demonstrated the benefits of SPI, both in large [8] and in small [9-11] organisations. However, it has been suggested that one of the de-motivators for SPI among project managers and senior managers relates to a shortfall of direct evidence of the business benefits of SPI [3]. For senior managers, evidence of the positive relationship between SPI and business success would help to assuage this concern. In order to conduct a business success investigation we must first identify a suitable reference framework of the dimensions of success for software SMEs. In the business literature, the term *success* is used interchangeably with the term *performance* and in a general sense they both represent the achievement of something desired, planned or attempted [12]. However, beyond this general description, controversy exists in relation to what exactly is meant and understood by the term business performance [13]. Businesses measure performance for a variety of different reasons including, the identification of improvement opportunities, determinations in relation to customer satisfaction, to enhance understanding of their own processes and to assess the degree of success achieved [14]. This variety of reasons for measuring performance has given rise to a variety of different performance measures that can be classified into one of two groups: financial and non-financial [7].

2.1 Financial Measures of Performance

Traditionally, business performance has been measured in purely financial or accounting terms [15]. Profitability, usually measured by return on investment (ROI), has by convention, been used to assess performance and is widely regarded as the ultimate bottom line test of success [13]. In addition to ROI, other financial measures of business performance include return on sales, sales per employee, productivity and profit per unit production [16]. The financial perspective has been reported as having a significant impact on performance – with Reid and Smith [17] concluding that the pursuit of the highest rate of return on investment is a primary consideration for owners and managers. This view is long established in the business success domain with Ansoff asserting in 1965 that "return on investment is a commonly and widely accepted yardstick for measuring business success" [18].

While financial return is an important indicator of business success, "profits are not necessarily the sole purpose of a firm" [19] and it has been observed that it is far from the only important measure [12], with claims that short term financial measures of performance that emphasise a quick return on investment can come at a cost to long term growth [20]. Financial measurement can be considered as tangible evidence of performance but other important performance measures should also be assessed so as to prevent the "inadequate handling of intangibles" and the "improper valuation of sources of competitive advantage" [21]. The measurement of customer satisfaction demonstrates the importance of intangible measures and highlights the danger of focusing solely on financial data: a company that posts successful financial returns might appear to be performing well but, if all of the clients are dissatisfied, the future profitability prospects for the company will be at risk. As a result of the shortcomings of purely financial performance measurement, there has been a "shift from treating financial measures as the foundation for performance measurement to treating them as one among a broader set of measures" [22] and this has given rise to multidimensional performance measurement frameworks.

2.2 Multidimensional Performance Measurement Frameworks

Owing to the dissatisfaction with traditional accounting-based performance measurement systems, multi-dimensional performance measurement frameworks were created as an alternative approach to business performance measurement [23]. As well as accommodating established financial measures of success, these new frameworks incorporated non-financial, future looking performance measures.

A number of multidimensional performance management frameworks have been created, each trying to unlock the vital measurements that would best provide a complete view of the business performance. The performance pyramid [24] contains a pyramid of measures aimed at integrating performance through the hierarchy of the organisation. The macro process model [25] identifies links between the five stages in a business process (*inputs*, *processing system*, *outputs*, *outcomes* and *goals*), arguing that each stage is the driver of the performance of the next. Kanji's Business Scorecard (KBS) defines four fundamental dimensions to be managed and measured: *organisational value*, *process excellence*, *organisational learning* and *stakeholder delight* while the performance prism [26] consists of five interrelated perspectives: *stakeholder contribution*, *stakeholder satisfaction*, *strategies*, *processes* and *capabilities*. However, it is the Balanced Scorecard (BSC) [27] approach that is the most popular multidimensional performance measurement framework [28] and which has exercised the most influence in the domain of performance management [29]. The BSC identifies four measurement perspectives: *financial*, *customer*, *internal business processes*, and *learning and growth*. While the BSC presents a packaged performance measurement approach that is considered to offer "good coverage of the dimensions of performance" [30], the novelty of the approach has been questioned, with claims that similar multidimensional approaches have existed since at least the 1960s [31]. Furthermore, some research has criticised the BSC as being difficult to implement and potentially not suited to small companies [32-34] – though it has also been noted that SMEs can derive benefits from the BSC approach without having to implement an administratively demanding measurement regime and that SMEs obtain the most value from the BSC when it is used to as a frame of reference for addressing general business goals [35]. Despite these criticisms of the BSC, it is the most widely adopted [36-38] and most notable [39] performance measurement framework.

While the BSC approach could be applied to any business type, the software development business, often characterised by high levels of dynamism and uncertainty, requires a broader approach to performance measurement [4]. Consequently, Sureshchandar and Leisten [4] have adapted the BSC approach, rendering a strategic performance measurement and management framework for the software development industry, the HSC. The HSC comprises of six perspectives: *financial, customer, business process, intellectual capital, employee* and *social* (refer to figure 1). While the initial three perspectives are similar to the BSC, the latter three – *intellectual capital*, *employee* and *social* – are new considerations and they reflect some of the key items that may affect the performance of a software business.

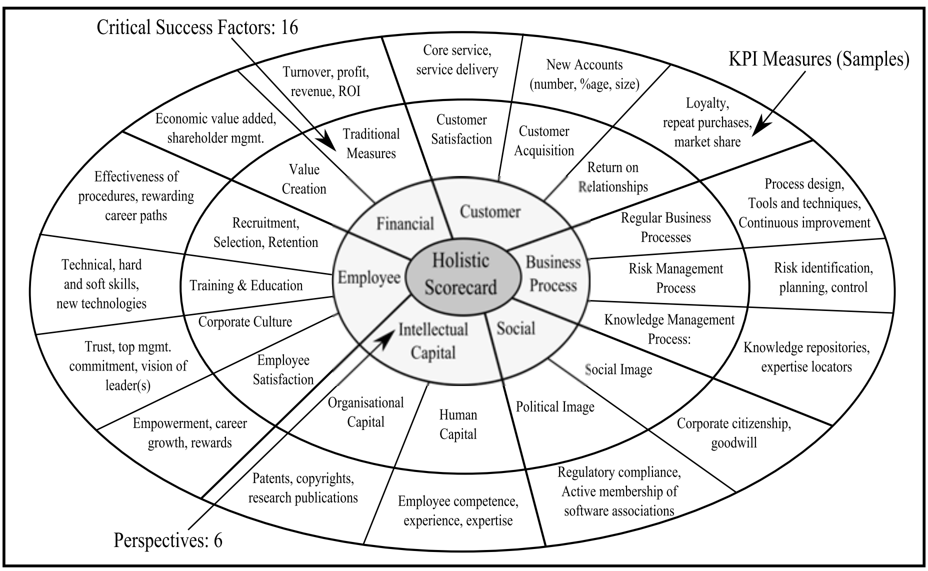


Fig. 1. Holistic Scorecard Overview

The HSC is a software development focused extension of the dominant business performance measurement framework, the BSC, and it outlines a framework for examining performance in software development companies. As indicated by Andersen, Cobbold and Lawrie [35], such balanced scorecard-based approaches are beneficial for SMEs when implemented in a fashion that supports the definition and measurement of strategic business goals. We have harnessed the HSC to support the construction of a business success survey instrument that can be used to determine the extent to which a software development company is achieving its objectives.

3 Harnessing the HSC to Examine Business Success

While the HSC identifies a broad spectrum of the performance parameters that are important for software development companies, it does not provide a survey-based instrument for identifying and measuring these parameters. Neither does the HSC offer guidelines on how to reliably collect the business objectives data. Therefore, using the HSC (refer to figure 1) as a reference, we constructed a business success survey instrument. Each of the six perspectives and the sixteen Critical Success Factors (CSFs) are identifiable in the resulting survey instrument – such that it is clear that the survey instrument has been derived from the HSC. The survey instrument is deployed in two phases (refer to figure 2): firstly, it is utilised to determine the business objectives for the forthcoming period; secondly, it is deployed in order to determine the extent to which the objectives are achieved. This two-phased approach to determining the extent of business success helps to ensure that the reported success in achieving business objectives is free from biased or false recollections – it also helps to formally identify the objectives in settings where no such formal description exists. Additionally, a series of questions are added to the survey instrument to support the disclosure of objectives that are beyond the scope of the HSC framework, as encouraged by the HSC creators [4]. The survey instrument was carefully constructed using the HSC as a reference and was subsequently subject to a pilot implementation with an SME industry partner. This piloting stage in the instrument creation ensures that the instrument is complete and fit for purpose. Following the pilot phase, a final rendering of the survey instrument for the examination of business in software development companies was produced.

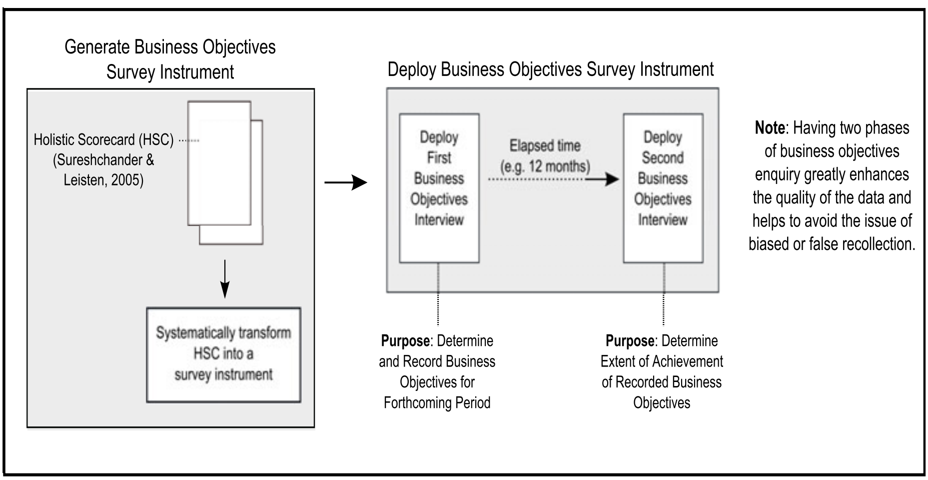


Fig. 2. Two phased approach to determining achievement of business objectives

3.1 Using the HSC Survey Instrument in Software SMEs

The first business objectives interview is designed to elicit business objectives for the forthcoming year, and over a six month period, we used the instrument to investigate the business objectives of seventeen additional SMEs. Each interview took around ninety minutes and the post-interview transcription required approximately six hours per interview. The participating SMEs are all primarily involved in the development of software and are from a broad range of sectors, including mobile telecommunications, insurance, web development, data mediation, embedded systems and email systems. While brief notes were taken during the interviews, the primary interview noting was conducted after the interview using a voice recording. Each voice recording was diligently examined in order to accurately record the response of the interviewee, taking care to note key phrases and remarks. Following the completion of the interview transcriptions, the data was analysed so as to identify the importance of the individual objectives. This analysis resulted in a post-interview spreadsheet for each participating SME, which assigned a weighting to each objective using the following Likert scale: 0 (no objective exists in this area), 1 (an objective exists, but with no explicit target), 2 (an objective exists, with an explicit target), 3 (a major objective exists, but with no explicit target), and 4 (a major objective exists, with an explicit target). Having completed the initial data transcription and objective grading exercise, a second pass was taken across all interviews to ensure the consistency of grading. Thereafter, we calculated the average importance of the various objectives across all participating organisation by summing the individual recorded priorities for the objectives and dividing the total by the number of participating organisations.

In addition to examining the average importance of the various business objectives, it is helpful to also examine the standard deviation within this grouping (the standard deviation being “the average of the distances of all the scores around the mean” [40]). By examining the standard deviation, it is possible to “gauge how consistently close together the scores are, and correspondingly, how accurately they are summarized by the mean” [41]. Lower standard deviation figures demonstrate greater uniformity in a data set and examination of the standard deviation for the business objectives data set reveals that in general, the deviation from the mean is generally relatively low – 41 of the 49 objective areas have a standard deviation of less than 1.

4 Business Objectives for Software SMEs

Following the data analysis and business objective prioritisation, we conducted an evaluation of the data. The data evaluation, which took approximately two months to complete, supported the development of an understanding of the data and facilitated the identification of key findings. The evaluation utilised the spreadsheets, averages and standard deviations output from the earlier data analysis, and revealed that the participating SMEs consistently have high priority objectives in six key areas: *revenue*, *profit*, *extension of product offerings*, *new client acquisitions*, *repeat business from existing clients*, and *business process management*. Growth in revenue is the single most important objective for SMEs, followed closely by profit considerations. Objectives in relation to profitability appeared to be somewhat eclipsed by a more basic need for survival – highlighting the difficult operating realities faced by some software SMEs. After revenue and profit targets, the next highest priority objectives are reported to be the extension of product offerings and the acquisition of new clients. Many of the participating SMEs could not identify the exact product extensions, stating only that they had strong intentions in this area and that product extension initiatives would be client-led. In relation to new client acquisition objectives, the majority of the participating SMEs had clearly identifiable targets. Gaining repeat business from existing clients and business process management are the final two areas that are generally reported as having high priority objectives. The majority of participating SMEs report strong targets in relation to gaining repeat business from existing clients, while business process management objectives tend to be more diverse in nature – some SMEs intend to improve the sales process while other SMEs have an objective to change the deployment licensing model for their software products.

The evaluation of the business objectives data also reveals that there are a number of areas where software SMEs have low priority objectives. Most notable among these objectives are: *contributing to society*, and *redressing grievances* which essentially don't feature for the any of the participating organisations. The absence of objectives in these areas appears to be related to survivability concerns which exert a significant pressure on the business as a whole. There was also a strong message from the participating SMEs that they do not intend to invest in training programmes and that they essentially have no objectives with respect to seeking or retaining a recognised quality standard. SMEs can therefore be characterised as organisations where best practice models are only implemented where their absence is considered to be a barrier to sales development, thus confirming the findings of earlier studies [42]. Furthermore, in software SMEs, training is "on the job" and there is very little interest in pursuing research publications.

In addition to identifying the high and low priority objectives for software SMEs, we also made a number of additional interesting observations. Very few of the participating companies manage risks in an organised or systematic way and they have no plans to start formalising risk management. Risk management is one dimension of self-reflection and is a conduit for continuous improvement – therefore, SMEs might derive some of the benefits of continuous improvement by establishing a risk management discipline. We were also interested to discover that several of the participating SMEs held the view that maintaining existing levels of customer satisfaction was going to be difficult if the business was to expand – since the small number of existing clients were presently receiving very high levels of dedicated support. The participating SMEs also report that other than "on the job" skills development, there is very little focus on career development for staff and that career growth was not considered to represent a high priority objective for the business. Furthermore, there appears to be "no place to hide" for underperforming employees (who are perhaps weeded out). These findings are somewhat at odds with the theoretical high importance of knowledge workers in software development – where continued career development may lead to increased motivation and higher retention rates among staff members. A further interesting observation was made in relation to the patenting ambitions of the participating SMEs, where only a few of the organisations have expressed patenting targets. The general belief among the participating SMEs is that patents are very expensive to file and that they offer little protection for the technology. For those SMEs that are engaged in patenting, the principal reported benefit is the protection of the valuation of the company for investors or purchasers.

Using the business objectives data gathered in the initial deployment of the HSC-based business success survey instrument, we prioritised each of the HSC objectives with respect to their relevance for SMEs. This prioritisation involved taking the business objective averages calculated earlier and using these to develop a hierarchy of business objectives for software SMEs - as depicted in figure 3. An evaluation of this hierarchy allows us to examine the relevance of the HSC for software SMEs.

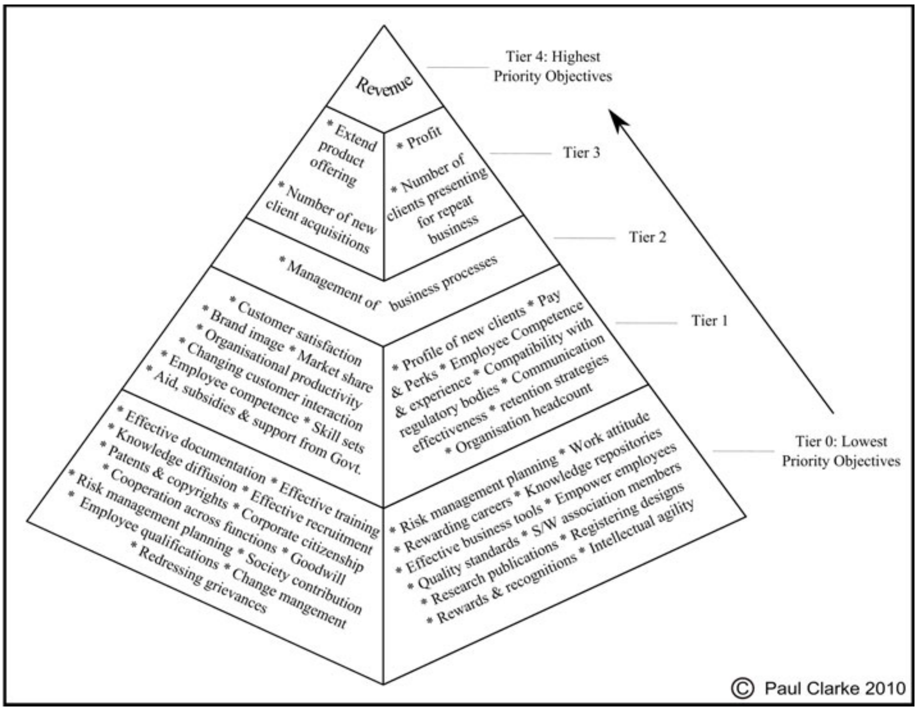


Fig. 3. Hierarchy of HSC Business Objectives for Software SMEs

5 Relevance of HSC to Software SMEs

For the purposes of this research, the HSC has provided a comprehensive framework from which to assess the objectives of software SMEs. Interviewees were expressly asked if there were any objectives that were not covered as part of the interview, and consistently they reported that the interview was comprehensive – with comments such as “quite comprehensive”, “it’s a fairly comprehensive framework” and “good questions”. However, there are some indications that the scope of the HSC may in fact be overly-broad for the purpose of examining software SMEs. Furthermore, a number of additional objectives were identified. Therefore, equipped with the data analysis and evaluation from this research, there are a number of recommendations that can be made with respect to the use of the HSC as a reference framework for future research in the area of business success for software SMEs.

The initial business objectives interview required on average a ninety minute interview with a senior manager from each of the participating SMEs, and later interview transcribing required a minimum of six hours per interview. This is a time consuming process for both the interviewee and the interviewer. Furthermore, the bulk of the HSC business objectives feature as relatively low priority items for the software SMEs in this study. Indeed, one of the interviewees commented that there was "a lot of emphasis on objectives which certainly in a small company doesn't ring true… [that] we've got revenue and product type objectives, other than that we tend to sort of blow with the wind a little and react, rather than being overly pro-active in the sense of setting any particular targets." Therefore, our first recommendation is as follows:

> **Recommendation 1.** If a future study of business objectives in SMEs were to use the HSC (or the HSC-based survey instrument produced by this research), the researchers could consider removing or consolidating the objectives that are in the lowest tier of the hierarchy in figure 3.

While recommendation 1 could be adopted by a future research effort in the software SMEs sector so as to make the identification of objectives more efficient, we consider that it is important to retain closing questions that permit the interviewee to comment on any additional objectives. It is difficult for any survey instrument to be absolutely complete and the inclusion of such closing questions permits the elicitation of objectives that are beyond the scope of the survey instrument or that have possibly been overlooked. In our own application of the HSC-based survey instrument to SMEs, such closing questions allowed us to discover a number of additional objectives that are not native to the HSC. Consequently, our second recommendation is that questions in relation to a number of additional objectives should be included in the survey instrument:

> **Recommendation 2.** Future research into the business objectives in software companies should include questions relating to objectives in the areas of (1) financial liquidity (sometimes termed cash flow); (2) off-shoring or outsourcing some aspects of the development work; (3) mergers and acquisitions (M&A).

6 Discussion and Conclusion

Software processes and SPI support software development efforts, and the success of these efforts affects the success of the overall business. Therefore, when making determinations in relation to the efficacy of software processes and SPI activities, we will sometimes need to examine business success. The case for SPI would benefit from additional studies that demonstrate the positive influence of SPI on business success, especially in SMEs. In order to support research efforts that examine the relationship between SPI and business success, we have identified a comprehensive reference framework, the HSC [4], for examining business success. In addition, we have outlined an approach to applying this framework to examining business success in software development organisations. This involves a two-phased engagement with companies (refer to figure 2), an approach which improves the reliability of the success data, especially in companies where business success criteria are not well defined in the first instance.

SPI is just one of many factors that can affect the success of software development organisations. Therefore, attempts to correlate SPI efforts with business success may require multi-organisational research. Nonetheless, we should have a sound and reliable approach for determining business objectives and for evaluating business success – so that we have the possibility to correlate software process initiatives with business success. In this paper, we have presented one such approach. Furthermore, we have applied this approach to seventeen software SMEs. The results of this application indicate that revenue, profit, client acquisitions and extension of product offerings are strong business success criteria for software SMEs. However, we also find that SMEs have weak or non-existent objectives in relation to contributing to society, redressing grievances, patenting, conducting research and seeking recognised quality standards. In addition, our research has indicated that the HSC reference framework would benefit from the addition of objectives related to financial liquidity, off-shoring/outsourcing software development, and mergers and acquisitions. In future work, we will revisit the participating SMEs and use the survey instrument to determine the extent of business success relative to the stated objectives.

We believe that the approach to examining business success that has been outlined in this paper is of use to future researchers in the software process and SPI domains. Furthermore, we believe that the approach outlined may be of benefit to software development practitioners, whose SPI initiatives could be more successful if guided by the key business objectives identifiable using our survey instrument. The findings of the initial application of the business success survey instrument to software SMEs has provided an interesting insight into the objectives of software SMEs, and has permitted the construction of a hierarchy of objectives for software SMEs. This initial application has also produced a number of important lessons which have been outlined for the benefit of future researchers in the software process and SPI domains.

Acknowledgments. This work is supported, in part, by Science Foundation Ireland grant 03/CE2/I303_1 to Lero, the Irish Software Engineering Research Centre (www.lero.ie).

References

1. Rico, D.: Using Cost Benefit Analyses to Develop Software Process Improvement (SPI) Strategies. Wright-Patterson Air Force Base: A DACS State-of-the-Art Report, Air Force Research Laboratory – Information Directorate, Dayton, Ohio, USA (2000)
2. Horvat, R., Rozman, I., Gyorkos, J.: Managing the Complexity of SPI in Small Companies. Software Process: Improvement and Practice 5(1), 45–54 (2000)
3. Baddoo, N., Hall, T.: De-Motivators for Software Process Improvement: An Analysis of Practitioners' Views. Journal of Systems and Software 66(1), 23–33 (2003)

4. Sureshchandar, G.S., Leisten, R.: Holistic Scorecard: Strategic Performance Measurement and Management in the Software Industry. Measuring Business Excellence 9(2), 12–29 (2005)
5. Van Solingen, R.: The Cost and Benefits of Software Process Improvement. In: Proceedings of the 8th European Conference on Information Technology Evaluation, pp. 455–465. MCIL, Reading (2001)
6. Baddoo, N., Hall, T.: Motivators of Software Process Improvement: An Analysis of Practitioners' Views. Journal of Systems and Software 62(2), 85–96 (2002)
7. Hart, S.: Dimensions of Success in New Product Development: An Exploratory Investigation. Journal of Marketing Management 9(1), 23–41 (1993)
8. Van Solingen, R.: Measuring the ROI of Software Process Improvement. IEEE Software 21(3), 32–38 (2004)
9. Biro, M., Ivanyos, J., Messnarz, R.: Pioneering Process Improvement Experiment in Hungary. Software Process: Improvement and Practice 5(4), 213–229 (2000)
10. Sanders, M. (ed.) The SPIRE handbook, better, faster, cheaper software development in small organisations. Centre for Software Engineering Limited, DCU, Dublin, Ireland (1998)
11. Von Wangenheim, C.G., Weber, S., Hauck, J.C.R., Trentin, G.: Experiences on Establishing Software Processes in Small Companies. Information and Software Technology 48(9), 890–900 (2006)
12. Maidique, M.A., Zirger, B.J.: The New Product Learning Cycle. Research Policy 14(6), 299–313 (1985)
13. Morgan, R.E., Strong, C.A.: Business Performance and Dimensions of Strategic Orientation. Journal of Business Research 56(3), 163–176 (2003)
14. Parker, C.: Performance Measurement. Work Study 49(2), 63–66 (2000)
15. Jennings, D.F., Seaman, S.L.: High and Low Levels of Organizational Adaptation: An Empirical Analysis of Strategy, Structure, and Performance. Strategic Management Journal 15(6), 459–475 (1994)
16. Ghalayini, A.M., Noble, J.S.: The Changing Basis of Performance Measurement. International Journal of Operations & Production Management 16(8), 63–80 (1996)
17. Reid, G.C., Smith, J.A.: What Makes a New Business Start-Up Successful? Small Business Economics 14(3), 165–182 (2000)
18. Ansoff, H.I.: Corporate strategy. McGraw-Hill, New York (1965)
19. Nonaka, I., Toyama, R.: The Theory of the Knowledge-Creating Firm: Subjectivity, Objectivity and Synthesis. Industrial and Corporate Change 14(3), 419–436 (2005)
20. Hayes, R.H., Abernathy, W.J.: Managing our Way to Economic Decline. Harvard Business Review 58(4), 67–77 (1980)
21. Bharadwaj, S.G., Varadarajan, P.R., Fahy, J.: Sustainable Competitive Advantage in Service Industries: A Conceptual Model and Research Propositions. The Journal of Marketing 57(4), 83–99 (1993)
22. Eccles, R.G.: The Performance Measurement Manifesto. Harvard Business Review 69(1), 131–137 (1991)
23. Bourne, M., Mills, J., Wilcox, M., Neely, A., Platts, K.: Designing, Implementing and Updating Performance Measurement Systems. International Journal of Operations & Production Management 20(7), 754–771 (2000)
24. Lynch, R.L., Cross, K.F.: Measure up! yardstick for continuous improvement. Basil Blackwell, Cambridge (1990)
25. Brown, M.G.: Keeping score: Using the right metrics to drive world-class performance. Quality Resources, New York (1996)

26. Neely, A.D., Adams, C., Kennerley, M.: The performance prism: The scorecard for measuring and managing business success. Prentice Hall, London (2002)
27. Kaplan, R.S., Norton, D.P.: The Balanced Scorecard - Measures that Drive Performance. Harvard Business Review 70(1), 71–79 (1992)
28. Kennerley, M., Neely, A.: Performance measurement frameworks: A review. In: Business Performance Measurement - Theory and Practice. Cambridge University Press, Cambridge (2002)
29. De Waal, A.A.: Behavioural Factors Important for the Successful Implementation and use of Performance Management Systems. Management Decision 41(8), 688–699 (2003)
30. Hudson, M., Smart, A., Bourne, M.: Theory and Practice in SME Performance Measurement Systems. International Journal of Operations & Production Management 21(8), 1096–1115 (2001)
31. Pandey, I.M.: Balanced Scorecard: Myth and Reality. Vikalpa 30(1), 51–66 (2005)
32. Gautreau, A., Kleiner, B.H.: Recent Trends in Performance Measurement Systems - the Balanced Scorecard Approach. Management Research News 24(3), 153–156 (2001)
33. McKenzie, F., Shilling, M.: Avoiding Performance Measurement Traps: Ensuring Effective Incentive Design and Implementation. Compensation and Benefits Review 30(4), 57–65 (1998)
34. Chow, C.W., Haddad, K.M., Williamson, J.E.: Applying the Balanced Scorecard to Small Companies. Management Accounting 79(2), 21–27 (1997)
35. Andersen, H., Cobbold, I., Lawrie, G.: Balanced Scorecard implementation in SMEs: reflection in literature and practice. In: Proceedings of the Fourth SMESME Conference, Department of Production, 2GC Limited, pp. 103–112. Aalborg University, Aalborg (2001)
36. ADB: Balanced scorecard for state-owned enterprises. Asian Development Bank, Technical Assistance Project Team 3933-PRC, Philippines (2007)
37. Lim, A.H.L., Lee, C.S.: Integrated Model Driven Business Evaluation Methodology for Strategic Planning. International Journal of Business Information Systems 3(4), 333–355 (2008)
38. Barnes, D., Hinton, M.: The benefits of e-business performance measurement systems. CIMA Publishing, Oxford (2008)
39. Tapanya, S.: Examining the Factors which Influence Performance Measurement and Management in the Thai Banking Industry: An Application of the Balanced Scorecard Framework (2004)
40. Malim, T., Birch, A. (eds.): Research methods and statistics. MacMillan Press Ltd., Basingstoke (1997)
41. Heiman, G.W. (ed.): Understanding research methods and statistics. Houghton Mifflin Company, Boston (2001)
42. Coleman, G., O'Connor, R.: Investigating Software Process in Practice: A Grounded Theory Perspective. Journal of Systems and Software 81(5), 772–784 (2008)

Five Agile Factors: Helping Self-management to Self-reflect

Christoph J. Stettina and Werner Heijstek

Leiden Institute of Advanced Computer Science, Leiden University,
Niels Bohrweg 1, 2333 CA Leiden, The Netherlands
{stettina,heijstek}@liacs.nl
http://www.liacs.nl/~{stettina,heijstek}

Abstract. In this paper a tool is proposed to foster reflection in agile software development teams. Based upon the qualitative model of Moe et al. [11], we contribute a quantitative questionnaire organized along five dimensions of agile teamwork analogous to the "Five Factor Model" in contemporary psychology. To test this survey tool and its alignment with existing studies, we have executed an empirical validation of the tool with 79 individuals and 8 international Scrum teams. We find that inter-team agreement on the factors is high and that the survey tool is found very useful. The instrument offers a comparable measure to agile teams and gives recommendations for each of the factors helping to understand individual as well as organizational level barriers.

Keywords: self-management, software development, agile teams, scrum, organizational management and coordination, process implementation and change.

1 Introduction

With the introduction of agile methods [4] such as Scrum and Extreme Programming (XP), the emphasis on people and their integration into the organizational process of software development has become increasingly important. Scrum, as an adaptive and empirical process, for example, aims to replace *command-and-control* management with *collaborative self-managing teams* [11].

Self-organizing project teams have been found successful by Takeuchi and Nonaka [19] while studying product development projects in large Japanese companies. Since then they have been confirmed to have high productivity and increased speed in problem solving [7,20]. While recognized as a premise for innovative projects, they are considered to be one of the biggest challenges for the adoption of agile methods. Benefits and limitations of agile development have been repeatedly reported to be dependent on human and social factors. The related changes in company culture and the awareness necessary were found to be difficult to adapt in practice [4,11,10]

While more embedded within the process surrounding software development than the pure function of writing code, agile teams are exposed to organizational

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 84–96, 2011.

barriers to a greater extent. In traditional and plan-driven *command-and-control* environments there exists a clear separation of roles, driven by self-managing professionals. In collaborative self-managing teams instead it is more important that team members understand individual as well as organizational level barriers [10]. We aim to improve understanding of these barriers in this contribution.

2 Objectives

Implementation of agile and self-organizing teams can be aided by increased development team self-awareness. In order to protect themselves from management, agile teams have been observed to give the impression that the team is better than they were [10]. This *impression management* [10] has been found to be a reason for failure to learn and change operating modes inside agile teams, preventing key issues of the process to be addressed.

In this paper we propose a tool for self evaluation to improve reflection of agile software development teams. To this end, we developed an instrument providing a comparable and practical measure for team members and have linked it to feedback for reflection based upon the study design and findings of Moe et al. [11]. We pose the following research question:

> *To what extent can we use the findings of Moe et al. [4,11,10,13] to measure self-management in order to support reflection in agile teams?*

Our objective is to promote discussion inside agile teams through the adoption of an impersonal survey tool in order to better understand mechanisms of effective teamwork and organizational requirements.

3 Related Work

In contemporary psychology, the *Five Factor Model* (FFM) also known as the *Big Five Personality Traits* is a model describing human personality through lexical analysis. The five factors were discovered and defined by factor-analyzing hundreds of measures of known personality traits [3].

Dybå and Dingsøyr conducted a structured literature review on empirical studies to address the scientific level of evidence behind agile software development methodologies identifying 36 out of 1996 studies matching their criteria [4]. To examine teamwork in agile software development teams the group developed five dimensions of agile teamwork [11] building up on work of Salas et al. [17]. They have placed their dimensions based upon a set of open-ended interview questions within an action research program with companies applying Scrum and evaluated their qualitative design conducting interviews with all team members in three longitudinal projects [11]. In the scope of the three years lasting program they found the absence of redundancy and the conflict between team level and individual level autonomy as one of the biggest barriers in implementing self-managing agile teams [10].

This instrument as originally developed by Moe et al. [11] consists of the five dimensions *shared leadership*, *team orientation*, *redundancy*, *learning* and *autonomy* as outlined in table 1. They have developed a set of open-ended interview questions for each of the dimensions to be conducted with all respective members of a Scrum team. Build on theoretical and empirical ground [4,12,17,10] their five dimensions of agile teamwork and their qualitative questionnaire forms the basis of our quantitative research design.

4 Method

With the goal to promote understanding and reflection on organizational and team level barriers [10] from a development unit's perspective we provide an instrument to be applicable from within the team. To simplify the collection process we have thus developed an anonymous questionnaire to promote more objective answers.

To reduce bias we encouraged the team members to provide their honest opinions by emphasizing the anonymous treatment of data. No results other than the processed outcome for the whole team would be distributed or given to their superiors. While we provided personalized links for each team member to ensure the consistency of input, no personal details were stored or used within the examination. Furthermore, some of the questions would only strengthen the agile factor when disagreed upon. This prevents a high ranking when answering all questions positively.

To furthermore increase transparency of the data we documented the level of agreement, the variance of answers given by the team members. This should help pointing at inconsistency within the team.

4.1 Questionnaire Design

To enable data collection via online surveys we adapted the qualitative questions of Moe et al. [11] into a quantitative design. The question sentences (table 1) have been held as close as possible to the original design. A screen shot of the online questionnaire page can be found in figure 1.

The questionnaire has been changed by adding "I feel" at the beginning of each sentence. This has been done to enable team members to better identify themselves with the research while keeping a comparable measure to original findings. Then, for each of the questions the participants were given a standard Likert scale to express their perceptions. To prevent inconsistency among the rating items we used a standard Likert scale consisting of 5 items: *Strongly Agree* = 5, *Agree* = 4, *Neutral* = 3, *Disagree* = 2, *Strongly Disagree* = 1.

4.2 Team Agreement

Variance (σ^2) is a measure of how far each value in a set of responses is from the mean. Variance is a useful measure for the level of agreement within a team, based on our survey, because variance is proportional to the scatter of the response metrics and independent of the number of responses.

Table 1. Five dimensions of agile teamwork and related personal questions for the agile team radar as inspired by Moe et al. [11]

Shared Leadership
Creation and maintenance of the team's shared mental model and transfer of leadership according to key knowledge, skills and abilities, shared decision authority

- I feel everyone is involved in the decision-making process
- I feel team members make important decisions without consulting other team members
- I feel the team vision is well defined and presented
- I feel the team is designed (and redesigned) according to its purpose

Team Orientation
Promotion of team cohesion counteracts social loafing and increases individual responsibility, team goals are given priority over individual goals

- I feel the team takes into account alternative suggestions in team discussions
- I feel the team values alternative suggestions
- I feel team members relate to the tasks of individuals
- I regularly comment on a co-worker's work

Redundancy
Cross-functionality avoids bottlenecks and enables possibility to shift workloads and mutual assistance

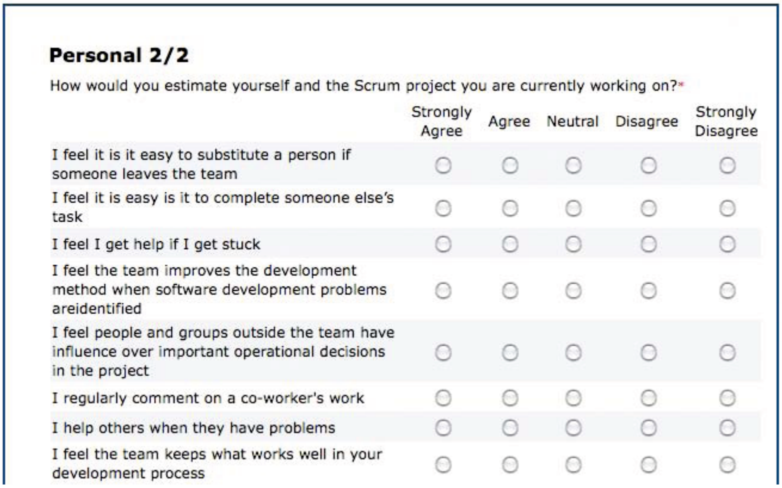
- I feel it is easy to complete someone else's task
- I feel I get help if I get stuck
- I help others when they have problems
- I feel it is easy to substitute a person if someone leaves the team

Learning
Interdisciplinary knowledge acquisition to promote self-optimization in a wider environment

- I feel the team keeps what works well in the development process
- I feel the team improves the development method when software development problems are identified
- I feel the team gives feedback on all aspects of each others work

Autonomy
External influences on the activities of the team, a precondition for self-management. Although sometimes beneficial, such influences can discourage group thinking.

- I feel the team looses resources to other projects
- I feel people and groups outside the team have influence over important operational decisions in the project
- I feel decisions made by the team are respected by people and groups outside the team



Personal 2/2

How would you estimate yourself and the Scrum project you are currently working on?*

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
I feel it is it easy to substitute a person if someone leaves the team	○	○	○	○	○
I feel it is easy is it to complete someone else's task	○	○	○	○	○
I feel I get help if I get stuck	○	○	○	○	○
I feel the team improves the development method when software development problems areidentified	○	○	○	○	○
I feel people and groups outside the team have influence over important operational decisions in the project	○	○	○	○	○
I regularly comment on a co-worker's work	○	○	○	○	○
I help others when they have problems	○	○	○	○	○
I feel the team keeps what works well in your development process	○	○	○	○	○

Fig. 1. Online Questionnaire

The variance is defined as

$$\sigma^2 = \frac{\sum (X-\mu)^2}{N} \tag{1}$$

where μ is the mean.

A lower variance therefore corresponds with a greater level of agreement within a team. The maximum variance in a team is the variance of the maximum and minimum values that can be given in response to an answer. The minimum variance is 0, denoting complete agreement. On a Likert scale from 1 – 5, the maximum variance is

$$\sigma^2 = \frac{\sum (X-\mu)^2}{N} = \frac{\sum X^2}{N} - \mu^2 \tag{2}$$

$$\text{MAX}(\sigma^2\{1,5\}) = \frac{1^2+5^2}{2} - \left(\frac{1+5}{2}\right)^2 = 4 \tag{3}$$

5 Results

To test our questionnaire and to inquire its matching to existing research, the questions have been presented to a group of international project teams, practitioners and experts applying the Scrum methodology. The instrument has been provided to the participants as a set of online survey questions in random order. The participants had to be actively involved in a Scrum development team. All questions had to be answered in order to count the respective data set as valid.

To look for international Scrum teams interested in the study, one of the authors searched Scrum/Agile oriented groups within business related networks.

After identifying related individuals from different online community platforms, user groups as well as originating from direct and indirect contacts, the author sent invitations for participation to 150 Scrum related professionals. Those who were interested in participation received an anonymized link allowing identification of teams within the online survey system. In addition each ScrumMaster of a potentially interested Scrum team received a set of open-ended questions regarding the project environment.

After data collection, the given answers were accumulated into global and team samples as shown in table 2 and figure 2. The total number of valid data sets collected contains 79 individuals and 8 teams from 13 countries. Most of the participants belong to the group of software developers (47%) and ScrumMasters (18%). Other groups, however, emerged within the data collection phase. Their data has been taken into evaluation as long as the individuals were committed to a *Pig* role within the Scrum project: Product Owner (8%), Quality Assurance (6%), Agile Coach (6%), Consultant (9%), Interaction Designer (1%), CTO (5%). The gross amount of relevant working experience among the participants is situated around a work record of 1-5 (38%) and 6-10 (29%) years.

After primary analysis, the author decided on 8 teams to be taken into team analysis. The teams had to consist of at least four members with, depending on the team size, at least two-thirds of the team having answered the survey in order to represent a consistent group image. The remaining survey answers were only analyzed globally.

5.1 Team Sample

Table 2 contains each team's self-assessment scores based on the Likert scale data from the questionnaire as mean values for each team.

The minima reveal a consistency towards the dimension of autonomy and there is a noticeable tendency towards learning among the maxima. Autonomy consistently earns the smallest score for all teams, while learning is the highest perceived characteristic for half of the eight teams and changes between redundancy and team orientation for the other half. This results show a similar trend in distribution as those presented in the original findings of Moe et al. [11].

Team agreement, expressed by variance (σ^2) is mentioned in table 2 below the aggregated team level measurements. We observe a pretty high (0.06-0.33) level of agreement within the teams as represented by a fairly low variance. Also the agreement on the five factors is pretty high (0.18-0.20). Teams agree least on redundancy and shared leadership and most on team orientation and autonomy.

The consistent low rating on low autonomy and high agreement that rating among the team members are a pointer to organizational level barriers and can be tracked back to our first two questions for the factor (table 1). The least agreement on redundancy can be a pointer to contended ideas regarding specialization in agile teams. Many participants reacted skeptical towards the implementation of cross-functionality. Although being aware of the "quagmire" effect of specialization [10], many could not think of how to overcome the idea as a waste of resources.

Table 2. Descriptive variables, radar results (x) (**min** & **max**) and agreement (σ^2)

		T1	T2	T3	T4	T5	T6	T7	T8	AVG. AGR.
country		UK	US	UK	NO	NL	SE	IN	NZ	
team size (pers.)		4	9	5	12	6	4	8	6	
collected answers		4	6	5	6	5	3	8	4	
avg. exp. (yrs.)		7.75	13.7	6.6	12.7	2.6	10	7	3.5	
shared leadership	x	**4.13**	**3.83**	**3.90**	**3.83**	**3.10**	**3.17**	**3.59**	**3.69**	
	σ^2	*(.05)*	*(.08)*	*(.29)*	*(.47)*	*(.06)*	*(.22)*	*(.08)*	*(.36)*	*(.20)*
team orientation	x	**4.56**	**4.21**	**4.15**	**3.88**	**3.30**	**3.83**	**3.69**	**3.88**	
	σ^2	*(.14)*	*(.15)*	*(.27)*	*(.34)*	*(.01)*	*(.06)*	*(.09)*	*(.39)*	*(.18)*
redundancy	x	**4.38**	**3.67**	**3.85**	**4.10**	**3.30**	**3.67**	**3.94**	**3.25**	
	σ^2	*(.08)*	*(.22)*	*(.14)*	*(.16)*	*(.32)*	*(.18)*	*(.28)*	*(.22)*	*(.20)*
learning	x	**4.58**	**4.22**	**4.20**	**3.50**	**3.33**	**3.56**	**3.58**	**3.75**	
	σ^2	*(.02)*	*(.25)*	*(.03)*	*(.32)*	*(.36)*	*(.25)*	*(.13)*	*(.19)*	*(.19)*
autonomy	x	**3.50**	**3.61**	**3.27**	**3.17**	**3.07**	**2.78**	**3.13**	**2.92**	
	σ^2	*(.03)*	*(.16)*	*(.24)*	*(.33)*	*(.32)*	*(.18)*	*(.11)*	*(.08)*	*(.18)*
average agreement	σ^2	*(.06)*	*(.17)*	*(.19)*	*(.33)*	*(.15)*	*(.18)*	*(.08)*	*(.25)*	*(.18)*

Members of T1 (UK) and T7 (India) agree most. T1 (UK) provides a back-end software for a major Massive Multiplayer Online (MMO) game publisher. T7 (India) worked on a e-commerce solution. Both were collocated development teams with similar roles and good team consistency. Members of T4 (Norway) and T8 (New Zealand) agree least. T4 (Norway) is employed by a company providing smart card based public key solutions for security transactions, consisting of developers from 2 separate locations running "several parallel projects". Team T8 (New Zealand) consists of a business analyst, a quality assurance specialist and two developers working for a state insurance agency. Both are rather diversified teams with different roles. T4 and T8 have a notably increased variance for shared leadership and team orientation while T1 and T7 agree on those. Although the level of agreement does not reflect on agile values, it indeed seems to correlate with the consistency of the teams.

5.2 Global Sample

To have a more detailed view on the data we have compiled a global team radar consisting of the answers of all 79 participants, as depicted in figure 2. In consistence with the findings of Moe et al. [10] we found autonomy to be significantly lower than the rest of the factors. We could not find a significant difference between the means of redundancy and learning, team orientation and shared leadership.

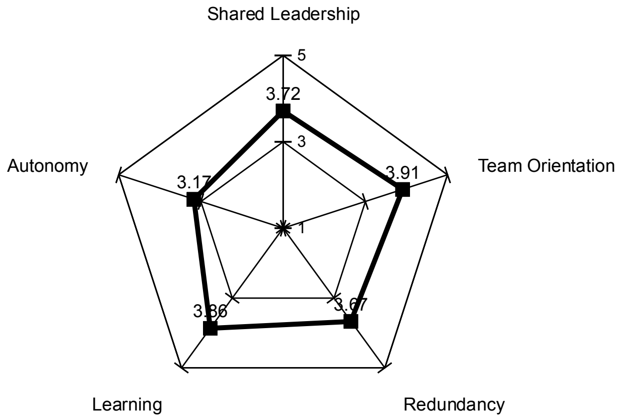


Fig. 2. Global Team Radar

5.3 Validity Considerations

Due to the low amount of data sets containing 79 individuals conclusions should be drawn carefully. Here we have stressed particular attention to the quality of collected data. Throughout the whole process of data collection we encouraged participants in giving realistic answers and emphasized the anonymous treatment of data to establish a reasonable level of trust. Only complete data sets and teams with a minimum amount of participants were taken into evaluation.

Quantitative data collection typically grounds evidence on big data sets. As we base our evidence on small team data sets instead, we have improved the transparency of data by adding the variance of given answers among the team members.

The distribution of given answers reveals an expected bias of participants towards positively perceived answers. In psychology this effect is being referred to as Socially Desirable Responding (RDS) [15]. This effect can be lowered by anonymous self-administration, meaning that when the subjects' personal details are not required the person does not feel directly and personally involved in the answers he or she is going to give. The second provision we applied to reduce social desirability, is the self-administration of the survey through a computer. The self-administration of the survey through a computer neutralizes here social desirability through the impersonality of the machine.

5.4 Discussion

The noticeable tendencies reveal a global minimum for measured autonomy and maxima for learning and team orientation. The data suggests that Scrum teams seem not to be well prepared, to cope with the cultural environment existing in their companies in order to maintain a level of autonomy required to apply the methodology. It seems that the internal factors are indeed supporting Scrum within a team, by the willingness to share leadership and good team orientation

- values which arguably might be perceived as being passive and existing within small development teams with divided roles anyway. Redundancy, scoring the second lowest value here, resulting in a lack of cross-functionality could in fact create a breeding ground for interpretations that some teams are not actively prepared for a faithful implementation of Scrum. This strengthens the need of further involvement of developers in discussions regarding implementation of agile processes.

The application of our questionnaire was met with interest. This is also reflected by the relatively high response rate: 79 respondents out of circa 150 inquired professionals. During data collection we received questions and suggestions from participants, especially from those with most experience in application of agile methods. However, it was not always easy to collect consistent data from a whole team, and thus out of 79 participants just 8 teams could be taken into team analysis. This might be partially caused by the invitation offered through superiors. In two cases there was direct interest of clients or *Product Owners* with an offshore development team. In this case the contractors were assumed to be interested in learning about the consistency of the hired development team, leading to poor commitment to the survey. Data collection should be motivated by the desire to learn and improve inside the team and should not be used by means of organizational control. Commitment thus is to be expected when executed on team initiative.

6 Recommendations

After data collection we have been repeatedly asked by the teams and *Scrum-Masters* for recommendations with respect to the findings as during the design of the study we did not think of recommendations. As the five factors alone provide a comparable measure but little practical advice to the audience we would like follow-up on this. In the following section we thus provide a list of advices on each of the five factors from current literature.

6.1 Shared Leadership

Literature argues that leadership should be transferred accordingly to the key knowledge, skills and abilities necessary for a particular issue at a moment in time [16,8]. The team leader's task as argued by Salas et al. [17] therefore should be the creation and maintenance of the team's shared metal model while the teams collaborative process. Moe et al. [10] give the example of a "chief architect" on one hand and of a newly hired developer on the other: while the chief architect took over most of the decisions in one company, leading to frustration of team members, a newly hired software developer had to fight for attention in another company. Team members should share decision authority to promote commitment [19]. Communication plays an important role here and the common goal should be known and respected within the team and organization [10].

6.2 Team Orientation

This dimension can be directly found in the framework of Salas et al. showing improved individual effort and performance. Lack of team orientation respectively leads to demotivation, social loafing, diffusion of responsibility and sucker effects, thus lowering the cohesion of the team [22]. Moe et al. [10] have found out that team members gave a too high priority to individual goals rather than team goals. Shared team orientation promotes cohesion of the group and counteracts social loafing as team members perceive that the task and the team itself is important [9]. Organizations [23] with greater influence of task skills as well as rewarding systems for team performance increase team cohesion and team orientation [18]. Job rotation and a culture of trust in collaboration can help to improve this and cross training can be valuable by increase the team's flexibility [10].

6.3 Redundancy

The concept of redundancy is equivalent to the characteristics of Backup Behavior described by Salas et al. [17]. Cross-functionality allows members to substitute each other in case of demand creating involvement and innovation of team members due to broader expertise. It is reported as crucial for self-managing teams [14] and appears as "multiskilling" in socio-technical literature [5]. Lack of Redundancy means specialization of team members, dependency of task accomplishment on availability of certain team members leading to bottlenecks when these are unavailable. It also leads to a general lack of diversified views enhancing the product due to concentration of knowledge.

To improve redundancy literature generally recommends to collocate the team in the same room [1]. Moe et al. [10] recommend to appreciate generalists inside the team and company culture and to select them during team building and recruitment. Job rotation can further contribute to improve knowledge redundancy by integrating knowledge from different domains [6].

6.4 Learning

Learning describes a team's ability in identifying weak points and improving the development process. It is one of the ideas of Scrum originating from the new product development literature [19] known as *multi-learning*. Multilevel and multifunctional learning allow team members to acquire broad knowledge outside their direct product scope, allowing the team to respond quickly and to solve problems fast [19]. Job rotation can help to integrate knowledge from different domains and appreciation of organizational concerns [6], but must be legitimized by the organization. Efforts to collect data and to improve should be motivated by the desire to learn and improve inside the team and should not be used to push organizational control.

6.5 Autonomy

Team autonomy is necessary so the Scrum team perceives its total responsibility over the product without external influence on the team's work plan inside a sprint. It is described as the influence of management and other individuals outside the team. Lack of team autonomy is believed to lead to excessive overtime, high defect rates and personnel burnout. In Scrum, it damages the concept of self-organization [2], thus disturbing the team cohesion. Autonomous and self-organizing teams are recognized as a premise, but also as one of the biggest challenges of agile methods [4].

Autonomy of a team is affected by individual as well as organizational level, self-management thus must be fostered on both levels [10]. Assigning people on more than one project at a time leads to competing for team members and unequal distribution of resources, thus should be avoided [10]. Collocation of the the team in the same room further helps [10] .

7 Conclusion

In this paper we presented and empirically validated a tool to improve reflection of agile software development teams. With this survey tool, we measured the cornerstones of agile teamwork in 5 dimensions. We found that the organizational and individual levels of autonomy and redundancy are the dimensions with the lowest scores as given by the users of this tool. This finding is consistent with the original findings of [11,10]. We introduced a measure for agreement regarding the dimension measurements and found that it was high in our empirical study. This indicates both that team members have similar notions of each dimension and how it applies to their particular team situation. In addition, the teams found the survey tool in general a useful method to reflect.

For future use a dedicated web application can be provided to improve the usability and accessibility of the tool, as the current data has been collected with a generic survey application. Data collection via a customized online tool would furthermore allow the collection of additional teams contributing to the framework's improvement and recommendations could be provided and updated online according to latest research. Psychometric scale questions [21] could be incorporated into future versions of the survey tool to be able to measure the degree of accuracy or truthfulness the participant tends to give to the answers.

Although processes and routines are recognized as organizational capital there is currently little tooling to support their design and the effects are much more uncertain. We believe that quantitative data collection methods applied within teams can provide intermediate feedback and help us to understand organizational process implementation and change.

Acknowledgments. The work in this paper was kindly supported by the EDAFMIS project in the framework of ITEA2.

References

1. Allen, T.J.: Communication networks in R & D laboratories. R & D Management 1(1), 14–21 (1970)
2. Boehm, B., Turner, R.: Balancing Agility and Discipline: A Guide for the Perplexed. Addison-Wesley Longman Publishing Co., Inc., Boston (2003)
3. Digman, J.: Personality structure: Emergence of the five-factor model. Annual Review of Psychology 41, 417–440 (1990)
4. Dybå, T., Dingsøyr, T.: Empirical studies of agile software development: A systematic review. Inf. Softw. Technol. 50, 833–859 (2008)
5. Emery, F., Thorsrud, E.: Democracy at Work - The Report of the Norwegian Industrial Democracy Program. Springer, Heidelberg (1976)
6. Fægri, T.E., Dybå, T., Dingsøyr, T.: Introducing knowledge redundancy practice in software development: Experiences with job rotation in support work. Inf. Softw. Technol. 52, 1118–1132 (2010)
7. Guzzo, R.A., Dickson, M.W.: Teams in organizations: Recent research on performance and effectiveness. Annual Review of Psychology 47(1), 307–338 (1996)
8. Hewitt, B., Walz, D.: Using shared leadership to foster knowledge sharing in information systems development projects. In: HICSS 8, p. 256a (2005)
9. Karau, S., Williams, K.: Social loafing: A meta-analytic review and theoretical integration. Journal of Personality and Social Psychology 65(4), 681–706 (1993)
10. Moe, N., Dingsøyr, T., Dybå, T.: Overcoming barriers to self-management in software teams. IEEE Software 26, 20–26 (2009)
11. Moe, N., Dingsøyr, T., Røyrvik, E.: Putting agile teamwork to the test – an preliminary instrument for empirically assessing and improving agile software development. In: Abrahamsson, P., Marchesi, M., Maurer, F. (eds.) Agile Processes in Software Engineering and Extreme Programming. LNBIP, vol. 31, pp. 114–123. Springer, Heidelberg (2009)
12. Moe, N.B., Dingsøyr, T.: Scrum and Team Effectiveness: Theory and Practice. In: Agile Processes in Software Engineering and Extreme Programming pp. 11–20 (2008)
13. Moe, N.B., Dingsøyr, T., Dybå, T.: A teamwork model for understanding an agile team: A case study of a scrum project. Inf. Softw. Technol. 52, 480–491 (2010)
14. Morgan, G.: Images of Organization. SAGE Publications, Thousand Oaks (2007)
15. Paulhus, D.L.: Socially desirable responding: the evolution of a construct. In: The Role of Constructs in Psychological and Educational Measurement, pp. 46–69. Lawrence Erlbaum, Mahwah (2002)
16. Pearce, C.L.: The Future of Leadership: Combining Vertical and Shared Leadership to Transform Knowledge Work. Academy of Management Executive 18(1), 47–57 (2004)
17. Salas, E., Sims, D., Burke, C.: Is there a big five in teamwork? Small Group Research 36(5), 555–599 (2005)
18. Shea, G., Guzzo, R.: Group effectiveness: What really matters? Sloan Management Review 28, 25–31 (1987)
19. Takeuchi, H., Nonaka, I.: The new new product development game. Harvard Business Review (1986)
20. Tata, J., Prasad, S.: Team Self-Management, Organizational Structure and Judgments of Team Effectiveness. Journal of Managerial Issues 16(2), 248+ (2004)

21. Thompson, E.R., Phua, F.T.T.: Reliability among senior managers of the marlowe-crowne short-form social desirability scale. Journal of Business and Psychology 19(4), 541–554 (2005)
22. Thompson, L.: Making the Team. ch. 2, ch. 4. Prentice-Hall, New York (2002)
23. Walton, R.E., Hackman, J.R.: Designing Effective Work Groups. Jossey-Bass Publishers, San Francisco (1986)

A Detailed Software Process Improvement Methodology: BG-SPI

Banu Aysolmaz and Onur Demirörs

Informatics Institute, Middle East Technical University, 06531 Ankara, Turkey and Bilgi Group Ltd. Middle East Technical University Technopolis, 06531 Ankara, Turkey
{aysolmaz,demirors}@metu.edu.tr

Abstract. Software Process Improvement (SPI) methodology is defined as definitions of sequence of tasks, tools and techniques to be performed to plan and implement improvement activities. Well-known SPI frameworks like CMMI and ISO/IEC 15504 define SPI methodologies in an abstract manner. We developed an SPI methodology, BG-SPI, providing a ready-to-use SPI scheme with guidance on an iterative SPI lifecycle, composed of task definitions with details on resources, tools, roles, participation of groups, process assets, and other process specific supporting items. Utilizing BG-SPI with support of SPI experts, organizations can easily plan and manage SPI lifecycle. BG-SPI methodology is applied over 10 organizations with different size, sectors and SPI motivations. This paper explains BG-SPI and provide insight on how a detailed SPI methodology helps the SPI projects in various aspects.

Keywords: Software process improvement, SPI, SPI methodology, SPI lifecycle, CMMI, ISO/IEC 15504.

1 Introduction

Software Process Improvement (SPI) focuses on improving the time, cost and quality of engineering and management practices in software organizations. SPI initiatives in software organizations are frequently performed based on well defined reference models such as CMMI and ISO 15504.

SPI methodology can be defined as definitions of sequence of tasks, tools and techniques to be performed to plan and implement improvement activities [3]. SPI methodologies are described as part of or in relation to well-known process improvement frameworks like ISO/IEC 15504 [4], CMMI [6]. Other SPI frameworks that describe SPI methodologies are SPIRE[7], SATASPIN [8], PRISMS [9], MESOPYME [2], MoProSoft [10], MPS [11]. Related parts of the two most frequently used reference models, ISO/IEC 15504-4 [5] and IDEAL[12] define SPI methodology in a broad sense. They emphasize the importance of planning SPI activities, name the key activities and concepts and describe their relations. Despite detailed descriptions of reference models, tools and techniques and types of assessment methods, the existing frameworks do not provide a detailed SPI methodology to plan and manage SPI initiatives.

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 97–108, 2011.

In the methodologies defined, the organization of key concepts and implementation details are left to the practitioners. They suggest SPI programs to be organized in waterfall-like lifecycle and in a top-down fashion [7, 9]. As a result organizations frequently do not utilize these methodologies and define their own approach to manage the SPI projects.

We have developed BG-SPI (Bilgi Group Software Process Improvement Methodology), that provide detailed guidance on how to conduct SPI including the lifecycle, tasks, approach, resources, tools and other supporting assets. Utilizing BG-SPI with the support of independent SPI experts, organizations can easily plan and manage the SPI lifecycle by tailoring a ready-to-use SPI scheme. Compared to traditional approaches, BG-SPI provides an agile approach to conduct process improvement in short term increments with a clear long term roadmap. BG-SPI includes detailed inscriptions of SPI activities, also process assets and documentation. Organizations that utilize BG-SPI eliminate the demotivation to conduct SPI caused by ambiguity, achieve short term benefits and overall enable SPI projects to be conducted more effectively.

BG-SPI is utilized to conduct SPI activities in ten different software organizations with different characteristics. SPI consultants from Bilgi Group have been involved to guide the application of the methodology. It is observed that utilizing a detailed SPI methodology enhances the SPI application in many ways. In this paper, we explain the BG-SPI and present the benefits observed in the SPI activities of the organizations.

The rest of the paper is organized as follows. Section 2 provides general information on SPI methodologies. Section 3 explains BG-SPI with its process description. Section 4 discusses SPI experiences utilizing BG-SPI and the results. Section 5 discusses the conclusions and lessons learned.

2 SPI Methodologies

SPI methodologies defined as part of SPI frameworks are composed of high level activities or phases. They provide guidelines and highlight points to be considered while organizing the SPI activities. For example, ISO/IEC 15504-4 [5] suggests defining an SPI lifecycle and an action plan in Process Improvement Programme Plan. Other than highlighting major headings of the action plan, many decisions are left to the practitioners. Similarly, IDEAL [12] advises the activities to be determined in the “improvement agenda” of the SPI strategic plan. The practitioners on the other end need a practical SPI scheme, specific guidelines on how they will organize and conduct SPI; and detailed descriptions of activities.

The frameworks for small and medium sized organizations provide more detail on the implementation of SPI, as these organizations require ready to use descriptions. MESOPYME [2] provides a specific implementation approach. With the same perspective of this study, [2] discusses that specific implementation solutions are required to plan SPI activities, and “current methods do not provide guides to elaborate these elements”. The solution is using “action packages” to start SPI activities in process areas, proven to be successful in their experiences.

SPIRE [7] is a framework suggesting a more traditional approach as the SPI methodology. With a waterfall lifecycle, it focuses on benefits of SPI and maintaining the plan. Being an earlier framework, it provides an experience base. PRISMS also utilize waterfall lifecycle, focusing on business goals to identify key process areas [9]. With a top-down approach, planning is conducted by quality experts and implementation by process owners, as does OWPL [19].

ASPE-MSC [20] explains a detailed SPI methodology with an iterative lifecycle, emphasizing the SPI plan preparation. Competisoft [13], an evolution of MoProsoft [10], defines an incremental improvement process influenced also from agile methodologies. The model defines the roles, expected work products and a template for these products. Consultant guide is also suggested at Competisoft. SATASPIN [8] provides a good example of distributed SPI initiatives.

All these frameworks focus on the fact that organizations need more guidance to initiate and conduct SPI, suggesting solutions for different aspects they focus. They all infer a top-down paradigm, usually with a centralized mechanism. The lifecycles vary. As the roles to be involved, some suggest well-defined allocation of the responsibilities and involvement of external experts. They are based on and encourage use of a well-known SPI framework like CMMI or SPICE.

Our experiences also support that the initiating step of SPI is the most critical, as organizations find it hard to plan the activities without a guidance. To overcome the problems and provide a well-defined SPI guidance, we developed an SPI methodology called BG-SPI. It utilizes the ideas and best practices from the summarized SPI frameworks, together with other standards and experiences. BG-SPI methodology is explained in the next section.

3 BG-SPI Methodology for SPI

BG-SPI is an SPI methodology that defines a process to implement SPI activities in an iterative lifecycle. It also includes related process assets that organizations can utilize as a baseline to fulfill the requirements of SPI models and standards like CMMI [6], SPICE [4], ISO 9001:2008 [15] and IEEE Software Engineering standards [16, 17, 18]. BG-SPI follows the outline of SPI process suggested by commonly known models like IDEAL [12], ISO/IEC 15504-4 [5]. It also incorporates the best practices suggested by other more detailed SPI frameworks like MESOPYME [2], OWPL [19], ASPE-MSC [20], PRISMS [9], MoProSoft [10]. BG-SPI presents a well-defined set of activities for a practical SPI implementation by providing a ready-to-use SPI scheme and generic set of process definitions and assets. BG-SPI is utilized and enhanced in many SPI initiatives of different kinds of organizations.

From a high level perspective, BG-SPI process is similar to those of the ISO/IEC 15504 Part 4 [5] and IDEAL model [12]. However, BG-SPI aims to provide a specific guide. The SPI process of BG-SPI is depicted in Fig. 1. The process is modeled using eEPC notation [24] which is based on activity flow. The columnar view is used, where the activities of the roles (shown at the top of the columns) are depicted along the related column. Each step of the methodology is explained in the following sections.

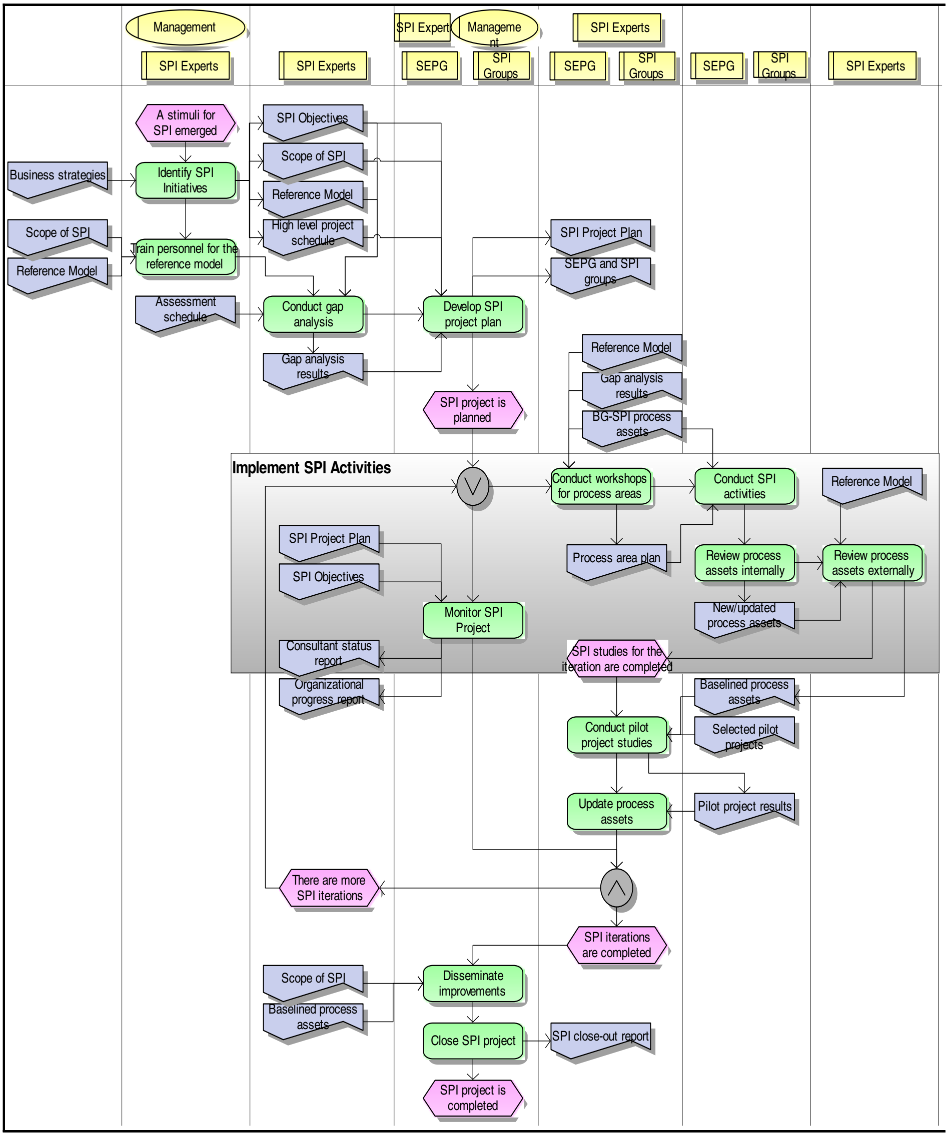


Fig. 1. The implementation process of BG-SPI Methodology

3.1 Identify SPI objectives

This step starts when an SPI stimuli emerges at the organization. For some, the main stimulus is certification enforced by industry, while some consider marketing, or some are only motivated to cut costs, improve timeliness and quality aspects of their processes.

At this point, with an external view, it is important to determine organization's real expectation from SPI, normalize it with respect to business strategies, and document as "SPI objectives". Organizations may select to apply SPI in some or all of its related departments and project types, determining the "scope of SPI".

Considering the business strategies and the SPI stimuli, "reference model" for assessment and improvement is identified as CMMI or SPICE. Considering organization-specific issues and target maturity level, high level details and tailoring to reference models are identified. A "high level project schedule" is developed.

3.2 Train Personnel for the Reference Model

In this phase, the organization is ready to initiate technical SPI activities. Considering the scope of SPI, the related organizational units are identified. Training and workshop sessions of the chosen reference model are conducted. To provide a common understanding of the SPI activities, a general SPI training is conducted. In this way, awareness and knowledge of SPI is disseminated for the related organizational units. The workshops also establish a baseline for the next step of conducting gap analysis.

3.3 Conduct Gap Analysis

To identify the current state of the organization, an assessment is conducted by SPI experts to evaluate the conformance to the reference model. SPI experts are a group independent of any process area studies and they are frequently consultants. A detailed "assessment schedule", including interviews and related personnel, is identified. Conducting the assessment, the gap between the reference model and the current state is identified, analyzed and documented as "gap analysis results".

The assessment is critical, as it is the basis for revealing the improvement opportunities and determining the detailed SPI schedule. Moreover it has a distinctive importance as it is the initial step to actively disseminate and internalize the idea of SPI in the organization. The personnel coordinating the assessment usually become the ones in software engineering process group (SEPG) (as used in IDEAL [12]), the group coordinating and managing the SPI project. Also, the personnel in assessment interviews provide insight on who should participate in the future process area improvement groups (named here as SPI groups).

3.4 Develop SPI Project Plan

Materializing the results of the gap analysis and utilizing the inputs of the previous steps, the improvement lifecycle model is identified and SPI activities are planned in "SPI Project Plan". This step is conducted as a project planning process, ending with a work breakdown structure, schedule, resources and budget.

Implementing SPI as a project is suggested by most of the SPI frameworks like ISO/IEC 15504-4 [5], IDEAL [12, 22]. Although this is a common approach, we encounter problems in practice. Our experiences show that it is hard to get acceptance for the idea to conduct SPI as a project, as it is much different from development projects. To assure its benefits, the management level is explained that by means of managing SPI activities as a project, allocation of resources and monitoring of objectives can be assured. Technical people are explained that they can transparently dedicate their time to SPI activities and expose their efforts and the results clearly to the management level. In this way, internalization of SPI is highly enhanced. Hence, this supports that determining project drivers for an SPI implementation is an important factor to enhance the commitment of individuals [21].

During SPI project planning, existing SPI know-how and assets are tailored. First, a "Software Engineering Process Group (SEPG)" is established. A ready to use list of responsibilities is suggested to and tailored for the organization. SEPG is defined as the group to coordinate and monitor all SPI activities, be the communication point among all stakeholders, manage and synchronize all assets and report to the upper management.

Involving process owners in SPI activities from the start is important to provide ownership and confidence for SPI [23]. Considering this, the groups responsible for the process areas (named as "SPI groups") are identified. They are usually assigned for the high level process groups like management, engineering, support and process management. Within these SPI groups, lower level responsibilities are identified for specific process areas. The SPI project plan is established and approved by all related groups including management, SEPG and SPI groups. In this way, commitment is obtained for the SPI activities.

BG-SPI suggests an iterative lifecycle model for the SPI activities. The project is composed of iterations that are micro-improvement cycles of about 6 months. It is of high importance to prioritize the goals and divide the increments accordingly [1]. Considering this point, the allocated time for each process area is identified using the results of the gap analysis. The cycles are identified to cover all processes of a high level process area like support and management areas. This also simplifies the organization with the SPI groups and implementation of the pilot projects.

If not required otherwise, initial processes are determined as the ones focusing on process definition and improvement. In this way, the approach to define processes, build up organizational process asset repository, and determine approach for improvement is identified at the beginning. The studies for the rest of the process areas utilize this approach to develop their own process assets, which assures a standard way of process development from the beginning. This approach also lets the studies to start with only the SEPG and SPI experts together. In this way, the group establishes an internalized workshop format and utilizes it for the rest of the studies.

During the gap analysis, organization's training needs are revealed. The identified training needs are integrated with the project plan. The trainings are placed before the start of the related process area workshops. In this way, it is ensured that the SPI groups are knowledgeable and motivated for the related SPI studies.

In all SPI iterations, implementation of related processes in pilot projects, and incorporating identified improvements are planned. Even if some processes are found to be meeting the requirements of the reference model fully during gap analysis, they are placed in the schedule so that that process can be synchronized with the new studies and further improvement opportunities can be identified.

3.5 Implement SPI Activities

This step in BG-SPI process is composed of several activities and shown with gray area in Fig.1.

The first activity is “conduct workshops for process areas”. The approach of BG-SPI for implementation requires a strong participation with SPI experts. For each process area as identified in the SPI project plan, SPI experts and related SPI group starts with a workshop. Workshops are usually organized once in a week or two weeks. A predefined workshop format is followed. SPI experts start the workshop with an introduction of the reference model for that process area. The findings from the gap analysis are discussed. The generic process definitions and assets provided in BG-SPI methodology (“BG-SPI Process Assets”) are analyzed. Organization specific characteristics and practices are discussed by working over these assets. The possible utilization and adoption to BG-SPI assets are identified. Available organizational assets, infrastructure and tools are evaluated, alternatives and solutions are discussed to fit to SPI objectives. Accordingly, a list of to-do’s is identified, including the process definition, process assets, organizational activities, decisions to be given, infrastructures to be established and reviews. All these activities are planned in a “process area plan” with the consent of SPI experts, including work items, responsibilities, resources and dependencies.

Second activity is “conduct SPI activities”. After the workshop the SPI group, with the coordination of SEPG, conducts the activities as planned in detailed process area plan. Before SPI expert review, prepared assets undergo an internal review mechanism, the activity of “review process assets internally”. During the activity of “review process assets externally”, the SPI experts verify the processes for conformity to the reference model and standards to be followed, compatibility with the existing and newly defined organizational assets and consistency between the processes. In the next workshop, open and completed work items are revised and discussed. The completed process assets are baselined and placed under configuration control.

The other activity during this step, parallel to other studies is to “monitor SPI Project”. SEPG prepares “organizational progress reports” as defined in the SPI project plan. The SPI experts prepare monthly “consultant status reports”, evaluating the progress both in high and low level work items, and comparing the actual and planned status of objectives.

With this approach, smaller iterations for process areas (micro improvement cycles) are conducted within each bigger SPI iteration. With the help of the BG-SPI approach and support of the SPI experts, the roadmap is well defined, preventing the confusions in the SPI lifecycle. Also, the SPI lifecycle is continuously under control and monitored with a joint mechanism of workshops and progress reports.

3.6 Conduct Pilot Project Studies and Update Process Assets

Upon completion of an iteration, a set of pilot processes are selected to apply the improved processes. The number of projects are kept between 2-4, as too many projects would make the pilot studies hard to manage. If the project durations are too long in the organization, projects which are in different phases of their lifecycle are chosen. In this way, different process areas can be implemented in a shorter time.

The outcome of this activity is "pilot project results". These results are utilized to identify the needs for changes and update the SPI project plan. The SEPG again has a critical role to assure that project members apply the new processes, collect the negative and positive feedbacks, find out the points they cannot apply, and manage the new updates to processes accordingly.

As the pilot projects are conducted for each high level iteration, the feedbacks are obtained for the iterations, too. This enables parallel work while the group starts studies for the next iteration. The results of the first pilot projects can be utilized as input to the next iteration. Also, the step by step implementation of processes eases the adoption by the organization.

3.7 Disseminate Improvements and Close SPI Project

Upon completion of the iterations, the dissemination of new processes to all organization is started. This step is critical for sustaining the improvements in the organization, and conducted with a dissemination strategy. Usually, the implementations of the new processes are initiated with the newly starting projects. In this way, a step-wise transition occurs.

The activity of "close SPI project" is conducted when completion criteria are completed as identified in SPI project plan. The results are documented with "SPI close-out report". For continuous improvement, it is advised that SPI is conducted with recurring projects in the organization.

As discussed before, lifecycle model utilized by the SPI methodologies is a distinctive property. BG-SPI is employing an iterative life cycle. In the high level, SPI project is planned to include high level iterations including process area sets. The iterations are further divided to micro improvement cycles, each planned in more detail before the start. This approach brings both good control and flexibility for managing SPI activities; providing better response to uncertainty while providing good management practices. For the application paradigm, BG-SPI applies a distributed approach. While activities are initiated in a top-down manner with the support of the management with a high level plan covering all process areas, the technical groups (SEPG and SPI groups) initiate detailed studies in low level as planned in high level schedule.

The most important aspect of BG-SPI is its level of detail in terms of guiding SPI studies. The BG-SPI provides a well-defined template of SPI activities, which eliminates the risk of demotivation caused as the organization can't foresee its roadmap and stay in confusion. Indeed, an important part of the SPI studies is about the social aspects. The importance of management commitment is a well-known issue [14, 18]. ISO/IEC 15504-4 [5] mentions the risk of "senior management not expressing informed, sustained commitment". To overcome this issue, the involvement of the management is planned throughout the BG-SPI lifecycle, the objectives are made clear, and the results of the studies are quantified with the reports. SPI project management process is the most important tool for this, by means of which resources are allocated and SPI expectations and objectives are identified clearly. This provides not only management commitment, but also internalizing SPI studies throughout the whole organization. Changing the organizational culture to conduct SPI in a habitual way is the most critical benefit of the SPI methodology. For this, providing the management commitment and making SPI part of everyone's work, as assured with the SPI project, is critical. Through the application of BG-SPI, it is also the responsibility of the SPI experts to highlight the achievements of the group and emphasize the benefits acquired.

A methodology which is so deterministic may have the risk of not being appropriate to many cases. We have applied BG-SPI in many different SPI studies, as summarized in the following section. The results assure that BG-SPI is applicable to many different cases, and know-how can be utilized for many other cases too.

4 Experiences on Applying BG-SPI

We have implemented BG-SPI in a number of organizations to conduct SPI activities. The size of the related units of these organizations differ. In this paper we provide our observations on 10 implementations. One of the ten organizations is a micro-enterprise, having a development team of size less than 10, and one has size 10-25. Others are larger, three having unit size between 25-50, and the rest between 50-100. The sectors and application types also vary between banking, military, government and embedded applications. The organizations from military sector has for official certification goals due to acquisition regulations in Turkey. 9 out of 10 organizations were aiming to reach a maturity level of established processes. Planned durations of the SPI initiatives were about 13-15 months. Four of the six organizations that planned certification acquired the certification successfully. The remaining two are continuing activities as planned.

At the moment, SPI activities are going on in three organizations. Five of the organizations completed their first SPI projects as planned, with or without certification. Two of them canceled the SPI initiative after covering parts of the SPI plan including definition of processes. In one of the organizations, the reason of cancellation is change in the stimuli for conducting SPI. The other had to stop the initiative due to unavailability of related resources.

Considering these experiences, we observe some factors facilitating the success of SPI. As mentioned in many studies, management commitment and involvement of process owners are found to be the key points for success. When these are not achieved on time, process improvement teams may have the illusion that the studies are going well, until they encounter a resistance at the time of the pilot projects. This is caused by the fact that process owners don't know the new processes and doubt if it is applicable for them. To overcome this, it is very critical to manage the involvement of the process owners and establish SPI as part of the organizational culture. If the management can't take actions to break the resistance, usually SPI initiatives do not achieve all of their goals.

Other principle for effective management of the SPI projects is found to be the certification goal. Organizations with certification pressure from the acquirers are much effective in planning and implementing SPI projects. This is caused by the fact that the obligation enhances the management commitment, and the personnel, rather than questioning the initiative, focus on achieving the objectives as planned.

We observed that the organizations from similar sectors encounter similar technical problems while they develop process assets, and the domain knowledge helps a lot to solve these problems. For example, military organizations usually develop huge long projects involving many technical difficulties. Companies developing banking applications usually develop small projects attached around a framework application; which makes it hard to differentiate between maintenance and new development. Despite these variations, we haven't identified any factor of success depending on the sector. We observe that the existence of the SPI experts, especially external consultants, increase the success potential of SPI projects. Indeed, by means of BG-SPI, the dependency on SPI experts decreases as all activities are well-defined. However, the existence of the SPI experts stimulate the motivation to better allocate resources, and enable the personnel to get focused as they perceive SPI experts as a sign of management commitment. The personnel feel comfortable to know they can consult them in case of problems. Also, expensive rework in later phases is largely eliminated by an independent verification and validation mechanism.

Another aspect of BG-SPI methodology is the ready-to-use process assets. In many cases, the organization can use these time tested assets with minimum tailoring effort, and the groups can easily determine the list of action items by going over these assets. The usage of these generic process assets may bring the internalization risks for the processes. However, in our studies, we observed that this property enhanced the practicality of BG-SPI, decreasing the time to develop organizational process assets. As all of the SPI lifecycle is planned in a well-defined manner, the organization already establishes the approach to develop the assets, which prevents the feeling that the assets don't belong to them. It is still important to keep in mind that for process areas with no or very little previous implementation, the group is more eager to use available process assets of BG-SPI. However, relying on these assets makes the settlement of these processes difficult. This is a risk to consider during definition of process assets. To prevent this risk, it is important to train the personnel and increase the awareness, and motivate them to reveal the solutions matching better to the organization's way of doing things.

5 Conclusion and Future Work

In the high level, BG-SPI follows the high level approaches suggested by well-known SPI frameworks like CMMI, ISO/IEC 15504. Like IDEAL, BG-SPI divides SPI implementation into phases. To apply those, it provides detailed practical guidelines. It follows some principles of iterative and agile development philosophies to detail the activities. The iterative planning in high level, dividing the iterations more as micro-improvement cycles and increasing the planning details before starting the activities are examples of these.

Other important aspects of BG-SPI are the weekly meetings, intensive involvement of process owners, close review and reporting mechanisms. Starting the studies with process areas of organizational process definition and improvement is another highlight of BG-SPI, which is an approach to institutionalize and internalize processes. BG-SPI also utilizes the SPI experts in the lifecycle to enhance the effectiveness of SPI process, but prevents to create dependency on them, which improves the sustainability of SPI programs. In our experiences, this ready-to-use scheme of BG-SPI is found to be helpful for the critical step of SPI planning, resulted in successful SPI implementations and fostered internalization of SPI practices.

The ratio of certificated organizations is a measure of success for BG-SPI. All four of the organizations that completed the SPI activities got official certification as planned.

For the future work, we find it important to collect SPI data in a systematic way in later SPI programs so that we can derive more extensive conclusions from the experiences. These conclusions will constitute an important repository for SPI area that can be utilized by other practitioners and academicians in the field. Also, in this way, BG-SPI will be improved and better meet the needs in the field.

Another improvement opportunity we foresee is to customize BG-SPI to meet different needs of sectors. The customization can give the most benefit for process assets provided by BG-SPI. By incorporating domain knowledge into the process assets, we can speed up the SPI process definition activities during SPI implementation.

References

1. Wiegers, K.E., Sturzenberger, D.C.: A Modular Software Process Mini-Assessment Method. IEEE Software 17(1), 62–69 (2000)
2. Calvo-Manzano, J.A., Agustin, G.C., Gilabert, T.S.F., Seco, A.D.A., Sanchez, L.Z., Cota, M.P.: Experiences in the Application of Software Process Improvement in SMES. Software Quality Journal 10, 261–273 (2002)
3. Yeşildoruk, F.Ç., Bozlu, B., Demirörs, O.: The Tool Coverage of Software Process Improvement Frameworks for Small and Medium Enterprises. In: Bomarius, F., Oivo, M., Jaring, P., Abrahamsson, P. (eds.) PROFES 2009. LNBIP, vol. 32, pp. 290–302. Springer, Heidelberg (2009)
4. ISO: Software Process Assessment - Part 2: A reference model for processes and process capability. Technical report. ISO/IEC 15504 TR2:1998, International Organization for Standardization (1998)
5. ISO: Software Process Assessment - Part 4: Guidance on use for process improvement and process capability determination. Technical report. ISO/IEC 15504-4:2004, International Organization for Standardization (2004)
6. SEI.: CMMI for Systems Engineering, Software Engineering, Integrated Product and Process Development, and Supplier Sourcing (CMMI-SE/SW/IPPD/SS, V1.1) Staged Representation. Technical report, CMU/SEI-2002-TR-012 ESC-TR-2002-012, Software Engineering Institute (2002)
7. Sanders, M. (ed.) The SPIRE Handbook—Better, Faster, Cheaper: Software Development in Small Organisations. Dublin City University, Center for Software Engineering (1998)
8. Varkoi, T., Mäkinen, T.: Software process improvement network in the Satakunta region: SataSPIN. In: Proceedings of the EuroSPI 1999, Pori, Finland (1999)
9. Allen, P., Ramachandran, M., Abushama, H.: PRISMS: An Approach to Software Process Improvement for Small to Medium Enterprises. In: Proceedings of the Third International Conference on Quality Software, Dallas (2003)
10. Oktaba, H.: MoProSoft: A Software Process Model for small enterprises. In: Proceedings of 1st International Research Workshop for Process Improvement in Small Settings, pp. 93–101. SEI (2006)
11. Montoni, M., Santos, G., Rocha, A.R.: MPS Model and TABA Workstation: Implementing Software Process Improvement Initiatives in Small Settings. In: Fifth

International Workshop on Software Quality, WoSQ 2007, ICSE Workshops, Minneapolis (2007)
12. McFeeley, B.: IDEAL: A User's Guide for Software Process Improvement. Technical report, CMU/SEI-96-HB-001, Software Engineering Institute, Carnegie Mellon University (1996)
13. Oktaba, H., García, F., Piattini, M., Ruiz, F., Pino, F.J., Alquicira, C.: Software Process Improvement: The Competisoft Project. IEEE Computer, 21–28 (2007)
14. Dyba, T.: Factors of Software Process Improvement Success in Small and Large Organizations: An Empirical Study in the Scandinavian Context. In: Proceedings of the 9th European Software Engineering Conference, Helsinki, Finland, pp. 148–157 (2003)
15. ISO 9001: 2008, Quality management systems – Requirements, International Standards Organization (2008)
16. ISO/IEC/IEEE: Systems and software engineering – Life cycle processes – Project management. ISO/IEC/IEEE 16326:2009, 2009-12-15, Switzerland (2009)
17. Software Engineering Standards Committee of the IEEE Computer Society: IEEE Recommended Practice for Software Requirements Specifications. IEEE Std 830-1998, New York, USA (1998)
18. Software Engineering Standards Committee of the IEEE Computer Society: IEEE Recommended Practice for Software Design Descriptions. IEEE Std 1016-1998, New York, USA (1998)
19. Alexandre, S., Renault, A., Habra, N.: OWPL: A Gradual Approach for Software Process Improvement in SMEs. In: Proceedings of the 32nd EUROMICRO Conference on Software Engineering and Advanced Applications (EUROMICRO- SEAA 2006), pp. 328–335 (2006)
20. Wangenheim, C.G.v., Weber, S., Rossa Hauck, J.C., Trentin, G.: Experiences on establishing software processes in small companies. Information and Software Technology 48(9), 890–900 (2006)
21. Abrahamsson, P., Livari, N.: Commitment in Software Process Improvement. In: Search of the Process, Proceedings of the 35th Hawaii International Conference on System Sciences (2002)
22. Kautz, K., Hansen, H., Thaysen, K.: Applying and Adjusting a Software Process Improvement Model in Practice: The Use of IDEAL Model in a Small Software Enterprise. In: Proceedings of the 2000 International Conference on Software Engineering, ICSE 2000, The New Millennium, pp 626–633 (2000)
23. Mishra, D., Mishra, A.: Software Process Improvement Methodologies for Small and Medium Enterprises. In: Jedlitschka, A., Salo, O. (eds.) PROFES 2008. LNCS, vol. 5089, pp. 273–288. Springer, Heidelberg (2008)
24. Davis, R., Brabänder, E.: ARIS Design Platform: Getting Started with BPM, 1st edn. Springer, Heidelberg (2007)

Motivation and Empowerment in Process Improvement

Marion Lepmets and Eric Ras

Public Research Centre Henri Tudor,
29 J.F.Kennedy ave., Luxembourg, Luxembourg
{Marion.Lepmets,Eric.Ras}@tudor.lu

Abstract. Clarity in goals, tasks, and responsibilities empowers employees to undertake an organizational change. Assessing processes prior to process improvement allows high involvement in setting the improvement goals, awareness of the organization's business goals, and understanding the roles and responsibilities in process improvement. This study describes the results of an international survey about goal internalization, motivation, and empowerment in process improvement. The results indicate that process assessment contributes to the goal internalization of process improvement. However, process assessment alone does not contribute extensively to the empowerment and motivation of employees.

Keywords: process improvement, process assessment, goal internalization, motivation, involvement, empowerment.

1 Introduction

A lot of studies have been conducted on process improvement in the last two decades, particularly in the software engineering domain. There are numerous case studies about the success and the key success factors of process improvement [1-7]. Yet over 70% of the process improvement projects fail because of the poor understanding of the competencies, roles, and responsibilities in process improvement activities [8].

In the past 20 years, empowerment has been accepted by a vast majority of enterprises managers and been applied in their practices [9]. Based on Spreitzer [10], there are two contrasting perspectives of empowerment: the relational and the psychological. While the relational perspective views empowerment as an objective top-down process of power, the psychological perspective views empowerment as a subjective phenomenon. The latter is achieved only when a perception of empowerment within the employee is produced. Psychological empowerment has recently received more and more attention because it is believed to be central to many behavioral outcomes sought by practitioners [9].

According to Menon [11], psychological empowerment is a cognitive state characterized by a sense of perceived control, perception of competence and internalization of organization's goals, and objectives. Menon suggests that the internalization of organization's goals captures the energizing property of a worthy cause or exciting vision provided by the organizational leadership. Goal internalization has been related to becoming more motivated in executing ones tasks and in becoming

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 109–120, 2011.

inspired of organizational goals and the organizational vision [11]. High involvement in decision-making and clarity in goals and responsibilities in processes has also been found to motivate practitioners in process improvement [8, 12-14].

In this study, we would like to find out if goal internalization motivates and inspires employees also in carrying out process improvement. Process improvement initiatives are the means to develop the organization's processes to more effectively meet the organization's business goals. Process assessments are used to find out the capability of the processes to reach these goals [15].

In psychological empowerment, it is believed that the valued goals can be set only as a result of lining the transformational attitudes and beliefs of the employees with the organization's mission and objectives. The effect of the transformational influence is to energize employees to participate in the process of transforming the organization [11]. In other words, high involvement in setting process improvement goals should provide motivation and inspiration in working towards the organization's business goals. This in turn, will lead to higher possibilities of success in process improvement initiatives.

We conducted an international survey among process improvement industry to find out how goal internalization can affect and inspire employees to carry out the change in their organizations. In the following chapters we describe the goals of the study, the design of the survey based on the measurable goal internalization items described by Menon [11], provide the sample description, and the results of the data analyses. We close the paper with possible future works.

2 Research Goals and Methodology

Resistance to organizational change by employees is the major obstacle in reaching success in process improvement. Based on the empowerment studies [11-13], we know that high involvement in setting the improvement goals, awareness of the organization's business goals, and their alignment to the improvement goals motivates employees to carry out and contribute to an organizational change like process improvement.

Process assessment can be seen as a means to involve and grow the awareness of process improvement goals throughout the organization. Process assessment allows revisiting and communicating the organization's business goals to all involved in process improvement, and aligning the goals of process improvement to the organization's business goals. This involvement can, in turn, contribute to the goal internalization and motivation among the employees.

We would like to find out if the organizations who conduct process assessment before starting a process improvement have higher goal internalization than the organizations that do not assess processes before improving them, and thus increase the chances to succeed. Based on empowerment as a result of clarity in the roles and responsibilities in the organization [12], we also look at the allocation of the roles and responsibilities in process improvement and its relation to the goal internalization items.

The research methodology employed in this study was that of a survey research. A survey strategy is of advantage when the research goal is to be predictive about certain outcomes. A survey is an appropriate strategy when the form of research question is "who", "what", "where", "how many" or "how much". As Yin states, the different research strategies are overlapping and case studies could also be used for this study, although they should preferably be used when the form of the research question is "why" [16].

3 Designing the Survey Questionnaire Based on Goal Internalization Items

Menon describes the construct of goal internalization in five measurable items [11]. These items capture the goal internalization effect leading to being inspired by the organizational goals or organizational vision. As a result of his study where he measured psychological empowerment, Menon concludes that goal internalization is highly correlated with affective organizational commitment and job involvement.

The measurable goal internalization items that Menon used in his study were the following:

1. I am inspired by what we are trying to achieve as an organization
2. I am inspired by the goals of the organization
3. I am enthusiastic about working toward the organization's objectives
4. I am keen on our doing well as an organization
5. I am enthusiastic about the contribution my work makes to the organization

In our study, we aim to discover if employees become inspired and enthusiastic about goal internalization after being involved in the decision making on what needs to be improved, i.e., in process assessment. We have two sets of questions targeting goal internalization in our survey: the first one addresses it after process improvement and the second one after process assessment itself. In our survey, we set the following measurable items for goal internalization regarding process assessment and improvement, and used a Likert scale from "Strongly Agree" to "Strongly Disagree", and additional options of "Not Applicable", and "I don't know".

After evaluating/assessing processes, I was enthusiastic about:

1. the organization's goals
2. working toward the organization's objectives
3. doing well as an organization
4. the contribution my work makes to the organization
5. aligning my work with the organization's goals
6. being able to decide myself how to improve my work

After improving processes, I was enthusiastic about:

1. the organization's goals
2. working toward the organization's objectives
3. doing well as an organization
4. the contribution my work makes to the organization
5. aligning my work with the organization's goals
6. improving my own work in my own way

Processes can be improved on different organizational levels with an organization-wide improvement initiative on one end, and improving one's own way of working by oneself on the other end of the scale. In both of the above-listed categories, the sixth goal internalization item shows the personal improvement approach where one determines by himself what is going to be improved and how, and can therefore be more loosely aligned to the organization's objectives.

4 Data Collection and Description of the Sample

An online survey was used to collect data from industry about the empowerment and goal internalization in process improvement. Since only the organizations interested and/or experienced in process improvement were targeted, non-probability sampling method was used in this research, more precisely the snowball technique. The request for distributing and responding to the survey was sent to companies, process improvement consultants, researchers, non-profit organizations promoting process improvement worldwide, and to various working groups in ISO/IEC SC7 (International Organization for Standardization subcommittee 7), that develop the international standards on software, systems, and services.

After two months, 50 completed responses were received. Out of the 50 responses, the distribution between software development and IT service providing organizations was almost equal forming more than 50% of all the responses.

Ten responses out of 50 came from organizations providing IT services, nine from software development organizations, and another ten from organizations both developing software and providing IT services. There was one response from an organization that did not categorize into any given business area (Fig. 1).

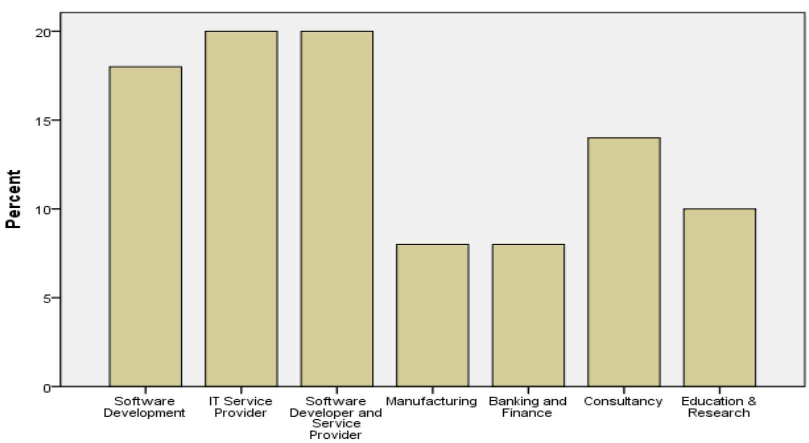


Fig. 1. Core business area of respondents' organizations

Over half of the responses came from large organizations employing more than 250 employees (62%), 18% from medium-sized organizations employing 50 to 249 employees and 10% from both small (with 9 to 50 employees) and micro (up to 10 employees) organizations.

The responses have the following geographical distribution: 68% of responses came from Europe with Finland being the most active respondent, 16% of responses came from USA, 6% from Canada, 4% from Mexico and Australia, and 2% from Peru. Unfortunately no responses were received from India.

4.1 Conducting Process Assessments

Process assessment allows revisiting organization's business goals, and involvement of all people related to the processes being improved. In order to know which respondents conducted process assessments, we sought information about different starting points of the process improvements and different ways of measuring process progress.

Table 1 illustrates the responses to the survey question about when process improvements were started. The respondents could select any relevant starting points listed in the table. Table 1, and the following tables in the paper, illustrate the highest values that are being described in the text with a "*", the number of survey respondents in the table captions as "n", and the table items in the main text are in *italic* for better readability. Table 1, below, shows that the most popular starting point for process improvements among the survey respondents was, in fact, *conducting a process assessment*. 58% of the respondents started their process improvements after they had conducted a process assessment.

Table 1. Starting point of process improvements (n=50)

At the end of the project or service delivery	After hearing about a better method or tool	After completing a training course, reading a professional textbook or attending a conference	After receiving critical customer survey results	After a process assessment	When management is planning an organizational strategic change	When customer(s) requests it
42%	48%	24%	46%	58%*	44%	34%

As Jones [17] points out, it is not wise to start process improvement if managers do not collect data to demonstrate the progress. There are various ways to measure the progress and Table 2 illustrates how process improvements were measured among the respondents' organizations.

In almost half of the cases, illustrated in Table 2, process improvements were measured based on *the customer and stakeholder satisfaction*, *evaluating the achievement of organization's goals*, and by *measuring personal performance and productivity*. Standard or process model based assessments indicate the strengths and weaknesses of current processes and suggest how to improve them. *Standard or model based process assessments*, or re-assessments if applied for measuring the progress of implemented improvements, were carried out only in 38% of cases. Despite process assessments being the most common starting point for process improvements, the progress is not as often measured by re-assessments. This could be explained as management's objective to see the benefits of process improvement faster without investing into another process assessment. Thus, there are different reasons for conducting process assessments: for the input to process improvement when conducted prior to it, and for the measurement of the progress of the already implemented improvements.

Table 2. Measuring process improvements (n=50)

Improvements are not measured	Measuring personal performance and/or productivity	Evaluating the achievement of product or service quality requirements	Evaluating the achievement of project or service performance objectives	Measuring project productivity	Conducting model/standard based process assessments	Evaluating stakeholder/customer satisfaction	Measuring organizational productivity	Evaluating the achievement of organizational goals	Calculating the return on investment to process improvement
8%	46%*	40%	34%	36 %	38%	50%*	26%	46%*	10%

In this paper, we address the input and setup of process improvement, and will therefore use only the responses where process assessments were conducted prior to the improvements in the data analyses.

4.2 Goal Internalization

Table 3, below, illustrates the responses to goal internalization of process assessment on the Likert scale. The respondents were most enthusiastic about *working towards the organization's objectives* after the process assessment (72%). As we predicted, the sixth item – *deciding how to improve my own work in my own way*, addressing

Table 3. Goal internalization in process assessment (n=50)

After evaluating processes/process assessment, I was enthusiastic about:	**Strongly agree**	**Agree**	**Disagree**	**Strongly disagree**	**N/A**	**Don't know**
the organization's objective	10%	40%	16%	0%	22%	12%
working towards the organization's objectives	14%*	58%*	8%	0%	12%	8%
doing well as an organization	22%	48%	10%	0%	12%	8%
the contribution my work makes to the organization	18%	46%	14%	0%	18%	4%
aligning my work with the organization's goals	18%	52%	10%	12%	14%	4%
being able to decide myself how to improve my work	20%*	42%*	12%	6%	14%	6%

process assessment on a personal level was the least inspiring after process assessment (62%). The 42% of respondents, who did not start their process improvements by conducting a process assessment, as indicated in Table 1, explains the high number of responses in the "N/A" and "Don't know" columns.

Table 4 below illustrates the responses to goal internalization of process improvement. Goal internalization after process improvements received slightly higher results than that after process assessment, and remained around 70%. Most of the respondents became enthusiastic *about the alignment of the organization's and process improvement goals* (78%), and *working towards the organization's objectives* (76%).

Table 4. Goal internalization in process improvement (n=50)

After process improvements, I was enthusiastic about:	Strongly agree	Agree	Disagree	Strongly disagree	N/A	Don't know
the organization's objective	24%	48%	18%	2%	4%	4%
working towards the organization's objectives	24%*	52%*	12%	2%	6%	4%
doing well as an organization	26%	54%	12%	4%	4%	0%
the contribution my work makes to the organization	24%	52%	10%	4%	8%	2%
aligning my work with the organization's goals	22%*	56%*	12%	0%	8%	2%
improving my own work in my own way	22%	42%	18%	4%	12%	2%

5 Data Analysis

This chapter illustrates the goal internalization results of the received responses – first the goal internalization after process assessment, and then the goal internalization after process improvement are described.

5.1 Goal Internalization after Evaluating Processes

One of the starting points for process improvement is process assessment that allows revisiting organization's business goals, involves all people related to the processes being assessed and improved. We asked the respondents about their goal internalization after process assessment on a Likert scale. Table 5, below, illustrates two sets of responses – the ones that conducted process assessment prior to process improvement and those who did not.

We can see that the employees who conducted process assessment were only 10.8% more enthusiastic about the goals of the organization than those who did not.

Similarly, the enthusiasm about the contribution that one's work makes to the organization was 9.9% higher among the respondents that conducted process assessment prior to process improvement (Table 6).

Table 5. Enthusiasm about organization's objectives after evaluating processes (n=50)

	After evaluating processes, I was enthusiastic about the organization's objectives					
Process improvements were started:	**Strongly agree**	**Agree**	**Disagree**	**Strongly disagree**	**N/A**	**Don't know**
after process assessment	4	12	4	0	6	3
without process assessment	1	8	4	0	5	3

Table 6. Enthusiasm about work contribution after evaluating processes (n=50)

	After evaluating processes, I was enthusiastic about the contribution my work makes to the organization					
Process improvements were started:	**Strongly agree**	**Agree**	**Disagree**	**Strongly disagree**	**N/A**	**Don't know**
after process assessment	5	14	3	0	6	1
without process assessment	4	9	4	0	3	1

Goal internalization is closely related to goal alignment, i.e., in aligning ones goals with the organization's business goals. Table 7 illustrates that respondents who carried out process assessments were 15.2% more enthusiastic in goal alignment than those who did not start improvements with process assessments.

Table 7. Enthusiasm about goal alignment after evaluating processes (n=50)

	After evaluating processes, I was enthusiastic about aligning my work with the organization's goals					
Process improvements were started:	**Strongly agree**	**Agree**	**Disagree**	**Strongly disagree**	**N/A**	**Don't know**
after process assessment	7	15	1	1	4	1
without process assessment	2	11	4	0	3	1

5.2 Goal Internalization after Process Improvement

Tables 3 and 4 indicated that the enthusiasm after process improvement was higher than that after process assessment. Table 8, below, illustrates the goal internalization after process improvement among the respondents who started their process improvement after conducting process assessment, illustrated in the row titled *After process assessment*. The Likert scale is applied to goal internalization items that are *in italic* in the table.

The biggest difference between the respondents who conducted process assessments and those who did not lies in becoming *enthusiastic about the organization's objectives* (the ones who conducted process assessment were twice as enthusiastic, frequency indicated with "*"), and *doing well as an organization* (the ones who conducted process assessment were 26% more enthusiastic).

Table 8. Goal internalization after process improvement

When did you start process improvements?	Strongly agree	Agree	Disagree	Strongly disagree	N/A	Don't know
	After improving processes I felt enthusiastic about the organization's objectives					
After process assessment	6*	18*	2	0	2	1
Other	6	6	7	1	0	1
	After improving processes I felt enthusiastic about working toward the organization's objectives					
After process assessment	8	14	3	0	3	1
Other	4	12	3	1	0	1
	After improving processes I felt enthusiastic about doing well as an organization					
After process assessment	8*	16*	2	1	2	0
Other	5	11	4	1	0	0
	After improving processes I felt enthusiastic about the contribution my work makes to the organization					
After process assessment	5	15	2	2	4	1
Other	7	11	3	0	0	0
	After improving processes I felt enthusiastic about aligning my work with the organization's goals					
After process assessment	7	16	3	0	3	0
Other	4	12	3	0	1	1
	After improving processes I felt enthusiastic about improving my own work in my own way					
After process assessment	5	11	7	2	4	0
Other	6	10	2	0	2	1

The following figure (Table 9) illustrates the enthusiasm after process improvement *about the organization's objectives* – the respondents who started their process improvements after process assessment were 32.3% more enthusiastic than those who did not.

Table 9. Enthusiasm about organization's objectives after process improvement (n=50)

	After improving processes, I felt enthusiastic about the organization's objectives					
Process improvements were started:	**Strongly agree**	**Agree**	**Disagree**	**Strongly disagree**	**N/A**	**Don't know**
after process assessment	6	18	2	0	2	1
without process assessment	6	6	7	1	0	1

To find out whether clarity in the roles and tasks is getting people more inspired after process improvement, we asked the respondents whether they allocate the roles and responsibilities before the process improvement begins. Table 10 illustrates slightly higher results of goal internalization after process improvement among the respondents who conducted process assessment prior to the improvement.

Respondents were twice more enthusiastic *about the organization's objectives*, and 20% more enthusiastic *about working toward the organization's objectives* when they had allocated process improvement roles and responsibilities.

Table 10. Allocating roles and goal internalization after process improvement

Before process improvement did you:	Strongly agree	Agree	Disagree	Strongly disagree	N/A	Don't know
	After improving processes I felt enthusiastic about the organization's objectives					
Allocate roles and responsibilities	8*	16*	4	0	1	1
Other	4	8	5	1	1	1
	After improving processes I felt enthusiastic about working toward the organization's objectives					
Allocate roles and responsibilities	6*	18*	4	0	1	1
Other	6	8	2	1	2	1
	After improving processes I felt enthusiastic about doing well as an organization					
Allocate roles and responsibilities	8	16	4	1	1	0
Other	5	11	2	1	1	0
	After improving processes I felt enthusiastic about the contribution my work makes to the organization					
Allocate roles and responsibilities	7	16	3	2	1	1
Other	5	10	2	0	3	0
	After improving processes I felt enthusiastic about aligning my work with the organization's goals					
Allocate roles and responsibilities	5	18	4	0	2	1
Other	6	10	2	0	2	0
	After improving processes I felt enthusiastic about improving my own work in my own way					
Allocate roles and responsibilities	4	15	6	2	2	1
Other	7	6	3	0	4	0

6 Conclusions

Goal internalization can affect and inspire employees to carry out the change in their organization. We assumed that process improvements that start with process assessments have higher goal internalization than those that do not, because process assessment can raise awareness of the organization's business goals, and can involve the stakeholders of process improvement. Realizing the possible impact of process assessment on goal internalization and motivation among employees, allows minimizing the resistance to organizational change once the improvements need to be implemented.

In this study, we presented the data analysis results of an international survey. Among the various ways to evaluate software development, our study indicated that process assessments increase awareness about organization's objectives best as it revisits the organization's business goals. Also, conducting process assessments led to a better alignment of organization's business goals and the goals of process improvement.

Based on Weiss et al. [18], process assessment is a criteria-oriented process where the software practices are measured against external criteria. A goal-oriented process, such as goal/question/metrics paradigm, assesses software development in the context of individual, project, and organizational goals. Weiss et al. argue that goal-orientation helps to reinforce the feeling within each organization that the recommended improvement will create benefit and eliminates much of the resistance to change that comes from long explanation and discussion of the external assessment criteria [18]. Our argument was that process assessment allows revisiting the organization's business goals and increasing the awareness about them. Although the data of the conducted survey supports this argument, additional studies comparing the impact of the goal-oriented and criteria-oriented processes on goal internalization should be conducted.

At the same time, we discovered that the overall goal internalization was higher after process improvement than after process assessment, which means that additional studies should be conducted into the evaluation of the process assessment process. More attention should be paid in process assessments to the awareness of business goals, to their alignment with the process improvement goals, and higher involvement of the employees. Since the results of this paper illustrate descriptive analysis, hypotheses tests will be conducted on the data in the near future. Additional studies will also be conducted to understand the impact of process assessment on motivation, and empowerment of employees contributing to an organizational change.

Acknowledgments. The present project is supported by the National Research Fund, Luxembourg and cofounded under the Marie Curie Actions of the European Commission (FP7-COFUND).

References

1. Arent, J.: Transforming Software Organizations with the Capability Maturity Model. In: Bomarius, F., Oivo, M. (eds.) PROFES 2000. LNCS, vol. 1840, pp. 103–114. Springer, Heidelberg (2000)
2. Lepasaar, M., Varkoi, T., Jaakkola, H.: Models and success factors of process change. In: Bomarius, F., Komi-Sirviö, S. (eds.) PROFES 2001. LNCS, vol. 2188, p. 68. Springer, Heidelberg (2001)
3. Paulk, M.C., Weber, C.V., Chrissis, M.B.: The Capability Maturity Model for Software. In: Elements of Software Process Assessment and Improvement, pp. 3–15. IEEE Computer Society Press, Los Alamitos (1999)
4. Varkoi, T.: Software Process Improvement Priorities in Small Enterprises. In: Information Technology. Licentiate of Technology. Tampere University of Technology, Pori (2000)

5. Humphrey, S.W.: Managing the Software Process, Software Engineering Institute (1989)
6. Grover, V.: From Business Reengineering to Business Process Change Management: A Longitudinal Study of Trends and Practices. IEEE Transactions on Engineering Management 46, 36–46 (1999)
7. Zahran, S.: Software Process Improvement - Practical Guidelines for Business Success. Addison-Wesley Professional, Reading (1998)
8. Morten Korsaa, M.B., Messnarz, R., Johansen, J., Vohwinkel, D., Nevalainen, R., Schweigert, T.: The SPI manifesto and the ECQA SPI manager certification scheme. Journal of Software Maintenance and Evolution: Research and Practice (2010)
9. Bo, Y.: Influence of Organizational Boundary on Psychological Empowerment in Multi-organization Network. In: Maoshan, Q. (ed.), vol. 5, pp. 195–198 (2009)
10. Spreitzer, G.M.: Toward a common ground in defining empowerment. Research in Organizational Change and Development 10, 31–62 (1997)
11. Menon, S.T.: Psychological Empowerment: Definition, Measurement and Validation. Canadian Journal of Behavioral Science 31(3), 161–164 (1999)
12. Baddoo, N.a.H.: Tracy: Motivators of Software Process Improvement: an Analysis of Practitioners' Views. Journal of Systems and Software 62, 85–96 (2002)
13. Spreitzer, G.M.: Social Structural Characteristics of Psychological Empowerment. Academy of Management Journal 39, 11 (1996)
14. Tracy Hall, H.S., Beecham, S., Baddoo, N., Robinson, H.: What Do We Know about Developer Motivation? IEEE Software, 92–94 (2008)
15. Barafort, B., Di Renzo, B., Merlan, O.: Benefits resulting from the combined use of ISO/IEC 15504 with the Information Technology Infrastructure Library (ITIL). In: Product Focused Software Process Improvement (SPI 2002). Springer, Berlin (2002)
16. Yin, R.K.: Case Study Research: Design and Methods. Sage, Thousand Oaks (2002)
17. Jones, C.: The economics of software process improvement. Computer 29(1), 95–97 (1996)
18. Weiss D.M., Bennett, D., Payseur, J.Y., Tendick, P., Zhang, P.: Goal-Oriented Software Assessment. In: International Conference on Software Engineering, pp. 221–231 (2002)

Improvement of Innovation Management through the Enlargement of Idea Sources

Martin Neumann[1], Andreas Riel[2], and Daniel Brissaud[2]

[1] Kolbenschmidt Pierburg AG,
41460 Neuss, Germany
martin.neumann@de.kspg.com
[2] Grenoble-INP, G-SCOP UMR5272,
Grenoble, F-38031, France
{andreas.riel,daniel.brissaud}@grenoble-inp.fr

Abstract. The capacity of innovation and creative performance have become a major stake for the success of companies. Today companies act in an environment which is more competitive and complex than ever before. Thus, to succeed in the global market, it is necessary to combine customer needs, productivity and competitiveness in the development of new product innovations. To this aim, innovation management has to leverage idea generation and capitalization. This paper investigates the importance of idea generation and idea sources in innovation management. A case study which links these theoretical principles with industry experiences shows that the enlargement of idea sources can boost innovation power.

Keywords: Idea Generation, Idea Sources, Innovation Management, Automotive Supplier Industry.

1 Introduction

The success of the product development process heavily depends on the input to the innovation management system that collects, examines, evaluates and selects new concepts and ideas. Stevens and Burley [1] have shown in their study that it takes 3,000 raw ideas to identify approximately 300 novel ideas out of which only nine are commercially significant. Finally only one single idea achieves a significant business success. This poor success rate proves that “It seems we need ideas, and we need lots of them” [2]. Thus idea generation processes have great importance. The main purpose of all idea generation activities is to ensure that the company does not leave the exploration phase of new product development to chance [3].

Changes in the idea generation process may also increase the quality of the produced ideas. Several researchers state that ideas developed from a deep understanding of the customer usually have higher value and better chances of success [4]. Companies have to be aware that idea generation does not happen

1. informally and without specific purpose [5],
2. sporadically [6], and neither
3. as a merely in-house method.

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 121–132, 2011.

In fact, all members of the innovation value chain should participate in a systematically and continuously organized idea generation process to guarantee sustainable innovation results and business success.

The main focus of this paper is on idea generation and idea sources as a further step towards the continuous improvement of an existing innovation management system [7]. Section 2 of this paper is devoted to theoretical considerations of the term idea generation and the importance of knowledge for innovations. Section 3 deals with the stakeholder concept to increase idea sources and to improve innovation management. Section 4 concludes the paper with a final evaluation.

2 Theoretical Considerations on Idea Generation Processes

2.1 Idea Generation – A Major Task within Innovation Management

In literature and often in practice, the innovation process is considered as a multiphase linear and/or iterative process. No consensus exists about the number and the definition of the individual phases [8][9][10]. A simple pattern was developed by Thom [10]. He divides the product development process in the phases of idea generation, idea acceptance and idea realization. These main stages are further divided into individual sub-phases and/or subtasks. The advantages of this generic model are on one hand its adaptability on all types of innovation and the other hand the explicit inclusion of a decision phase in the innovation process [11].

Rothwell and Zegveld developed an interactive model including both technology-push and market-pull models [12], in which “innovation is regarded as a logically sequential, though not necessarily continuous, process, that can be subdivided into a series of functionally separate but interacting and interdependent stages” [13] and can be thought of as a complex set of communication paths over which knowledge is transferred between the marketplace, the science base and the organization’s capabilities [12]. According to this definition, the most fundamental resource for innovation is knowledge and, accordingly, the most important process is learning.

Product innovations are mainly successful if they are systematically prepared, realized and implemented and they do not happen as a result of pure chance [14][15]. For that purpose it is necessary to create appropriate basic conditions for the innovation activities and to plan, manage and control individual innovation projects in coordination with other innovation activities [11]. These tasks are summarized under the term innovation management. Within the innovation management, idea management can be defined as a sub-process responsible for the generation of internal and external ideas, the structured evaluation of ideas and finally the selection of the most promising ideas for future business success [16].

According to these assumptions, the term fuzzy front end is essential because it is very similar to the above described process phases and explains the earliest stages of new product development, even before its first official discussions [17]. This early stage of the innovation process includes all the time spent on the idea as well as the activities enforcing it; from the first impulse and/or opportunity for a new product or a new service up to go/no go decisions concerning implementation and the start of development of the new product and/or service [18][19]. Apparently the fuzzy front end is comparable to idea generation but the front end primarily concentrates on opportunity identification and analysis prior to actual idea management [20][21][22]. However, the effective management of the early phase of the innovation process is the origin for innovative ideas for sustainable competitive advantage [23]. This influence of the front end on new product development has been verified by empirical studies [19][24][25][26]. Table 1 summarizes the main results of these studies.

Table 1. Studies confirming the impact of the front end on new product development

Object of investigation	Results	Source
144 German measurement and control firms	Companies which reduce systematically market and technological uncertainties during the fuzzy front end of innovation belong to the more successful innovators	[24]
497 New Product Development (NPD) projects from Japanese mechanical and electrical engineering firms	Key driver of project success is the intensity of planning prior to the start of development: relationship between front end factors and project success	[25]
475 Research and Development projects in Japanese electrical and mechanical engineering companies	Planning intensity during the early phase of innovation is linked to the project success	[26]
Conclusions from the studies: high importance of • early reduction of technical and market uncertainty • early involvement of all relevant project members • early interdisciplinary teamwork and communication • early involvement of top management and allocation of resources		

Because idea generation is critical to the development of new products [27], innovation management has to look systematically for new idea sources to enlarge the pool of ideas and guarantee long-term market success.

2.2 Knowledge and Learning as the Main Levers of Idea Generation

Apart from internal factors of a company, innovation strategies have to take into account external factors as well [28]. These external factors are [29]:

1. *Remote factors* are beyond the borders of a company like economic, regulatory, social, political and ecological variables.
2. *Industry factors* influence the company but the latter has only limited control on these variables, e.g. the competition and the supply chain.
3. *Company factors* are operational forces of a business which can be most influenced by a company such as customers, suppliers, competitive position and creditors.

Thus, idea generation occurs through interactions inside or outside an industrial firm and the sources can be individuals or groups [30]. Due to these comprehensive and profound interactions within the corporate divisions and/or the business environment the innovation management as the responsible managing link between idea generation and the whole innovation process represents a company-wide function with influence on the leadership of the whole corporation [31][32]. In their review of several studies on the success and failure of new product development Martinez-Sanchez et al. identified that the use of multifunctional teams and the adoption of inter-department responsibilities are positively related to the new product performance, including development and marketing time [33]. Therefore the central purpose of the innovation management is to ensure information flow (e.g. by organizational measures), and to initiate and continuously guarantee information and knowledge exchange [11].

Many authors articulate the vital role that knowledge and learning play in innovation activities, underlining the importance of processes and mechanisms for collecting information and creating knowledge from both internal and external sources [34]. In operational effectiveness, the main aspect involves organizational learning activities that bring understanding of action outcomes, causal connections and result in higher-order learning [35]. It is also important to consider aspects in the knowledge creation process: the organization's internal knowledge base, the acquisition of information and knowledge from external sources, the integration of internal and external knowledge and its application to problem solving, the creation of new knowledge and the generation of innovations from this integration, and finally the importance of the organization's capacity to absorb new knowledge [36]. This process of knowledge creation is depicted in Figure 1, according to Soo et al [37].

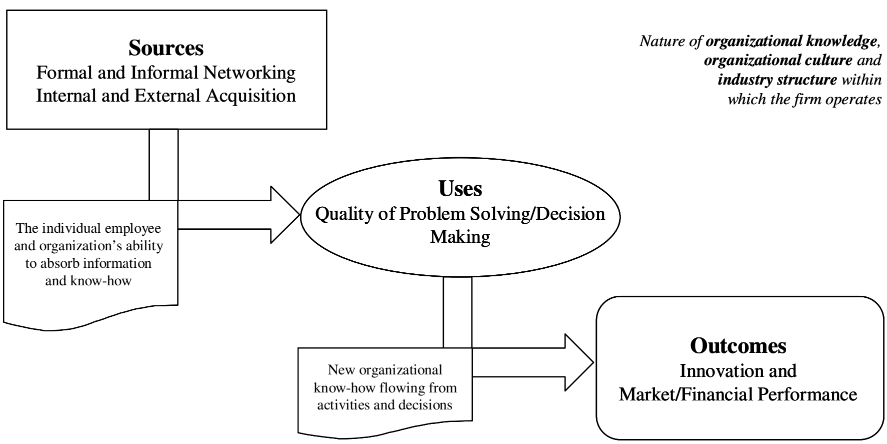


Fig. 1. The process of knowledge creation and innovation [37]

Scanning the environment, networks and alliances for alternatives and observing competitors also leads to potential alternative practices and ideas. Most firms are engaged in these activities simultaneously because they manage several concurrent projects at different stages in the product development process [38].

3 Stakeholders in the Idea Generation Process

The following section is based on a case study of an automotive supplier company running a strategic project to improve its existing innovation management [7] through the enlargement of idea sources and improvement of its idea management process taking into account the key findings pointed out in section 2.

3.1 The Stakeholder Model

The model of market stakeholders [39][40] can serve as a basis for the enhancement of idea generation by the enlargement of idea sources. The basic idea is that not only one group of idea contributors should be responsible for innovations, but also other stakeholders of the corporate environment should be actively involved in the idea generation process. This approach is depicted in Figure 2.

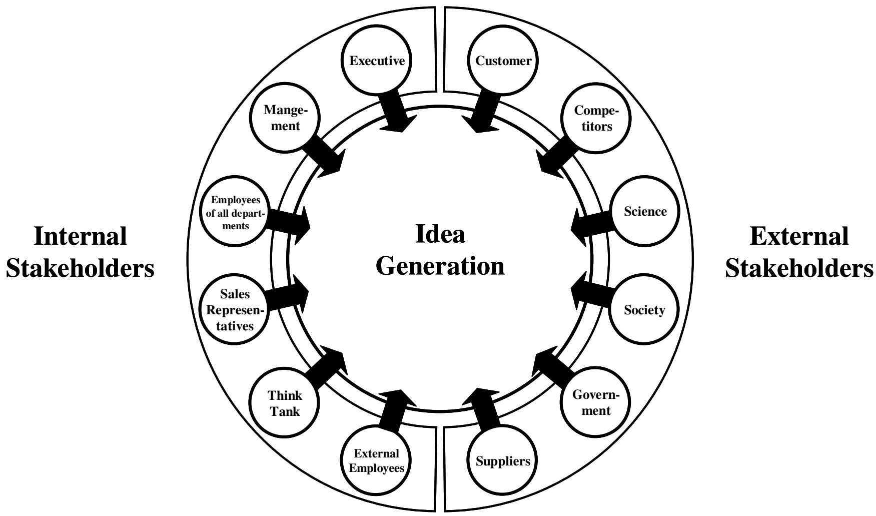


Fig. 2. Idea generation based on the stakeholder concept

To capture new ideas from different sources, it is essential to identify potential sources inside and outside the organization. Specific methods to access, to extract and to use their knowledge and their ideas have to be found.

There is a very important consideration in the involvement of a large group of different stakeholders in the idea generation process. As was shown in research on integrated product and system design [41], integrating stakeholders of the complete product/system life cycle throughout the entire product/system development process from the earliest phases on, is the key to creating sustainable innovation. The sustainability aspect is leveraged by the fact that only the integration of different views on the product/system in terms of its functions and its economic, ecologic, and social environment allows to identify requirements and constraints on the product/system in a holistic manner, and therefore to take them into account both in the composition of development teams, as well as in the design and architecture of the product/system. The same issue applies to idea generation and assessment, which is part of the earliest upfront phases in the product/system life cycle. Figure 2 distinguishes among stakeholders which are internal to a particular organization, and those which are external to it. There is no unique grouping of related stakeholders, however concepts from social science help in clustering stakeholders as it has been done in the model represented in Figure 2. In integrated design, [42] proposes groups ("worlds") of stakeholders which share

1. *Logic of Action:*
   Stakeholders expose and contribute what is essential for them.
2. *Scale of Value:*
   Means to measure and understanding of the value contribution.
3. *Collective Knowledge:*
   Knowledge that is shared among different worlds.

The model in Figure 2 has done exactly that for selected groups from the point of view of innovation management with respect to idea generation capacity.

3.2 Internal Sources of Ideas

The definition of internal stakeholders in Figure 2 is the basis for innovation management to whom in-house idea generation activities have to be addressed. In particular, employees are highly cited as sources of ideas [3][20][30]. Typically, only executives and R&D employees submit ideas. An organizational framework to manage idea generation involving all internal stakeholders is essential. Table 2 summarizes the direct and indirect ways to generate ideas from these internal sources.

To collect internal ideas, several companies have put in place idea databases, which are accessible to employees, inciting them to suggest ideas to the innovation management. Experience shows although the tool is available to all employees in all departments (like R&D, Sales, Purchasing, etc.) including management and executives, input from departments other than R&D and from employees in leading positions outside the R&D department is very low.

Another weakness of idea databases is that they mainly support the collection, evaluation and selection of inventions rather than innovations. While invention describes the first technical realization of a new problem solution developed as a result of research activities and leads to a legal basis for utilization of the results (for example in the form of patents), the term innovation implies also the utilization, integration and marketing of new solutions in usable products and services, going beyond the actual invention [11][15][43][44].

R&D is the basis for the development of innovations. It covers a set of specific processes that are created to gain knowledge and to discover new technical solutions to a problem [11][15][44]. In a technology-driven business environment Intellectual property plays a major role because it fulfills three main functions [45]:

1. protection of price and market share by excluding others from a specific marketplace;
2. insurance against legal action by other patent holders to mitigate risk of infringement and
3. financial asset in strategic alliances, in which technology is licensed, swapped, assigned, mortgaged, or held as a blocking strategy.

Table 2. Overview of internal idea sources

Stake-holder	Idea Sources	Direct ways to get ideas	Indirect way to get ideas
Executive	Executive in the company	• Idea generation activities and processes • Idea and/or innovation database • Idea management or corporate improvement suggestion system • Continuous improvement process • Inventions	• General overview of external stakeholders interests
Management	Management professional in the company	• Same as first source	• Same as first source
Employees of all Departments	Departments includes R&D, sales, purchasing, customer service, quality control, production, controlling etc.	• Same as first source	• Especially sales and customer service should capture customer ideas • Purchasing should collect supplier ideas
Sales Representatives	Sales representatives promote and sell products directly to customer in the field; may have their office at the customer's site	• Contractual agreements • Direct solicitation	• Organization of workshops or seminars at the customer's site, for example R&D can present and discuss new product solutions directly with the customer on-site
Think Tank	A group dedicated to coming up with new ideas, research and knowledge	• Outcome based innovations • Inventor circles	• This group can have members from all departments and so different aspects can be considered
External Employees	Collective term for loosely affiliated employees, like project-based employees, temporary employees, freelancers or students	• Same as first source	• Stimulus from outside • Possible solution to avoid to be professionally blinkered

Regarding the degree of novelty of the collected ideas in the database, the majority of the ideas is related to existing applications, so called incremental innovations [15], and describes useful modification of existing products for the day-to-day business.

Only a few ideas imply radical innovations [15], which are the essence of long-term sustainable innovation of an enterprise.

Providing the right tools to support idea generation and collection tool is important. However, innovation management also has to create an environment for the promotion of innovation. In general innovation management has to motivate all employees to take part in the innovation activities, and in particular give the impulses for ideas by confronting inventor circles or especially formed integrated teams of experts from different trades with new market trends to stimulate their idea generation. Information about such trends could be made available in an innovation database by using knowledge mining techniques [46] or discovered through external sources. For example, the regulatory environment may provide opportunities for customized products to meet regulatory requirements in areas such as pollution and emission control, a topic that influences automotive suppliers significantly.

3.3 External Sources of Ideas

As shown in Figure 2, the analysis of external sources of ideas will concentrate on following six main sources: customers, competitors, science, society, government and suppliers. Within a company a lot of activities and techniques exist which are directly connected with idea sources and the generation of product ideas. Other actions are indirect idea sources and influence only indirectly the generation of product ideas. These information sources which up to now help mainly management and business development have to be analyzed for how they can be also used for a successful product idea generation. Table 3 shows the major existing external idea sources.

Usually a company has access to a lot of possible external idea sources like in table 3, which could be capitalized on. Some typical problems with the analyses of these external sources are:

- information of these external idea sources is widely spread within the company,
- no central storage of this knowledge exists and
- there is no systematic knowledge management implemented so far.

So the collection of information must be carried out individually, and it is necessary to know the right contact person within the company for the collection of specific information from the external idea sources.

To achieve sustainable innovation success, it is important to obtain internal acceptance for the usage of external idea sources. One possible way is to minimize the additional individual work effort for the internal contact person to get the information from outside and share this knowledge with other colleagues. When this approach is applicable it makes sense to widen the sources of ideas within the specific categories. The exploration of external idea sources is thus an internal step-by-step process.

For example, one good step in the direction of a better collection of customer ideas is the creation of permanent customer-related teams with team members from all customer-related departments. The main tasks of these teams are:

1. build-up knowledge about customers and share these customer insights with team members;

Table 3. Overview of external idea sources

Stakeholder	Idea Sources	Direct ways to get ideas	Indirect ways to get ideas
Customers	• Core customer groups	• Customer submitted ideas • Interviews • Customer contracts negotiations	• Customer analysis • Satisfaction surveys • Customer database • Internal customer-related teams
Competitors	• Direct competitors	• Competitive Intelligence • Direct talks during international fairs and summits	• Market research firms
Science	• Universities	• Sponsoring of university chairs • Master thesis projects • Networking	• Scanning new technology releases, like PhD thesis or other publications
Society	• Groups of interests like industry associations • Media sources	• Working groups • Contact with editors	• Publications from associations • Scanning media, especially internet research or patent research
Government	• Governmental departments	• Visiting respective website • Scanning new technology regulations	• Attend in national and international fund programs for innovation projects • Scanning commentaries concerning new laws
Suppliers	1. Suppliers of physical goods like Tier 1 and/or 2 supplier, etc.	• Supplier submitted ideas • Meetings • Contract negotiations	• Supplier analysis • Research for news from suppliers
	2. Suppliers of information, like consultants and research firms	• Direct talks • Visiting presentations • Networking	• Working with database of consultants • Use of provided information services

2. development of a homogenous and consistent understanding of the customer future production, which represents the company's level of information and which is binding for all business divisions;
3. discussion of the customer-related topics and estimation of a final result, which represents the company's common market view.

Within these teams the members have the possibility to share their knowledge with colleagues and make their market estimation transparent. For the management of these customer teams an IT-based technical solution can be created and continuously improved to collect and to store systematically the data of the customer team members.

4 Conclusion

This paper points out that idea generation has to be understood as part of the whole innovation process. To support new product development, idea generation must be paced in the larger context of the whole innovation process and it has to be defined as a systematic sub-process of innovation management on a regular basis. To accomplish this goal, idea generation activities should not only be based on well-known internal sources but also have to be expanded to external sources. It is a major mistake to think ideas can only come from inside the company. This error has been termed the not-invented-here (NIH) syndrome where companies reject ideas generated outside its walls because they think those ideas are inferior to their own [47].

Knowledge plays an important role in this context. It is essential to build up a fundamental understanding in the company for the utilization of these new idea sources and the accompanying advantages of this new concept of innovation management. This may help to avoid internal restrictions. Moreover, it can be said that this challenge of openness may provide an exceptional stimulus on innovation and sustainable success.

References

1. Stevens, G.A., Burley, J.: 3,000 raw ideas = 1 commercial success? Research Technology Management 40(3), 16–27 (1997)
2. Dandy, G., Walker, D., Trevor, D., Warner, R.: Planning and Design of Engineering Systems, 2nd edn., p. 99. Taylor & Francis, Abingdon (2008)
3. Stasch, S.F., Lonsdale, R.T., LaVenka, N.M.: Developing a Framework for Sources of New Product Ideas. Journal of Consumer Marketing 9(2), 5–15 (1992)
4. Flint, D.J.: Compressing new product success-to-success cycle time deep customer value understanding and idea generation. Industrial Marketing Management 31(4), 305–315 (2002)
5. Adams-Bigelow, M.: First Results from the 2003 Comparative Performance Assessment Study (CPAS). In: Kahn, K.B., Castellion, G., Griffin, A. (eds.) The PDMA Handbook of New Product Development, 2nd edn., ch. 36, pp. 546–566. John Wiley & Sons, Inc., Hoboken (2005)
6. Tucker, R.B.: Driving growth through innovation: How leading firms are transforming their futures, 2nd edn. Berrett-Koehler Publisher, Inc., San Francisco (2002)
7. Neumann, M., Riel, A., Brissaud, D.: Innovation Management System for the Automotive Supplier Industry to Drive Idea Generation and Product Innovation. In: Proceedings of the 17th EuroSPI Conference, vol. 17, pp. 7.13–7.24 (2010)
8. Hauschield, J.: Innovationsmanagement, 2nd edn., Vahlen, München (1997)
9. Kleinschmidt, E., Geschka, H., Cooper, R.G.: Erfolgsfaktor Markt. Kundenorientierte Produktinnovation. Springer, Heidelberg (1996)
10. Thom, N.: Grundlagen des betreiblichen Innovationsmanagements, 2nd edn. Peter Hanstein Verlag, Königstein (1980)

11. Stockmeyer, B.: Ansatzpunkte und Methoden zur Effizienzsteigerung im Innovationsmanagement der Ernährungsindustrie. PhD Thesis, Technische Universität München, Germany (2001)
12. Trott, P.: Innovation Management and New Product Development, 4th edn. Financial Times Prentice Hall (2008)
13. Rothwell, R., Zegveld, W.: Reindustralization and Technology, p. 50. Longman, Harlow (1985)
14. Griffin, A.: PDMA research on new product development practices. Journal of Product Innovation Management (JPIM) 14(6), 429–458 (1997)
15. Pleschka, F., Sabisch, H.: Innovationsmanagement. Schäffer-Poeschel, Stuttgart (1996)
16. Brem, A.: The Boundaries of Innovation and Entrepreneurship – Conceptual Background and Essays on Selected Theroretical and Empirical Aspects. Gabler-Verlag, Wiesbaden (2008)
17. Brem, A., Voigt, K.-I.: Innovation management in emerging technology ventures – the concept of an integrated idea management. Int. J. Technology, Policy and Management 7(3), 304–321 (2007)
18. Reid, S.E., de Bretani, U.: The Fuzzy Front End of New Product Development for Dicontinous Innovations: A Theoretical Model. Journal of Product Innovation Management 21(3), 170–184 (2004)
19. Herstatt, C., Verworn, B. (eds.): Management der frühen Innovationsphasen: Grundlagen – Methoden – Neue Ansätze, 2nd edn. Gabler-Verlag, Wiesbaden (2007)
20. Belliveau, P., Griffin, A., Somermeyer, S.M. (eds.): The PDMA ToolBook 2 for New Product Development. John Wiley & Sons, Inc., Hoboken (2004)
21. Khurana, A., Rosenthal, S.R.: Integrating the Fuzzy Fron End of New Product Development. Sloan Management Review 38(2), 103–120 (1997)
22. Khurana, A., Rosenthal, S.R.: Towards Holistic "Front Ends" in New Product Development. Journal of Product Innovation Management 15(2), 57–75 (1998)
23. Kim, J., Wilemon, D.: Focusing the fuzzy front-end in new product development. R&D Management 32(4), 269–279 (2002)
24. Verworn, B.: How German measurement and control firms integrate market and technological knowledge into the front end of new product development. International Journal of Technology Management 34(3-4), 379–389 (2006)
25. Verworn, B., Herstatt, C., Nagahira, A.: The fuzzy front end of Japanese new product development projects: impact on success and differences between incremental and radical projects. R&D Management 38(1), 1–19 (2008)
26. Stockstrom, C., Herstatt, C.: Planning and Uncertainty in New Product Development. R&D Management 38(5), 480–490 (2008)
27. Troy, L.C., Szymanski, D.M., Varadarajan, P.R.: Generating New Product Ideas: An Initial Investigation of the Role of Market Information and Organizational Characteristics. Journal of the Academy of Marketing Science 29(1), 89–101 (2001)
28. Connell, J., Edgar, G.C., Olex, B., Scholl, R., Shulman, T., Tietjen, R.: Troubling successes and good failures: Successful new product development requires five critical factors. Engineering Management Journal 13(4), 35–39 (2001)
29. Pearce, J.A., Robinson, R.B.: Strategic Management: Formulation, Implementation, and Control, 12th edn. McGraw-Hill/irwin, New York (2011)
30. Alam, I.: Commercial innovations from consulting engineering firms: An empirical exploration of a novel source of new product ideas. Journal of Product Innovation Management 20(4), 300–313 (2003)
31. Diller, H. (ed.): Vahlens Großes Marketing Lexikon. DTV, München (1994)

32. Pleschak, F., Sabisch, H.: Innovationsmanagement. Schäffer-Poeschel, Stuttgart (1996)
33. Martínez-Sánchez, A., Pérez-Pérez, M.: de-Luis-Carnicer, P., Vela-Jiménez, M. J.: Teleworking and new product development. European Journal of Innovation Management 9(2), 202–214 (2006)
34. Ayuso, S., Rodríguez, M.A., Ricart, J.E.: Using stakeholder dialogue as a source for new ideas: A dynamic capability underlying sustainable innovation. Working Paper No. 633, IESE Business School, University of Navarra, Spain (May 2006)
35. Argyris, C., Schön, D.A.: Organizational Learning II: Theory, Method, and Practice. Addison Wesley, Reading (1996)
36. Soo, C.W., Midgley, D.F., Devinney, T.M.: The process of knowledge creation in organisations. Working Paper 2002/30/MKT, INSEAD, Fontaineblaeu, France (February 2002)
37. Devinney, T.M., Midgley, D.F., Soo, C.W.: Knowledge Creation in Organizations: A Muliple Study Overview. In: Davis, J., Subrahmanian, E., Westerberg, A. (eds.) Knowledge Management – Organizational and Technological Dimensions, pp. 77–96. Physica-Verlag, Heidelberg (2005)
38. Rothaermel, F.T., Deeds, D.L.: Exploration and Exploitation Alliances in Biotechnology: A System Of New Product Development. Strategic Management Journal 25(3), 201–221 (2004)
39. Freeman, R.E.: Strategic Management: A Stakeholder Approach. Pitman, Boston (1984)
40. Freeman, R. E.: The Stakeholder Approach Revisited. In: Zeitschrift für Wirtschafts- und Unternehmensethik (zfwu), Jahrgang 5, Heft 3, pp. 228–241 (2004)
41. Riel, A., Tichkiewitch, S., Messnarz, R.: Integrated Engineering Skills for improving the System Competence Level. Software Process Improvement and Practice (SPIP) 14, 325–335 (2009)
42. Mer, S., Jeantet, A., Tichkiewitch, S.: La génie industriel dans un monde sans frontières. In: 2ème congres international franco-québécois, Albi, France, September 3-5 (1997)
43. Nagel, R.: Lead User Inovationen. DUV, Wiesbaden (1993)
44. Specht, G., Beckmann, C.: F&E-Management. Schäffer-Poeschel, Stuttgart (1996)
45. Simmer, R.: Using Intellectual Property Data for Competitive Intelligence. Licensing Executive Society USA & Canada Inc., Burnaby, Canada (2001), `http://patex.ca/pdf/publications/CH4-CompIntelRevB.pdf`
46. Asakiewicz, C.: Knowledge Mining – The Quantitative Synthesis and Visualization of Research Results and Findings. VDM Verlag Dr. Müller, Saarbrücken (2008)
47. Katz, R., Allen, T.J.: Investigating the Not Invented Here (NIH) syndrome: A look at the performance, tenure, and communication patterns of 50 R&D Project Groups. R&D Management 12(1), 7–12 (1982)

The Usability Approach in Software Process Improvement

Péter Balázs Polgár[1] and Miklós Biró[2]

[1] Eötvös Loránd University, Budapest, Hungary
sirpepe@elte.hu
[2] Dennis Gabor College, Budapest, Hungary
biro@gdf.hu

Abstract. In the past years, efforts in the field of Software Process Improvement were increasingly focusing on human aspects realizing that people participating in the processes have a high impact on the success of any improvement. While these aspects are included in newer models and methodologies, it is still vague how Software Process Improvement can be made suitable for different people. In this paper, we propose a new approach: the application of usability methodology for Software Process Improvement. We present an overview of the people aspects in Software Process Improvement and introduce a methodology to apply usability methodology in this context. We also formulate ideas on how to adapt concrete improvement methods based on the usability methodology.

Keywords: Usability, Software Process Improvement, SPI.

1 Introduction

Recently, more and more Software Process Improvement (SPI) research studies the impact of people aspects on SPI projects [1][8][9][12][13][17][19][20]. This impact stems from several factors. People taking ownership of the processes care more for the results and the proper execution, they are also more empowered for improvement and innovation, resulting in better processes and better products [2][3][4].

While SPI has ever greater emphasis on people issues, another discipline, usability is becoming more important as computers become ubiquitous. The usability methodology is about designing software and systems based on human needs, and as we are increasingly surrounded by computers, the ease of use of these devices becomes a major factor. Usability as a discipline has a history of helping to produce software, and more recently systems which are suitable for users, thus resolving many people related problems other engineering fields are not suitable to handle. The usability methodology builds on a wide range of methods based on psychology and ergonomics principles helping practitioners to design systems which support users in their tasks. The User Centered Design [26], forming the core value of the usability methodology, enables to view all development projects, including SPI projects with a fresh eye focusing on the humans involved in the systems.

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 133–142, 2011.

Our recognition of the potential of usability methods in handling the growing concern about people issues in SPI models lead us to the investigation of the approach described in this paper.

The remainder of this paper is structured as follows. The second section describes related work in the field of SPI focusing on people issues. The third section introduces the usability discipline discussing some of the basic concepts. We present the approach of applying the usability methodology to SPI in the fourth section, and also show under what conditions this is feasible. In the fifth section, we discuss the usage of five usability methods in the SPI context. Finally, section six contains some conclusions and plans for future work.

2 People Issues in SPI

Processes are considered the cornerstone for many organizations as the most effective way of producing quality products. Organizations also realize the need to improve these processes to become more successful in their business, to be more competitive, to make products of higher quality and cheaper than their competitors. In the end processes are still carried out by people, so the effective process completion relies on the abilities, skills and motivation of individuals. While employing excellent team members certainly helps, personalities of people can still make or break a process, influencing the end product. This inspires process improvement professionals to handle people issues.

The importance of people issues was realized gradually by practitioners. [1] describes how the focus got on people instead of the processes from the early days of process improvement:

- The traditional management model lasted till the end of the 1980s. It featured set rules for every task, with clearly specified roles. The hierarchical control was firm, flowing from the top of the organization. Because people were far from the origins of the processes and were not involved with their creation, processes were not well understood and the implementation was mechanical without consideration of the results on the applicators level.
- The Human Behavior Model started in the 1990s recognizing the importance of involving individuals in the process creation and improvement. This way a selected few were closely connected to the processes, making those empowered but leaving others unaffected. If these people left the organization, the empowerment left with them.
- The systems based people approach emerged at the end of 1990s. Networking was considered the core part of improvement and innovation. The selected empowered individuals were connected to a network of supporters from different parts of the organization.
- The learning organization concept starting form 2003 [2] was built upon a shared vision everyone in the organization agrees upon. The interaction provided with this shared vision created an empowering atmosphere, thus enabling more effective SPI through the organizations strategy.

A study about organizational learning [3] showed evidence to support this shift of focus to people, stating that 58% of the success factors for the implementation of innovation and improvement are influenced by human and organizational aspects.

Recent models also address people issues as an important factor in improvement:

– The ImprovAbility Model [4]: people aspects appear in most of the 20 parameters.
– Process and Enterprise Maturity Model [5]: people issues appear on most organizational and process maturity levels.
– The team centered processes from [6]: by looking at processes from a performer's perspective concludes that process needs to enable responses to situations.

Most recently, the SPI Manifesto [7] stated the principle: "We truly believe that SPI must involve people actively and affect their daily activities".

This reinforces the focus on human aspects shifting from expert designers to the process applicators in defining and improving the processes. This principle is also supported by a number of values in the SPI Manifesto:

– "Know the culture and focus on needs": for the SPI to work, the organizational culture should be studied, as the people making up the organization carry values and practices. The SPI must consider these values to succeed.
– "Motivate all people involved": motivated people are more eager to participate in innovation and improvement, striving to look for solutions in their work.
– "Base improvement on experience and measurements": the SPI efforts must be based on the actual practices done by the organization, and all improvement activities should be based on quantifiable data.
– "Create a learning organization": the main benefit of this value is the culture supporting the continuous improvement.

Based on individuals, culture can have a strong effect on every improvement effort. It is important to take into account even on the organizational level, but on the national level it has a high influence on both the development process and the improvement. This concerns especially international organization composed of individuals of many nationalities as presented in [8][9].

Culture is part of the environment context and can have a significant effect on team members' work. People belong to more than one cultural group which results in different thinking [10].

The most basic and usually the most visible display of culture are artifacts [11]. They represent physical or mental constructions of the organization. Most of the time process improvement relies on artifacts (for example documents) to carry out tasks and gather feedback.

Processes are represented by artifacts, namely the process descriptions. The ease of use or more specifically the usability of process descriptions was investigated in [12]. They found that there are usability related factors (for example understandable, tailorable, reusable, etc.), but their impact was not determined. Some of these factors were also proposed in [13]. Other studies showed that process descriptions have many usability problems [14], [15] and [16] impacting the application of the processes in a negative way.

A method to analyze the human aspects of software development was presented in [17]. The QUASE model described in this paper was built to identify shortcomings of the project environment from the human perspective through the quantitative measurements of perceived distance.

The agile methodology [18], created and propagated by software developers, puts the emphasis on the individuals instead of the processes: "we have come to value: Individuals and interactions over processes and tools". This human point of view for the agile methodology was investigated in [19]. It was shown that context factors, most importantly culture is in important aspect for the group and the shared values. They also mention the ETHICS [20] framework as similar to the agile methodology with one of the success factors being the human involvement. The importance of the team members was also shown in [21], stating that to use the agile methodology, you need agile people.

In summary, the human aspect is now a recognized factor in Software Process Improvement, however a coherent methodology to build upon this factor seems to be lacking.

3 Usability

Usability is part of the software engineering quality model [22], often the most important attribute of a product from the user's point of view. It also belongs to the broader field of Human-Computer Interaction studying how the humans use systems with software. Usability commonly refers to two things, first it is a quality component of a product, and secondly it is discipline of creating such products. We will use the term usability engineering to refer to the act of creating and reserving the term usability to the quality component.

To form a basis for our further discussions, we will use the definition from [23]: "3.1 Usability: Extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use." This definition implies that we cannot produce a system that provides the same results under different contexts, with different users and different goals. Also the definitions state that with usability we are not just striving to get things done (effectiveness), but we need to do it with as little effort and resources consumed as possible (efficiency), while providing the users with a positive feeling and motivation (satisfaction).

Accurately describing the three product usage aspects (context, user and task) is an important part of usability engineering activities. Practitioners developed many methods, some of them coming from other disciplines (for example psychology, marketing and anthropology). Methods can be grouped based on the delivered data type (quantitative or qualitative), on the goal of the study (summative or formative) or on the persons involved (experts or experts and users). While not all methods produce quantitative data, most can produce easily measured values [24].

Usability is sometimes considered to only deal with the design and deployment of the (Graphical) User Interface of a product. In fact, it engages the deeper layers of the systems too, namely the presentation, task and infrastructure layers [25]:

- Presentation layer means the visual/graphic components, how the system looks like, what the user directly sees when he interacts with the system.
- Task layer: actions and user models, and the operation of the application;
- Infrastructure layer: base technology and data accessible.

This partition shows the depth and the broadness of the scope of the usability discipline.

The most common approach to usability engineering is called User Centered Design (described in [26] as the more general Human-centered Design). While this is more about the general principle of putting the user in the center of the development effort, more implementations exist to realize this. While process standards do exist for usability engineering, especially [26], most practitioners prefer a more hands-on approach, as described in [27] focusing more on the methods and the practical application of the discipline. Recently as the agile paradigm [18] has become widely accepted in software development, the connecting usability teams have also adopted this concept resulting in the agile usability approach [28].

For long, usability was mainly a software-engineering related discipline. With recent technological advancement and ubiquitous computing, usability is now considered in a much broader sense, also applicable to complex systems. This is reflected in the [23] definition too, using product instead of software. The broader interpretation makes it possible to think in usability terms about complex themes like the interaction between citizens and the state (Citizen centered design, [29]).

4 Approaching SPI with Usability

While many practitioners stated that SPI should include people, and most models include people issues, a full methodology on how to approach this question is still lacking. Based on the concept of usability and usability engineering presented in the previous section, we propose to use the existing methodology of usability to deal with the people issues in SPI. In this context the product to improve the usability of are the used processes, while the usability engineering is part of the SPI efforts helping to improve the process by applying its methods.

Applying usability concepts in this environment has two advantages First its approach is focusing on the user and designing systems based on their needs. The resulting systems will have greater acceptance because of user involvement, and will be more efficient because it more accurately captures the needs and expectations of the process performers. If this is an SPI related system, besides acceptance, the performers will have an easier time to follow it, as it was designed with the specific context in mind. The other advantage is the already established set of methods of usability engineering applicable to many kinds of tasks. While some of the methods need adaptation to be usable in an SPI environment, the base ideas stay the same.

To decide if the usability methodology is applicable in this context, we investigated the adaptability of the definition and the matching of the layers described in the previous section.

First, it has to be determined what the basic terms from the definition would mean in the SPI case:

- Product: The system where the SPI is going to be applied, a set of processes, a process model.
- User: The performer of the process, the person doing the task.
- Context: Work conditions and situations, including the organizational and other levels of culture. Some elements of the cultural context may be strongly connected to the user (for example when having a strong national cultural background)
- Task: The process that the user performs. While the preconditions of a given process is defined by outlying elements (business goals, organizational needs, standards) the exact realization, the design of the task and the task conditions are well within the scope of usability engineering.
- Effectiveness: The process has to come to an end with process goals successfully achieved.
- Efficiency: The process execution shall require as little resources and effort from the user as possible
- Satisfaction: The user's experience of the executed process should be positive, empowering.

Second, we describe the usability layers in the SPI context:

- Presentation layer: The quality of the tools supporting the process, the artifacts created
- Task layer: The process execution plan, the model of the system as an inner structure and its recognizability and understandability
- Infrastructure layer: The types of tools used in the processes.

With this, we established a connection and showed the applicability. There are two additional analogies however which make this connection more effective.

- Continuous improvement (CI) and iterative development (ID): The concept of CI is to gather feedback from the process continuously; modifying sub optimal parts based on the feedback while progressing in small steps. The ID (as described in [26]) in usability engineering is based on the assumption that human qualities and needs can not be captured at once. New designs have to be tested with users, and test results with additional gathered feedback incorporated in the next design version (this information gathering process is called formative testing referring both to testing with users to gather feedback and to the forming of the design). CI and ID seems to be based on the same core ideas, and as such the iterative approach of usability engineering complements CI.
- Scalability of efforts: While software processes and process improvement were more viable in large organizations (lack of available and trained staff being among the most important reasons), studies showed its applicability in small and medium sized enterprises [20]. In usability engineering, there are lightweight methods (especially promoted by Nielsen in [27]) aimed at smaller organizations with less resources consumed.

These analogies in approach and application help to create an integrated effort from usability engineering and SPI easier.

The learning organization concept [2] is based on a shared SPI vision, selected empowered individuals and a network of supporters. The application of the usability methodology creates empowered users by creating greater satisfaction. On the other hand, usability engineering makes the creation of shared vision easier by better mapping individual expectations and needs, thus helping in creating the network. This makes the usability methodology suitable for the learning organization concept.

There is previous research mentioning the application of the usability methodology in the field of SPI, but these studies concentrate on the process descriptions, on the physical artifacts of the processes (for example [12]). There is also some work concerning the usability of the tools used in SPI (for example [31]). While both of these fields are important, they represent just part of the scope of usability as they only deal with parts of the presentation and infrastructure layers.

5 Applying Usability Engineering Methods

In the previous section, we presented the viability of the usability approach for SPI and showed its potential values for practitioners. We also noted that one of the advantages of usability engineering is its established method set. To further define the approach of using usability engineering as part of the SPI, we present a brief overview on the application of some of the more widely used usability engineering methods. The methods form an important part of the creation of product of good usability, and most practical usage is based on them. We feel that the base concept of applying the usability approach to SPI would not be complete without some practical points.

Usability engineering has its roots among different disciplines, like psychology, ergonomics, market research and software engineering. Many of the usability engineering methods come from these disciplines adapted to the special requirements of usability. While some of the methods are very context-dependent, most of the widely used ones are well analyzed and are adapted universally to many situations with defined measurements and success factors. The following methods are described in detail in [32] with some helpful supplements. Further methods and practical applications can be found in [27].

Usability tests are the most important method types. Many variants exists, but all of them are characterized by testing the real system (or a proper simulation of) it in real life (or real life like laboratory) context by a representative sample of the users with realistic tasks. Usability tests can uncover most types of issues with measured data mainly on task completion times and success rates. With proper test setup, subjective data (satisfaction, motivation and acceptance) can be measured too.

In the SPI context applying usability test would be similar to pilot projects, but with some significant differences. While pilot projects executed on smaller parts of the organization produce relevant measured data, usability tests can be specifically designed to uncover issues in the system. In a sense while pilot projects are vertical tests (applied to certain parts of an organization), usability test are horizontal tests (applied to specific roles and tasks across organizational boundaries). Planning of usability tests includes the design of test communication, participant selection and task structuring which is more capable to produce statistically sound results.

Prototyping is a sub variant of usability tests based on early involvement of users based on low fidelity little design work. The early involvement strengthens commitment of the users, while concept and high level flaws are shown more easily due to psychological advantages. This means that on the side of the users, they are more likely to provide useful feedback on an early concept, while on the side of the designers with less invested effort into the system they are less likely to insist on certain solutions.

In SPI context prototyping would mean the early involvement of users and trying out new process parts. This is also promoted by the learning organization approach. Prototyping offers the ability to test more concepts early on, enabling to choose the best one for further development.

Walkthroughs are based on the code walkthrough technique used in software development. The task or process is evaluated on a step-by-step basis with each step analyzed with specific targeted questions, for example "Is the next step obvious and recognizable?", "Is the content of the step connected with the goal?" etc.

In SPI context walkthroughs can help in identifying hard to follow, inconsistent process parts. Walkthroughs compare to the widely used reviews and other inspection methods, while providing an added structure and depth to uncover issues.

Artifact analysis is a method to study the behavior of users without the presence of users. It rather analyses the artifacts created by the user's activities. In this context, artifact does not mean the direct result of the activity like a prepared document, rather its side effects. For example, with artifact analysis, we can evaluate the success of an industry fair by analyzing the attendant's photos taken on the fair.

In the SPI context artifact analysis is analogous to measurements, but with objects selected through actual usage traces. Artifact analysis usually searches for objects usually not measured Artifacts are selected on the basis of what the users really used that could have useful information for improvement. For example this could mean the analysis of filled in documents resulting from a process, or the number of help requests on performing process steps.

Personas are user types filled with personalized content. User types usually refer to specific subgroups of the user population with similar goals, motivations, attributes, tasks etc. While user types have a long history of usage in software development, they are consisting of statistics and raw values for whom it is hard to develop for. Personas are more than user types; they represent their types while adding personal content enabling to think about them as real persons. Personas have a psychological advantage when used; the developers have a real person in mind to develop for.

In the SPI context, personas would help representing user groups not directly participating in the process development. Representatives are sometimes not typical group members. Personas help to stay in touch with the ordinary process performers.

In summary, usability methods are well suited to be adapted to different contexts, although care should be taken applying them for the first time in a new situation.

6 Conclusion and Future Work

People issues in SPI are gradually recognized as an important success factor in improvement projects. This paper has introduced the approach of applying the usability methodology as a potential handling mechanism of the people issues in SPI. We showed the applicably of the concept and presented some methods for use in practice.

This approach has a promising potential, little research was however conducted until now on this subject. More evidence is needed for a complete evaluation of the values of this approach. Based on the feedback about the concept, further studies will be conducted on the application of methods in concrete improvement implementations. Ultimately, a formalized SPI model will be constructed building on the usability approach.

References

1. Korsaa, M., Johansen, J., Schweigert, T., Vohwinkel, D., Messnarz, R., Nevalainen, R., Biró, M.: The people aspects in modern (S)PI management approaches. Presented at the EuroSPI 2010 (2010)
2. Messnarz, R., Spork, G., Riel, A., Tichkiewitch, S.: Dynamic Learning Organisations Supporting Knowledge Creation for Competitive and Integrated Product Design. In: Proceedings of the 19th CIRP Design Conference – Competitive Design. Cranfield University Press (2009)
3. O'Keeffe, T., Harington, D.: Learning to learn: an examination of organisational learning in selected Irish multinationals. Journal of European Industrial Training 25(2/3/4), 137–147 (2001)
4. Chirstiansen, M., & Johansen, J.: ImprovAbility*TM* guidelines for low maturity organisations. Presented at the EuroSPI 2007 (2007)
5. Hammer, M. (n.d.) The Process and Enterprise Maturity Model, `http://www.hammerandco.com/HammerAndCompany.aspx?id=58`
6. Jacobson, I., Ng, P.W., Spence, I.: Enough Process - Let's Do Practices. Journal of Object Technology 6(6), 41–66 (2007)
7. Pries-Heje, J., Johannsen, J., et al.: The SPI Manifesto. Web Publishing (2010)
8. Biró, M., Messnarz, R., Davison, A.G.: The Impact of National Cultural Factors on the Effectiveness of Process Improvement Methods: The Third Dimension. Software Quality Professional 4(4) (2002)
9. Biró, M., Messnarz, R., Davison, A.G.: Experiences with the Impact of Cultural Factors on SPI. Presented at the EuroSPI 2001 (2001)
10. Hofstede, G.: Culture's consequences, 2nd edn. Sage publ., Thousand Oaks (2009)
11. Brown, A.: Organisational culture, 2nd edn. Pitman, London (1998)
12. Mahrin, M.N., Carrington, D., Strooper, P.: Investigating factors affecting the usability of software process descriptions. In: Wang, Q., Pfahl, D., Raffo, D.M. (eds.) ICSP 2008. LNCS, vol. 5007, pp. 222–233. Springer, Heidelberg (2008)
13. Kellner, M.I., Becker, U., Riddle, W., Tomal, J., Verlage, M.: Process guides: Effective guidance for process participants. In: Proceedings of the Fifth International Conference on the Software Process, pp. 11–25. ISPA Press, Chicago (1998)

14. Moe, N.B., Dybå, T.: The use of an electronic process guide in a medium-sized software development company. Software Process: Improvement and Practice 11(1), 21–34 (2006), doi:10.1002/spip.250
15. Scott, L.: Understanding the use of an electronic process guide. Information and Software Technology (2002)
16. Wang, Y.: Software engineering processes: principles and applications. CRC Press, Boca Raton (2000)
17. Prikladnicki, R.: QUASE - A quantitative approach to analyze the human aspects of software development projects. In: Proceedings of the 2009 ICSE Workshop on Cooperative and Human Aspects on Software Engineering, CHASE 2009, p. 78. IEEE Computer Society, Washington, DC, USA (2009)
18. Manifesto for Agile Software Development, `http://agilemanifesto.org/`
19. Siakas, K. V., Siakas, E.: The Human Factor Deployment for Improved Agile Quality. Presented at the EuoSPI 2006 (2006)
20. Mumford, E.: The ETHICS approach. Commun. ACM 36, 82 (1993)
21. Maierhofer, S., Stelzman, E., Kohlbacher, M., Fellner, B.: Requirement Changes nad Project Success: The Moderating Effects. In: Proceedings of 17th EuroSPI Conference Systems, Software and Services Process Improvement, Presented at the EuroSPI 2010, pp. 60–70. Springer, Grenoble (2010)
22. ISO/IEC 9126:2001 Software engineering – Product quality (2001)
23. ISO 9241-11: Ergonomic requirements for office work with visual display terminals (VDTs) – Part 9: Requirements for non-keyboard input devices (2000)
24. Tullis, T.: Measuring the user experience: collecting, analyzing, and presenting usability metrics. Elsevier/Morgan Kaufmann, Amsterdam, Boston (2008)
25. Jenson, S.: The simplicity shift: innovative design tactics in a corporate world. Cambridge University Press, Cambridge (2002)
26. ISO 13407:1999 Human-centred design processes for interactive systems (1999)
27. Nielsen, J.: Usability engineering, updated ed. Morgan Kaufmann Publishers, San Francisco (1994)
28. Memmel, T., Gundelsweiler, F., Reiterer, H.: Agile human-centered software engineering. In: Proceedings of the 21st British HCI Group Annual Conference on People and Computers: HCI..but not as we know it - BCS-HCI 2007, vol. 1, pp. 167–175. British Computer Society, Swinton (2007)
29. Hewitt, J.F.: Citizen-centered design (slowly) revolutionizes the media and experience of U.S. elections. Interactions 16, 18–25 (2009)
30. Kirk, D., MacDonell, S.: A Systems Approach to Software Process Improvement in Small Organisations. Presented at the EuroSPI 2009 (2009)
31. Al-Ani, B., Trainer, E., Ripley, R., Sarma, A., van der Hoek, A., Redmiles, D.: Continuous coordination within the context of cooperative and human aspects of software engineering. In: Proceedings of the 2008 International Workshop on Cooperative and Human Aspects of Software Engineering, CHASE 2008, pp. 1–4. ACM, New York (2008)
32. Sears, A.: The human-computer interaction handbook: fundamentals, evolving technologies, and emerging applications, 2nd edn. Lawrence Erlbaum Assoc, New York (2008)

A Study of Software Development Team Dynamics in SPI

Shuib Basri [1,2] and Rory V. O'Connor[1,3]

[1] Lero, the Irish Software Engineering Research Centre, Ireland
[2] Universiti Teknologi PETRONAS, Bandar Sri Iskandar, 31750 Tronoh, Perak, Malaysia
shuib_basri@petronas.com.my
[3] School of Computing, Dublin City University, Ireland
roconnor@computing.dcu.ie

Abstract. The software development team is a key factor in software projects, however, achieving and maintaining positive team dynamics in software development project especially when the software companies have fewer resources in term of people, money and time is a remarkable challenge. This paper explores the dynamics of software development teams (*structure, process, communication, learning and sharing)* and its impact on Software Process Improvement (SPI) in very small software organization, in order to understand the impact between these two variables. We undertook a series of interviews and focus groups with very small software companies and our results show that very small companies have a high level of team dynamics although their SPI initiatives are conducted on a small scale and in an informal and indirect manner. The results also indicated that this situation occurs due to the working and social relationship, willingness to share, having a good interpersonal skill and work closely each others.

Keywords: SPI, VSEs, Team Dynamics, Grounded Theory (GT).

1 Introduction

Software processes are related to software development and are highly dependent on human decision making and judgement. In software development, people factors are not the only important to be consider in the process but they are also a determiner in project success [1]. People involvement in improvement activities is important because employees must adopt process innovation in their day to day activities [2]. Moreover the ways people have been treated are the important factors in organization development and change [3]. In specific, employee participation is the strongest influence on Software Process Improvement (SPI) success and, in general, peoples are the main factor in software process improvement that needs to be encouraged and support in an organization [3]. Furthermore, lack of people involvement in development activities will disturb the improvement process. Hence the aim of process improvement will be fail if people are not commit to all the propose change activities [4]. In addition the strengths and weaknesses of the current process are inside the staff hands and knowledge [5]. Hence, that even though people are the main driver for software quality but the processes has been given more attention [6].

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 143–154, 2011.

Therefore in order to be success in SPI, organization must have a solid support from the software development and management team. In addition, the development and management team dynamics characteristics must exist to able to work together, share the knowledge and able to communication one another effectively. This is because the essence of software development is good relationship, effective communication and high esteem of teamwork among software development and management team in process improvement are critical in all time. This situation is become more important especially in Very Small Entities (VSEs) whose have limited resources, particularly in financial and human resources and are practicing unique processes in managing their business. Therefore, this paper aims at presenting a more comprehensive perspective of software VSEs team dynamics towards SPI initiatives. This paper is concerned with understanding VSEs issues regarding the impact of their software development team dynamics to their SPI initiatives.

2 Background

2.1 Very Small Entities (VSEs)

The definition of small and very small enterprises is challenging. To take a legalistic perspective the European Commission defines three levels of small to medium-sized enterprise (SME) as being: Small to medium - "*employ fewer than 250 persons and which have an annual turnover not exceeding 50 million Euro, and/or an annual balance sheet total not exceeding 43 million* Euro. The term "Very Small Entity" (VSE) had been defined by ISO/IEC 29110 as being "*an entity (enterprise, organization, department or project) having up to 25 people*" [8].

Micro enterprise including VSEs whose have limited resources, particularly in financial and human resources, are basically practicing unique processes in managing their business. These unique characteristics and unique situations have influenced VSEs in their business style compare to large companies [9]. In addition, these limitation and characteristics have given a big impact to companies' process infrastructures [10]. Moreover most of the management processes are performed through an informal way which most of decision-making, communication and problem solving been discussed orally and less documented. This indicates that people-oriented and communication factors are very important and significant in VSEs [11].

2.2 Software Process Improvement (SPI)

The primary goal of software development has changed from "conforming to plan" to "satisfy the customer - at time delivery, not a project initiation" [2]. Therefore the improvements of the development processes in order to be handle the rapidly changing environment and requirement are very significant. There are 4 categories that could influence organizations involved in SPI namely the economic, people, organization and implementation factors [12]. Research in very small teams found that over 12 months, monthly cost and benefits have shown a positive impact of their monthly value [13]. The people factors that are related to SPI have been discussed in literature. The success of software project and process is determined by the interest of software team on the project and process itself [14]. In small software organization the influence of key individuals is a major influence [15]. However staff participation also is essential in improvement activities as they have detailed knowledge and experiences of of the current process [16].

2.3 SPI and Teams

In software development, human factors are not the only important to be consider in the process but they are also a determiner in project success [1]. Software development is not just creating an effective programming and tools, but also depends on people, organization and procedure. People involvement in improvement activities is important because employees must adopt process innovation in their day to day activities. The lack of involvement will disturb the improvement process because if employee did not commit themselves to all the propose change activities, the aim of process improvement will be fail [4]. Moreover, the strengths and weaknesses of the current process are inside the staff hands and knowledge [5]. Hence those people are the main factor in software process improvement that needs to be encouraged and support in an organization [2]. Moreover, [6] stated that the effect of software development team on the software product quality claimed that even though people are the main driver for software quality but the processes have been given more attention. Therefore the involvement and full commitment from teams in process improvement is critical.

The dynamic performance software project which involved many processes are always depends on the team especially in quality of communication within team. Moreover the communication may take many forms, both verbal and non-verbal [17]. Previous research shows that the level of communication in software process in depends on the size of software project [18], where they authors claim that for a small project the interaction between team members is adequate but for a larger project a mix interaction between team member and specification are required. Communication also has a related impact with the team proximity in that the increase distance from one team to another could effected the team dynamics in which it will interrupt team communication, coordination, mutual support, effort and cohesion [19]. Hence the link between team member also becoming more difficult with the increase of the team member and this will impact the team dynamics [20].

2.4 Teams Dynamics

Team dynamics are the hidden strengths and weakness that operate in a team between different peoples or groups. Team dynamics could effect how team reacts, behaves or performs and the effects of team dynamics are often very complex [21]. There are various forces could influence team dynamics including nature of the task, the organizational context and team composition. McCarty in her dissertation on dynamics of successful software team identified four characteristics of team dynamics; positive, negative, internal and external team dynamics [22]. Positive team dynamics is referred as positive forces that could lead the team to create a high performing successful team. The present of social relationship in a team could increase team productivity and enhance social and interpersonal skill [23]. Hence, the positive mode of leadership (such as well focus directive, well plan and others) in software organization could enhance the positive team dynamics [24]. Negative team dynamics is referred as negative forces that could lead the decrease of team performance and preventing people from contributes with their full potential [22]. From management point of view, in software development organization people are required three types of needs that have to be fulfilled and satisfied; social, self esteem and self-realisation needs. Internal team dynamics are referring to the forces that exist within the team itself [22]. Team member will not cooperate if they do not feel that that are a part of the team [20]. Ayman argues that within a team, roles and norms must be clear [23]. Littlepage et al. adds that cohesiveness is essential for an effective team performance and will enhance team cohesiveness [25]. A cohesive team will freely challenge each others and easily sharing new knowledge with other team members. External team dynamics are referring to the present of external forces that beyond the team control and could impact the team performance [22]. The intrinsic and extrinsic factors in projects may motivate team. Intrinsic factors are the internal factors that consist in the task and team activity itself [26]. Extrinsic factors are external factors that influence team from the outside such as reward and recognition, feedback from the organization and customer, team member pressure and the working environments.

3 Research Study

In order to carry out this study, a parallel approach was decided, composed of a qualitative data collection (questionnaire, interviews and focus groups) and quantitative data collection (questionnaire), with data analysis being completed separately and finally the results were merged. The overall data collection process is shown in Figure 1. We interviewed and distributed a survey questionnaire to software VSEs in Ireland. These companies were all directly involved in software product development for a variety of business domains and were determined based on researchers' personal contacts. Due to space restrictions, in this paper we only present the results from the qualitative data.

For qualitative data collection two complimentary data collection methods, (i) individual and focus group interviews, and (ii) survey questionnaire have been adopted in this study. The individual interview approach was used in this study in order to discuss the topics in depth, to get respondents' candid discussion on the topic and to be able to get the depth of information of the study situation for the research context [27]. This process followed by semi-structured interviews approach which includes the open-ended and specific questions. This approach allowed us to gather not only the information foreseen, but also unexpected type of information [28]. The respondents for the individual interview session are all software development managers / CTO / owner-directors and the focus group was with software development staff. The focus group interview approached was used in this study because team members develop the software and the existence team interactions helped to release inhibitions amongst the team members and are from the same company as the individual interviews participants. Focus group interviews were also chosen because it was the most appropriate method to study attitudes and experiences;

to explore how opinion were constructed [29] and to understand behaviors, values and feelings, [30]. In order to gain more input and also to validate the above qualitative data for this study, we have developed and distributed a survey questionnaire to several Irish software VSEs. These companies were selected using personal contacts and were all directly involved in software product development, for a variety of business domains.

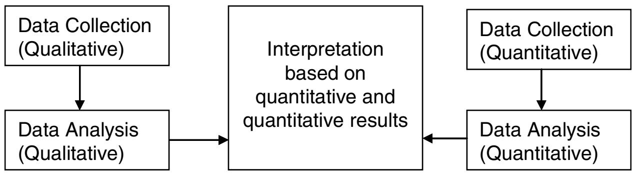


Fig. 1. Research Concurrent Design- Data Collection

For the open ended data, we analyze and categories the data according to the category that this study intends to understand. The answers were group, coded and list into a table in respect to the study category issues. In overall we followed the qualitative contents analysis approach and adopted the Grounded Theory (GT) coding process in analyzing the open-ended answer [29][31]. Furthermore, in order to produce details analysis result, we have divided the survey respondents into 2 main group namely the Micro VSE (M) (1-9 employees) and Larger VSE (L) (10-25 employees).

4 Study Findings and Discussion

4.1 Team Structure and Process

From the qualitative data analysis process which adopted the GT coding approach, we consider VSEs could be divided into 2 categories; the organization and team structure category and the team process category as shown in table 1. The organizational and team structure category indicates that due to the small number of people working in the organization, the team size is also small and this leads to a flat team and organizational structure. From the interviews analysis results indicate that all interviewees admitted that the companies have no formal team structure or a team structure only exists occasionally as maybe required for a particular project. For example, one team leader told us:

> *"There are 5 developers including me and peter. No we don't have a formal team development structure at the moment, we all have the same skill and it is very flat."*

In additional during the analysis researchers found that due to the small number of employees, flat organization and team structure and informal environment, interviewees perceive that all people in the company are at the same level. In addition the analysis show that they have the similar level of working experience, skills and very much depends on each other in performing their task. Besides the close working

Table 1. Team Structure and Process

Sub Category	Category	Main Category
Team Size - Small	Organizational and Team Structure	Team Structure and Process
Organizational and Team Flat Structure -		
Team Role - informal	Team process	
Team Involvement - direct		
Team Culture – informal		

space or area, high frequent and informal communication are also influences this perception. All these criteria have leaded VSEs in narrow down the gap between the management and the team development. An interview extract below best represents this situation:

> *"We don't have that [formal team structure] but I can see in a large company where might have that. In small company I think it is a bonus we know each other very well"*

The second category is the team process category which indicates the team role, team involvement and team culture issues. The analysis shows that the staff role which includes the role in team and the task they perform in development process is very informal and very general. This could imply that the development staff could work or be assigned a different role at any time in organization development project. In addition they also can work with others or different people and different position as and when they are required. These situations have explained that team involvement process in VSEs is direct and informal in development activities. An interview extract below represents the situation describe above:

> *"Usually either face to face between 2 developers or over Skype with 2 developer remotely communication. In general the developers work independently and have a sole responsibility for the project. Other times they assist each other for a single project."*

4.2 Communication Process

During the interviews sessions, the researchers have asked several questions on communication in order to understand this issue in VSEs. From the analysis, the researchers could divide the communication process in VSEs into 2 categories namely open and informal communication category and online communication category. The analysis also shows that the communication process in VSEs is influenced by the companies team structure and process and the working and management style as shown in table 2.

Table 2. Communication Process

Sub Category	Category	Main Category
Team Structure and Process Working and Management Style		Communication Process
Open Communication	Open and Informal Communication	
Informal Communication		
Communication tools	Online and Electronic Communication	
Internet/ Electronically		

In the open and informal category, the researcher has identified 24 interviews extracts that represent the category where people are more towards informal and direct/casual communication. This can be identified in the ways meeting are conducted, which are more informal, ‘stand up’, periodic and individual. In addition, the interviewees also agree that their day to day communication between staff is very direct and autonomous, due to the working environment in their company. This situation is confirmed by the interviewees, stating that because of the small team size that exists in the organization and the working style culture which is more toward autonomous work have create this situation. Below interview extracts below illustrate this:

“We have a formal meeting once a while but most of it is more informal. It is informal when we discuss development stuff like over the coffee. We usually share our code esp. with peter and he will look at it and share the idea. Later we will introduce to others and ask for feedback. We have informal meeting for a few minutes just to inform others regarding process before we start our tasks.”

In addition the analysis also indicates the relationship between staff in the company also influences the communication process in VSEs. The analysis shows that the family and flexible environment, frequent social interaction between people and flat organization structure have given an impact on communication process in VSEs. Beside that the closeness people working space or area and high frequent of sharing activity have contribute to the communication process in VSEs. Two supporting interview extracts would be:

“I see a very open, very congenial very friendly and professional environment… I see people on the equal sourcing, openly discussing,. There no very rigid formal hierarchy. The team easily talks to management as we sit side by side. “

“We work very close, meet for morning coffee. We always mix together and are very dynamic because we are small and easy to communicate each other.”

The second category in this part is online and electronic communication category. From the analysis, the researchers found that the use of communication tools such as email, phone, blog, skype and internet are very active in VSEs. Such communication tools are vital to the company that has a staff member working in different locations. From the analysis researchers found that the main purpose using communication tools beside to communicate between staff members, it also the tools that could close the gap between remote and collocated staff. The analysis also indicates that the use of communication tools is to allow staff to share and document all work related information or knowledge in informal way. The quote explain this:

"We always skype with and other tools chat message, VPN, blog and others. We have company internal blog to share the information among us"

4.3 Learning and Sharing Process

The interview data analysis elaborates how the learning and sharing process happens in VSEs. The analysis shows that the learning and sharing process main category could be detailed up into 2 important categories namely self learning category and sharing category as in table 3.

Table 3. Learning and Sharing Process

Sub Category	Category	Main Category
Communication Process Working and Management Style Team Structure and Process		Learning and Sharing Process
Training	Self Learning	
Self Learning		
Continuous Guidance		
Internal Training	Sharing	
Meeting		
Document		

In the self learning category, the analysis shows in VSEs there are no formal training given to employees in enhancing their knowledge or skills. In the analysis informal training has been defined as internal training, sharing and self learning. The analysis also has explained that people in VSEs are more dependent on self learning in mastering the technology or process that is used in the organization. Besides self learning, the analysis also shows on the job training, self exploring and continues guidance from expert with in the companies are the main process that frequently been practiced in enhanced staff knowledge and skills. The following extracts are illustrative of this point.

"We haven't done any formal training but we do give our employee an opportunity to attend various courses and seminars."

"It wasn't a formal training… what I mean once you get started you could find out, who to do certain things, someone have experience can show you the way of the main resources or he can read article with your interest you want to carried out certain task. It wasn't a formal training period, I just call training because I actually learn and still learning but now is not as before"

The second category in this part is the sharing category. The analysis shows that in VSEs the knowledge sharing process happens in 3 ways: informal training, informal meetings and document sharing. Informal training happened through informal and guidance from expert, peer to peer programming process, shared books and others material, internal training, high frequent open and direct discussion with team member and online sharing with others. The informal meeting process happens through an informal stand-up meetings, direct and open discussion and online meetings via email and Skype. While the document sharing process have been done through note sharing and online sharing (e.g blog, email) which are informal and very personal. In relation, the analysis indicate that the learning and sharing process in VSEs is been influenced and shaped by 3 main factors which are VSEs team size and process which are small team size and flat organization structure; working and management style which are more toward autonomous work and macro management process and, communication process which are indirect and informal process. In addition from the interviews data analysis shows that in general knowledge sharing activities either via electronic or personal means are important in maintaining and evolving the current VSEs software development process. The quotes below have explained the above situations.

> *"However when you want to do a new things and you want to introduce a new methodology you discussed with the rest of the team, that is good and also we are supporting, if you want to do something but you not sure, you can go to any others who has more an expertise in the same area"*
>
> *"We shared books and we buy books and we pass around. Generally it is informal process just asking question, grasp him and talk. Sometime we did pair programming but not always. Generally it is some kind of informal."*

4.4 SPI –Process Improvement and Assessment

The results from the survey questionnaires have indicated that in general respondents are agreed that their software development processes rapidly change and evolve overtime. They also claimed that their development process are regularly assesses and staffs always followed or applied the latest development process method. Moreover the analysis also shown that 90% of respondents felt that their development process evolves overtime. They stated that following the best practice, client requirement, team size growth, new idea and keep up with the technology change are the reasons for the improvement and evolution of development process. The following three extracts from the open-ended questions give an indication as to how the development process have been improved and evolved with a company:

> *"Software process change is due to growth of the organization. We started out as 2 people 4 years ago and now have 11, so things had to change along the way"*
>
> *"It will evolve as we grow in size and get more applications in production environment"*
>
> *" We still do the same basic thing in software process; we change some aspects of how we work. It's a little bit ad-hoc..."*

Furthermore that in question on related to the process loss issues shows that almost all or 80% of respondents' claimed that their software development processes are not affected by the process loss problem. They claimed that by using standard development tools, similar development process, having frequent guidance and mentoring activities, active in knowledge sharing and proactive coaching could avoid the process loss problems in software development process. The following extracts from the open-ended questions illustrate this situation:

> *"As a manager, I don't believe in using the latest and greatest techniques for the sake of it. We'll use something that fits our team dynamics and we'll spurn something that doesn't… whether that counts."*
>
> *"Our document process mostly electronically…we always sharing knowledge informally. Since this is family business, we always having informal regular meeting"*
>
> *"Not really, we still do the same basic things in our software development method. We change some aspects of how we work. It's a little bit ad-hoc... Agile method… I suppose"*

However the respondents also admitted that "laziness" attitudes among some staff and practicing informal and rapid changes in software development process are among the factors that could lead the process loss problem in software development process.

5 Conclusions

The findings indicated that respondents are agreed that their software development process frequently change and evolved over time. They also agreed that they regularly assess and update their development processes. However the finding showed that the changed and evolved processes are informal, indirect and very reactive which depends or is linked to customer requirements, developers' initiatives and technology changes. The results also indicate this situation was influence by the team structure and process is very flat and informal in VSEs. These issues have also determined the formality level of software process improvement activities in VSEs. The results also indicate that these issues also affect other critical main categories which related to VSEs software process and process. This also have create a close relationship which create a between software development team and indirectly create a high level of team dynamics and knowledge sharing activities in software development activities a shown in communication, learning and sharing category result. Beside that the external environment such as macro management style; autonomous working style active feedback from peers and management and direct involvement of management people in software activities which also have created conducive environment to the software development team in VSEs.

Furthermore the results have also shown that we found that all respondents agree that the software development team dynamics is very high. This could be identified from how the communication, relationship and learning and sharing environment status in VSEs. The results also indicated that the smaller the team in VSEs the higher level of team dynamics will be presents in the organization. In addition, the analysis also have indicate that VSEs staff have all the important criteria such as high skills, high motivated, active in sharing, direct involvement and open communication, which are important in software development process.

Acknowledgments. This work were supported, between Science Foundation Ireland grant 03/CE2/I303_1 to Lero - the Irish Software Engineering Research Centre (www.lero.ie) and Universiti Teknologi PETRONAS, Malaysia (www.utp.edu.my).

References

1. Rosen, C.C.H.: The Influence of Intra Team relationships on the systems Development Process: A theoretical Framework of Intra-Group Dynamics. In: 17th Workshop of the Psychology off Programming Interest Group, Sussex University (2005)
2. Bin Basri, S., O' Connor, R.V.: Organizational commitment towards software process improvement an irish software VSEs case study. In: 2010 International Symposium on Information Technology (ITSim), Kuala Lumpur, June 15-17 (2010)
3. Dyba, T.: An empirical investigation of the key factors for success in software process improvement. IEEE Transactions on Software Engineering 31(5) (2005)
4. O' Connor, R.V., Basri, S.: Exploring Managerial Commitment towards SPI in Small and Very Small Enterprises. In: Riel, A., O'Connor, R., Tichkiewitch, S., Messnarz, R. (eds.) EuroSPI 2010. Communications in Computer and Information Science, vol. 99, pp. 268–279. Springer, Heidelberg (2010)
5. Basri, S., O' Connor, R.V.: Evaluation of Knowledge Management Process in Very Small Software Companies: A Survey. In: Proceeding of 5th International (KMICe 2010) Conference on Knowledge Management, Kuala Terengganu, Terengganu, Malaysia, May 25-27 (2010)
6. Beaver, J.M., Schiavone, G.A.: The effects of development team skill on software product quality. ACM SIGSOFT Software Engineering Notes 31(3) (2006)
7. European Commission, The New SME Definition: User Guide and Model Declaration (2005), http://www.europa.eu.int/comm./enterprise/enterprisepolicy/sme_definition/sme_user_guide.pdf (accessed on: February 26, 2007)
8. Laporte, C.Y., Alexandre, S., O'Connor, R.: A Software Engineering Lifecycle Standard for Very Small Enterprises. In: O'Connor, R.V., et al. (eds.) Proceedings of EuroSPI. CCIS, vol. 16, pp. 129–141. Springer, Heidelberg (2008)
9. Mtigwe, B.: The entrepreneurial firm internationalization process in Southern African context: A comparative approach. International Journal of Entrepreneurial Behaviour and Research 11(5), 358–377 (2005)
10. Sapovadia, V., Rajlal, K.: Micro Finance: The Pillars of a Tool to Socio-Economic Development. Development Gateway (2006), SSRN http://ssrn.com/abstract=955062
11. Valtanen, A., Sihvonen, H.M.: Employees' Motivation for SPI: Case Study in a Small Finnish Software Company. In: Proceeding of the 15th European Conference, EuroSPI 2008. CCIS, vol. 16, pp. 152–163. Springer, Heidelberg (2008)
12. Hall, T., Rainer, A., Baddoo, N.: Implementing Software Process Improvement: An empirical Study. Software Process, Improvement and Practice 7(1), 3–15 (2002)
13. Batisha, J., de Figueiredo, A.D.: SPI in a Very Small Team: A Case with CMM. Software Process Improvement and Practice 5(4), 243–255 (2000)
14. Komiyama, T., Sunazuka, T., Koyama, S.: Software process assessment and improvement in NEC - current status and future direction. Software Process Improvement and Practice 5(1), 31–43 (2000)

15. Knauber, P., Muthig, D., Schmid, K., Widen, T.: Applying Product Line Concepts in SME. IEEE Software 17(5), 88–95 (2000)
16. Stelzer, D., Mellis, W., Herzurm, G.: Software Process Improvement via ISO9000. Result of two surveys among the European software houses. Software Process Improvement and Practice 2, 197–210 (1996)
17. Hall, T., Beecham, S., Verner, J., Wilson, D.: The Impact of Staff turnover on Software Project: The Importance of Understanding What makes Software Practitioners Tick. In: Proceedings of ACM SIGMIS CPR, pp. 30–39. ACM, New York (2008)
18. Phongpaibul, M., Boehm, B.: Improving quality through software process improvement in Thailand: initial analysis. ACM SIGSOFT Software Engineering Notes 30(4), 1–6 (2005)
19. Hoegl, M., Proserpio, L.: Team Member Proximity and Teamwork in Innovative Projects. Research Policy 33(8), 1153–1165 (2004)
20. Furumo, K., Pearson, J.M.: An Empirical Investigation of how Trust, Cohesion and Performance Vary in Virtual and Face to Face Teams. In: Proceedings of the 39th Annual Hawaii International Conference on System Sciences, vol. 1, pp. 26c (2006)
21. Scarnati, J.T.: On becoming a team player. Team Performance Management 7(1/2), 5–10 (2001)
22. McCarty, B.: Dynamics of a successful Team. What are the enablers and barriers to High Performing Successful Teams? MSc Dissertation, Dublin City University (2005)
23. Ayman, R.: Impact of team diversity on collaboration dynamics. In: Collaborating across Professional Boundaries (2000), `http://www.stuart.iit.edu/ipro/papers/pdf/ayman.pdf` (accessed on: April 25, 2007)
24. Singh, S.K.: Role of leadership in knowledge management: A study. Journal of Knowledge Management 12(4), 3–15 (2008)
25. Littlepage, G.E., Cowart, L., Kerr, B.: Relationships between Group Environment Scales and Group Performance and Cohesion. Small Group Research 20(1), 50–61 (1989)
26. Kirkman, B.L., Rosen, B., Tesluk, P.E., Gibson, C.B.: The impact of team empowerment on virtual team performance: The moderating role of face-to-face Interaction. Academy of Management Journal 47(2), 175–192 (2004)
27. Kvale, S.: Doing Interviews. The Sage Qualitative Research Kit. Sage, Thousand Oaks (2007)
28. Li, J.Y.: Process Improvement and Risk Management in Off-the Shelf Component-Based Development, PhD Thesis, Norwegian University science and Technology (2006)
29. Kitzinger, J.: Introducing focus groups. British Medical Journal 311, 299–302 (1995)
30. Patton, M.Q.: Qualitative Evaluation and Research Methods, 3rd edn. Sage Publications, Inc., Newbury Park (2002)
31. Elo, S., Kyngäs, H.: The qualitative content analysis process. Journal of Advanced Nursing 62(1), 107–115 (2008)

An Empirical Investigation into Social Productivity of a Software Process: An Approach by Using the Structural Equation Modeling

Murat Yilmaz[1] and Rory V. O'Connor[2,3]

[1] Lero Graduate School in Software Engineering, Dublin City University, Ireland
murat.yilmaz@computing.dcu.ie
[2] Dublin City University, Ireland
[3] Lero, the Irish Software Engineering Research Centre
roconnor@computing.dcu.ie

Abstract. The actual and expected benefits of fostering the alignment of people factors and cooperation among software development teams enables software development organization to improve software development productivity. Furthermore, software development productivity presents a significant challenge for both understanding and quantifying the performance characteristics of software development organizations. This paper introduces an approach to model software development productivity by using structural equation modeling (SEM), a technique that can be used for testing and estimating relationships using empirical data. We also present preliminary results from an exploratory study about the enabling social factors that affect software development productivity. Our quantitative analysis involves grouping productivity and social productivity factors for studying and identifying their implicit relationship. To this end, we issue questionnaires to test our hypothesis and to gather sample data. The paper concludes by showing initial results, limitations, and directions for future research.

1 Introduction

In contrast to other aspects of software process improvement (SPI), software productivity improvement is a multi dimensional concept with a means of achieving and sustaining a competitive advantage. As software development is considered to be a human endeavor (i.e. effort and intellectually intensive team work) [1], the human and social aspects of software engineering has turned into an important topic to investigate for both scholars and practitioners. It is therefore not surprising that experiencing greater production success heavily relies on how the teams socially communicate, and utilize their interactions. These interactions however, should be governed and coordinated to achieve the desired productivity levels both for individual and a team as a whole. Although extensive research has been carried out about many of the social aspects [2], no study exists which

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 155–166, 2011.

adequately uncovers the relationship between the productivity factors and the social aspects affecting productivity of software development projects. We therefore conduct a study of this relationship and in doing so, we identify an indicator for defining the social aspects influencing productivity, we term this indicator as *social productivity of software development*. Social productivity is a dimension of productivity which addresses improvement issues about social interactions corresponding to the basis of the social structure of a software team.

1.1 Objective of Research

Our preliminary study aims to empirically analyze the dynamic interactions between the factors of social productivity and productivity based upon software productivity literature and refined by our focus group studies. We apply a structural equation model (SEM) approach for evaluation of the central hypothesis (i.e. a positive correlation between influential factors of social productivity and software productivity) introduced in this paper to be tested and to provide an initial empirical support for our proposed model. One dimension of the model consists of several important productivity factors found in the literature, where other dimensions define several indicators of the social aspects of software productivity. Most importantly however, this paper is among the first attempts to use a technique like SEM to examine the impact of social aspects of software development productivity.

The remainder of this paper is organized as follows: In the next section, we introduce several definitions for the social dynamics of software development settings. The following section describes the analysis models and methods in more detail and presents some some preliminary results that prove the feasibility of our proposed model, and to verify our empirical approach. Finally, the last section concludes the paper with a brief summary of contributions and the directions for future research.

2 Social Dynamics

Social dynamics is an multi disciplinary field of science that concerns the process of analyzing socialites or social systems expressed by actors and their interactions based on rules or norms. These definitions provided by the field of social dynamics help us to highlight the important points of the Social Aspects of Software Engineering (SASE) [2]. Ultimately, SASE will help us to understand the social dynamics of a software organization in order to promote cooperation within software teams and organizations, and to respond to the dynamic trends of present and future of software development.

2.1 Productivity

Software production is the economic process of conversion of inputs to outputs based on resource consumption and allocation. Thus, one of the concerns of software process improvement is investigating methods to improve and measure the software productivity. In general, productivity is a value to measure the efficiency of this production process.

A common definition of software productivity from the literature is the ratio between the inputs (e.g. the cost of work/resources) versus the outputs (i.e. software artifacts or services) within the production process of software development [3]. However, it is hard to find a suitable way for measuring productivity [4] because, it may be considered differently for stakeholders from their distinctive perspectives. For example; from the viewpoint of developers, a productivity measure would be the amount of code produced for the software system, on the other hand from the user's perspectives; it could be the the degree of functionality achieved for the software system. An increase in the productivity is achieved when activities and resources in the software development process are use to add more value to the software product.

During several development activities multiple outputs are produced concurrently, hence Scacchi [5] suggest that a multi dimensional analysis of productivity is important in software development settings. Productivity can therefore considered to be a multi dimensional problem significantly affected by many factors including the quality of workforce, management capabilities and environmental conditions of a software organization. However, the social factors of software productivity can't easily be identified, e.g. cost of communication and social expenses [6].

Abdel Hamit [7] defines the notion of potential productivity where maximum productivity is only achieved if an individual or a team uses their maximum potential. He added two factors that are important for representing the shortfalls for software quality and productivity problems; (i) task characteristics (i.e. complex nature of a task) and (ii) team resources (i.e. fitting individuals or team skills over tasks and tools). However, these factors could increase the cost of communication and lower the motivation of individuals and software teams.

Over the past few decades, software productivity has been investigated by using several indicators affecting the productivity. One such approach is conducted by Pfleeger [8] who uses a statistical method called regression analysis. By using this technique, he constructs an estimation model of productivity where he calculates the effects of cost factors in a predictive manner. Moreover, regression analysis has also been applied for determining the correlation between size and effort for software development projects [9].

Finally, productivity improvements can be achieved by having a skillful team, improving the path of development by reducing rework, and by creating reusable and more manageable software artifacts [10].

2.2 Social Capital

The classical notion of capital states that the capital becomes apparent from the social interactions between capitalists and laborers. In other worlds, it is an end product of a social process. Social Capital can be defined as the capital which is attracted and held by social connections and networking so as to make a gain or profit. Lin [11] defines social capital as an "*investment of social relations with expected returns in the market place*". Bourdieu [12] defines the term social capital as a mass of present and future resources that are linked as a network of relationships. His definition designates that social capital is based on two components; (i) social relationships which affords possibilities to help them obtaining accessibility to the resources by their relationships, and (ii) resource quality.

Social capital can be seen as an-other resource to be captured by individuals. According to Portes [13] social capital is inherent in the fabric of actors and relationships. In order to own social capital, one should have linked with others. Therefore, social capital should be measured with respect to the quantity and quality of social connections that one might have. Coleman [14] argues that all kinds of social structures and relations enable some form of social capital. As a matter of fact, individuals intentionally connect with one and other to form social networks and expect benefits from these actions.

The level of social capital attainable by participants of a software development organization will ensure the enthusiasm of teams and individuals to cooperate in a voluntary manner. Social capital should help to improve the social coordination and stability. Therefore, it will enable us to have an efficient information exchange network.

2.3 Social Productivity

In the socio-economic landscape of software organizations, increasing the efficiency and productivity of individuals and organization by improving their social capital depends on the subset of various facts or several circumstances (e.g. quality of social interactions). The act of understanding the impact of social relations in process, tasks and activities of development can be considered as an important aspect of productivity.

We define, ***social productivity*** *as the production rate of software development increases if we give due consideration to maximizing the social relations.*

Therefore, we claim it is important to understand the concept of social productivity as a measure (level) for collaborative outcome by social interactions through a software company. Accordingly, this quantification can be use to improve the positioning of the teams and individuals in software organizations. It also can help the transformation process of actual resources (e.g. human knowledge, team skills, time, technology) into assets (i.e. software artifacts).

The notion of *social productivity of software development* aims to highlight the social outputs of organized groups and the importance of interactions and behaviors. It may have some beneficial usage for measuring values like cooperativeness or some harmful formations (e.g. conflict of interests) for not only for the software teams but also for the entire software development organization.

3 Models and Methods

This section describes models and methods that are used in our investigation of social productivity factors. First, we introduce the structural equation modeling which our productivity model is based on. Next, we highlight the benefits of a focus group and explain our effort to identify the factors of productivity in an industrial setting. Further, we depict our model for software productivity and the framework that we used to conduct the research.

3.1 The Systematic Approach

Here, we develop a systematic approach to address the relationship between productivity and social productivity (see figure 1). First, based on a structural equation model, we formed a hypothesis which states that social productivity is highly correlated with productivity. Second, we reviewed the software productivity literature to investigate the factors affecting productivity. After identification of several productivity factors, we proposed a set of factors affecting social productivity. To evaluate this proposal, third, we conducted a focus group research and consulted a software company for their opinion about these identified factors, and consequently utilize this information to change some our initial settings. Fourth, we create a survey instrument for testing and validating the causal relationships we proposed among several factors (i.e. observable and latent variables) and so as to refine the structural equation model. Finally, to identify and to examine causal relationships among several factors that are affecting the quality of software development, we conducted a survey by using graduate and post graduate university students.

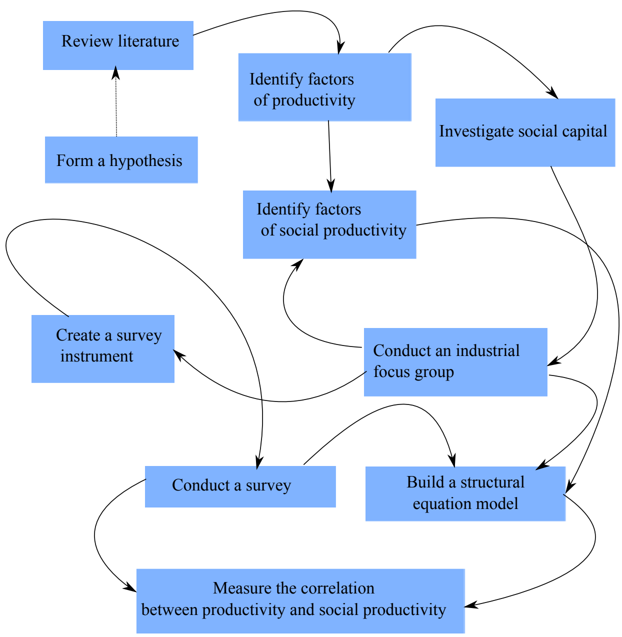


Fig. 1. The systematic approach for software productivity research

3.2 Structural Equation Modeling

As several aspects of social issues are qualitative by their nature, it is hard to develop a precise measurement model and to attain reliable and valid results from any type of people based measurements and observations. There is a number of limited quantitative approaches, even these are constrained with a classical viewpoint. They are mostly based on very basic statistical concepts. These concepts, however, have been limited with the numbers of factors as they were applied to software process and productivity improvement efforts. To fill this gap, we suggest that a complex phenomenon such as software productivity can be modeled by using a solid statistical approach to address problems that classical approaches can't easily handle.

Frequently used in social science studies, a family of flexible interrelated statistical techniques (i.e. multivariate, multiple regression analysis, factor analysis) for analyzing empirical data and testing variables and evaluating their network of hypothesized relationships is called structural (simultaneous) equation modeling (SEM) [15]. Based on patterns of statistical expectation, it is a confirmatory multivariate (multi equation) analysis technique for estimating the structural or casual relationship among the variables that are observed and latent, and specifying relations among these latent variables.

SEM models use a collection of simultaneous equations, which are based on a combination of observed and latent variables (hypothetical constructs or factors), which are introduced and frequently used by sociology and psychology research and econometric methods. A typical SEM structure has up to three simultaneous equations which includes (i) a measurement model that can have dependent variables, (ii) a sub-model with independent variables and (iii) a structural submodel for concurrent estimations [16].

Although it is a quantitative approach, SEM offers a start from a qualitative viewpoint. It has the ability to show how factors are correlated and also interrelated to one other. Therefore, it can be helpful for observing the relationship among several selected factors or coefficients. It helps us to investigate how a hypothetical model might be effectively fit with sampled data. In particular, a model based on regression, path, and confirmatory factor analysis will help us to analyze the importance of several factors such as observed and latent variables and their interdependencies.

3.3 The Measurement Model

We chose productivity and social productivity as latent variables (i.e. one type of factor) for our structure equation model. Although some approaches address productivity as a construct, no previous study has been found on social productivity factors and its relationships with productivity of software development.

Our model is based on factors affecting productivity and social productivity. By using the productivity literature in general, and software development productivity in particular we chose five factors that have been mostly referenced by researchers. The initial factors we found important were; (i) Motivation, (ii)

Process, (iii) Reuse, (iv) Complexity, and (v) Team Size. Next, we aimed to use four observed variables including; (i) Leadership, (ii) Trust, (iii) Communication, (iv) Team Cohesion for the measurement of social productivity.

It may be difficult to obtain rich and insightful data from practitioners in a specific area of interest using both qualitative and quantitative research method. However, we argue that a focus group is an efficient way to reach that information [17]. The group setting may be ideal for people to build new ideas on the top of other's opinions and further discussing their experiences. After having chosen factors of both productivity and social productivity, a focus group study was conducted to investigate opinions of software management teams in a middle size software company. The discussion group was composed of nine personnel from the management team and the CEO of the company (total ten participants). As suggested by Krueger [17], the session was facilitated by one of the authors who commenced an introduction to encourage participants and initiate the discussion setting. We asked the management team about their opinion on productivity factors and one individual from the management team took written notes. A guide containing five questions and a preliminary model of social productivity was prepared for the focus group discussion: (1) What is your definition of productivity in software teams?, (2) What is your opinion of the factors that are affecting the productivity?, (3) What do you think of the most important factors among these ones for productivity?, (4) How would you describe the social factors of productivity?, (5) What is your opinion of the social factors that are affecting the productivity?

The goal of the focus group study was to identify the opinions from industry about the most important factors that are affecting for both productivity and social productivity. One of the participants defined productivity as *working faster*, while one other introduced the term efficient to this definition. Participants discussed social aspects of productivity including the impacts of social values over productivity, the communication frequency, coordination efficiency, team augmentation, task rotation. In addition, the group discussed the selected items from the software productivity literature; the impact of complexity or size of a software project, and re-usability of the created software artifacts. After having a debate on several factors affecting productivity, the group decided by voting that complexity of a project and re-usability of software artifacts are more important than some other factors, i.e. skills and reuse. In short, focus group activity provides us an opportunity to discuss our ideas about productivity factors in an industrial setting. We refined our list of factors by using the information provided in this session.

In light of these results, we designed a survey instrument to measure the impact of the factors on both productivity and social productivity. We used 5-point Likert scale (i.e. a psychometric scale frequently used in social research) for every factor and furthermore we add two question where they were asked to rank their opinions in descending order of importance for productivity and social productivity factors.

3.4 Software Productivity as Linear Structural Relation

A generally accepted measurement model of productivity is lacking [18], hence, we suggest that productivity and social productivity can be presented as latent variables showing themselves through a set of factors. In addition, we argue that these variables also influence each other. Based on the several important factors affecting both variables, we draw a model of social productivity by using SEM and aim to specify their interrelationships (see figure 2) for a conceptual representation of the hypothesized model). The observed variables are shown in rectangular boxes and the latent variables are shown in circular boxes. Moreover, the lines connecting the variables illustrate the direct effects of the indicators on the latent variables.

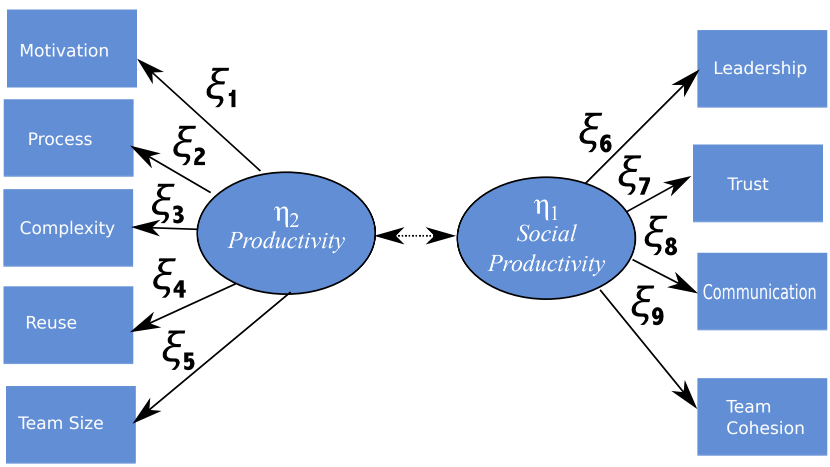


Fig. 2. Conceptual Model for Social Productivity of Software Development

For modeling social productivity and productivity factors, we use LISREL [19] (i.e. a software package frequently used for structural equation modeling) and proposed them as latent variables based on four and five observed variables respectively (nine indicators in total). The data was collected by surveys obtained from graduate and undergraduate university students. The analyses was conducted with two hundred and twenty-seven participants. About sixty-seven percent of the participants were post graduates. In this part of the work, we used a two-step approach, first we explored the measurement model which specifies the relationships between indicators and latent variables used. Secondly, using the results of the measurement model we test the structural equation model for an acceptable good fit. We suggest that all factors should be interacting with each other. The latent variables namely, social productivity and productivity, are bivariate correlated. The hypothesized model is presented in figure 2 where observed variables are depicted by rectangles and latent variables are illustrated

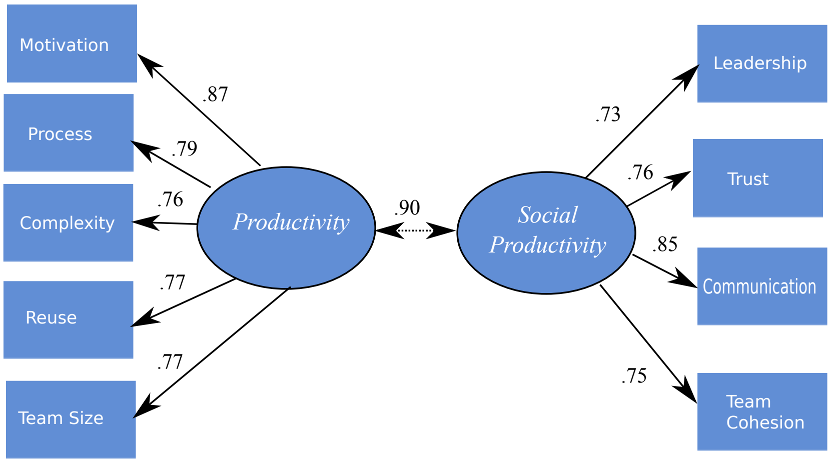


Fig. 3. Structural Equation Model for Social Productivity of Software Development

by circles, and further lines are used for portraying the relationships among the variables.

Two stages of data analyses were conducted to test the conceptual model depicted above. First, for testing the measurement model and second, for structural model. LISREL uses a maximum-likelihood method for fitting the mathematical model to collected data and for the estimation of model parameters. A chi-square test is used for observing the correlation between data and the model. Several type of indexes are used for investigating relationships between the model and the data including; (Goodness of Fit Index - GFI, a measure to fit model and covariance matrix), adjusted goodness of fit index (AGFI), root mean square error of approximation (RMSEA), normed fit index (NFI), comparative fit index (CFI) for assessing refined model relative to fit an independence model (i.e. null hypothesis which assumes variables among the relationships are uncorrelated). For GFI and CFI, value above .90 is acceptable. A null hypothesis (i.e. worst case scenario) is totally rejectable where $\chi^2(36, N = 227) = 3983.71, p < .001$.

Consequently, the measurement model (see figure 3) was found to differ a good fit for the data $\chi^2(26, N = 227) = 81.01, p < .001$, where $RMSEA = .081, GFI = .95, AGFI = .91, CFI = .99, NFI = .98$), where all of the structural correlations between observed and latent variables were statistically significant ($p < .001$) and ranged between *Motivation* (structural coefficient= .87, $p < .001$) and *Leadership* (structural coefficient= $.73, p < .001$). *Motivation* has the strongest relationship with productivity among all other factors, while *Communication* has the strongest connection between social productivity. It has been suggested that a chi-square difference test indicated significant improvement in fit between the independence model and the hypothesized model, $\Delta\chi^2(10, N = 227) = 3902.7, p < .001$).

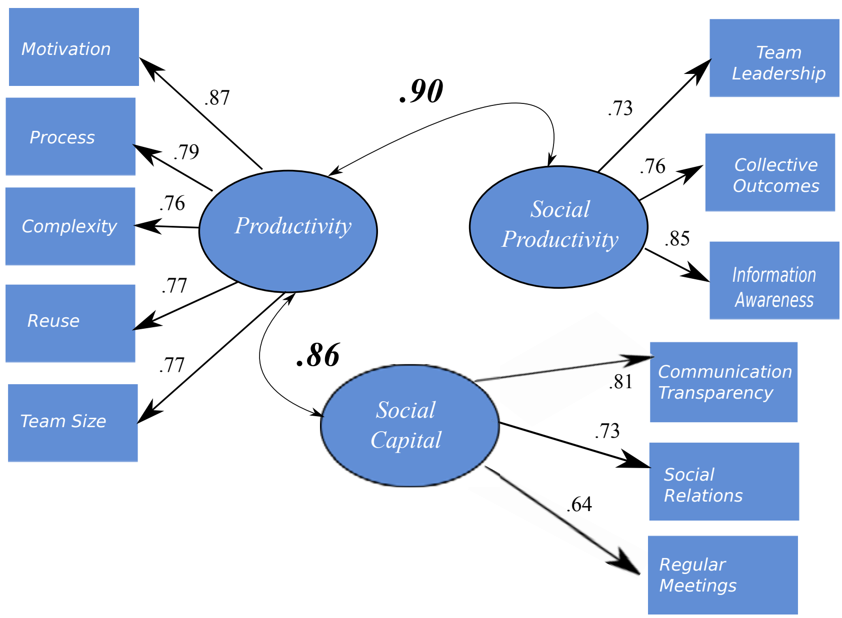


Fig. 4. Structural Model for Social Capital of Software Development

In the next step of the analysis, we refine our structural model to include social capital as an additional latent variable (see figure 4), and therefore we add new indicators affecting the social capital including; (i) communication transparency, (ii) social relations, (iii) frequency of meetings. The refined model was tested a good fit for the data, $\chi^2(41, N = 227) = 237.12, p < .001), RMSEA = .15, GFI = .84, AGFI = .75, CFI = .95, NFI = .95)$. Results of a chi-square difference test indicated that a significant improvement in fit, $\Delta\chi^2(14, N = 227) = 5237.82, p < .001)$. According to the path diagram, it seems team leadership, collective outcomes, and information awareness are significant predictors of social productivity improvement, respectively with structural coefficients $= (.73, .76, .85, p < .001)$. Moreover, social capital has three major ingredients which are communication transparency, social relations, regular meetings respectively with structural coefficients $= (.81, .73, .64, p < .001)$. It seems perfectly reasonable that both social capital and social productivity may be regarded as indicating high correlation with the productivity of a software process.

4 Conclusions and Future Work

In this paper, we propose an empirically validated model to measure the correlation between social productivity and productivity of software development. The evaluation of indicators are discussed with a focus group study by collaborating with the management team of a medium-sized software company. In addition, a survey instrument is created and tested with survey data collected from graduate and post graduate students (most of which have an industrial experience). Consequently, we constitute an initial model with 9 structural path relations. The goal of a structural equation model in this context supports the following outcomes. First, the observational results will provide insight into how different factors are affecting both productivity and social productivity. Second, by refining the first form, an improved model with 11 path relations is designed by using the notion of social capital.

We have only found a single study which investigates information system productivity based on structural equation modeling that may be related to our research. Based on participants from a Hong Kong information technology organizations, Foulds *et al.* [20] used structural equation modeling for developing and testing a framework for the productivity of large scale information system development. Their results show that better product descriptions and a dynamic approach to project management have a positive impact on system development productivity.

This study confirms that our approach should be useful for software productivity research for several reasons. First, we propose a linkage of structural equation modeling and SPI. In general, we suggest that, this approach can be useful for correlating latent (qualitative) variables and observable variables where empirical data can be collected. Consequently, the factors of interest can be revealed which aids managerial decision support. Second, we introduce the concept of social productivity and examined causal factors affecting productivity (e.g. leadership, team cohesion, collective outcome, trust) and identify their importance with respect to the opinion of our survey participants. Third, we introduce three variables to measure social capital of software development organizations (social relations, frequency of team meetings, communication transparency). Furthermore, we calculate several correlation values for factors investigated in both of our models.

SEM is a modeling method frequently used to solve several problems encountered in social sciences. Our first structural model indicates that there is a significant amount of correlation between productivity and social productivity, and the correlation of their interacting factors. Next, in the refined model of productivity, we introduced social capital as a new latent variable and formalized our second model based on these facts. In addition, by modeling various aspects of productivity using a structural model, a researcher can obtain clear insights into the factors that are affecting productivity. In light of this, our research makes a valuable contribution to the practice of software productivity improvement. Our next goal is to conduct the survey to evaluate our model on several software companies for comparison with our initial results.

Acknowledgments

This work is supported, in part, by Science Foundation Ireland grant number 03/CE2/I303-1 to Lero, the Irish Software Engineering Research Centre (www.lero.ie).

References

1. Fairley, R.: Managing and Leading Software Projects. Wiley-IEEE Computer Society Pr (2009)
2. Dittrich, Y., Floyd, C., Klischewski, R.: Social thinking-software practice. The MIT Press, Cambridge (2002)
3. Chemuturi, M.: Software Estimation Best Practices, Tools & Techniques: A Complete Guide for Software Project Estimators. J. Ross Publishing (2009)
4. Tangen, S.: Understanding the concept of productivity. In: Proceedings of the 7th Asia-Pacific Industrial Engineering and Management Systems Conference, Taipei, pp. 18–20 (2002)
5. Scacchi, W.: Understanding software productivity. In: Software Engineering and Knowledge Engineering: Trends for the Next Decade, pp. 273–316 (1995)
6. Abdel-Hamid, T.K.: The slippery path to productivity improvement. IEEE Softw. 13, 43–52 (1996)
7. Hamid, T.K.A., Madnick, S.E.: Lessons learned from modeling the dynamics of software development. Commun. ACM 32, 1426–1438 (1989)
8. Pfleeger, S.: Model of software effort and productivity. Information and Software Technology 33, 224–231 (1991)
9. Maxwell, K.: Applied statistics for software managers. Prentice Hall PTR, Englewood Cliffs (2002)
10. Boehm, B.: Improving software productivity. Computer 20, 43–57 (1987)
11. Lin, N.: Social capital: A theory of social structure and action. Cambridge Univ Pr, Cambridge (2002)
12. Bordieu, P.: The forms of capital. In: Handbook of Theory and Research for the Sociology of Education, vol. 241, pp. 241–258 (1986)
13. Portes, A.: Social capital: its origins and applications in modern sociology. Annual Review of Sociology 24, 1–24 (1998)
14. Coleman, J.: Social capital in the creation of human capital. The American Journal of Sociology 94, 95–120 (1988)
15. Hayduk, L.: Structural equation modeling with LISREL: Essentials and advances. Johns Hopkins Univ Pr, Baltimore (1987)
16. Kline, R.: Principles and practice of structural equation modeling. The Guilford Press, New York (2010)
17. Krueger, R., Casey, M.: Focus groups: A practical guide for applied research. Sage, Thousand Oaks (2009)
18. Jones, C.: Software Engineering Best Practices: Lessons from Successful Projects in the Top Companies. McGraw-Hill Osborne Media (2009)
19. Joreskog, K., Sorbom, D.: LISREL 8: users reference guide. Scientific Software International Inc., Lincolnwood (2001)
20. Foulds, L., Quaddus, M., West, M.: Structural equation modelling of large-scale information system application development productivity: the Hong Kong experience. In: 6th IEEE/ACIS International Conference on Computer and Information Science, ICIS 2007, pp. 724–731. IEEE, Los Alamitos (2007)

Agile Process Improvement: Diagnosis and Planning to Improve Teamwork

Mats Angermo Ringstad[1], Torgeir Dingsøyr[2,3], and Nils Brede Moe[2]

[1] Acando AS, NO-0159 Oslo, Norway
[2] SINTEF, NO-7465 Trondheim, Norway
[3] Dept. of Computer and Information Science, Norwegian University of Science and Technology, NO-7491 Trondheim, Norway

Abstract. Agile software development addresses software process improvement within teams. Process improvement, although a central concept in agile development, is still hard to achieve. This paper argues for the use of diagnosis and action planning to improve teamwork in agile software development. Diagnosis and action planning is illustrated in a small and immature team and in a large and more mature team. The action planning focused on improving shared leadership, team orientation and learning. The improvement project provided most new insight for the mature team.

Keywords: software process improvement, teamwork, agile software development.

1 Introduction

Software process improvement (SPI) [1] is an important part of all approaches to software development. In the plan-driven or traditional software development, the process improvement focus has mainly been on explicitly defining processes that can be standardized both within and across organizations [2]. SPI in this approach focuses on optimization. In agile software development [3], the goal of optimization is replaced by goals of high flexibility and responsiveness [4]. Subsequently, the agile perspective also changes the way of doing software process improvement. According to Salo and Abrahamsson [5] this requires new SPI mechanisms. Agile software development addresses software process improvement and management of software development practices within individual teams.

Given the focus on improving teamwork, there is a need for methods and techniques describing and diagnosing such teams. The research method action research [6] involves diagnosis and action planning, and fosters participative improvement. This method has further been suggested as a research method that can give results relevant to industry in addition to preserving scientific rigour. Our research question is: *How to efficiently improve teamwork in agile software development?*

The rest of this paper is organized as follows: First, we give an overview of theory on the topic of teamwork in agile software development teams. Further, we outline

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 167–178, 2011.

previous research on process improvement in this setting. Second, we describe the context of research, and how diagnosis and action planning was conducted, and continue to show results from this research during the diagnosis and action planning phases. Third, we discuss this way of organizing process improvement on agile software development and contrast it to previous work. Finally, we describe main conclusions and implications for theory and practice.

1.1 Characteristics of Agile Teams

To understand process improvement in agile software development, it is important to understand the nature of agile teams.

Agile development focuses on collaboration, informal communication and desire an organic organizational form [7]. Such organizations are characterized by being flexible, participative and encouraging cooperative social action.

Agile teams are usually co-located and arrange daily meetings, which means that the team-members can see what the others are working on and the tasks they are doing. Then team-members get immediate evidence of the progress of the work, can adjust their own work accordingly, and know who is responsible for which tasks [8]. This makes the work predictable and easier for the team to create a common understanding. Also the bottom-up approach of planning helps creating a common understanding [8]. Further, the agile team is supposed to be self-managed and empowered, which means that the team members are responsible for managing, monitoring and improving their own processes [9].

The literature on self-organizing and self-managing teams, claims that the decision authority and leadership needs to be shared [10, 11]. This means that leadership should be rotated to the person with the key knowledge, skills, and abilities for the particular issues facing the team at any given moment [12]. While the project manager should maintain the leadership for project management duties, team members should be allowed to lead when they possess the knowledge that needs to be shared or utilized during different phases of the project [13]. The jointly shared decision authority should replace the centralized decision structure where one person makes all the decisions and the decentralized decision structure where all team members make decisions regarding their work individually and independently of other team members [14].

For the team to be able to self-manage, it must have a degree of redundancy [11]. The members need multiple skills so that they are able to perform (parts of) each other's jobs and substitute each other as circumstances demand. In this respect, socio-technical literature is concerned with "multiskilling" [15]. Studies of self-managing teams also show that this kind of organization requires a capacity for learning that allows operating norms and rules to change in relation to transformations in the wider environment [11]. Therefore, to succeed with agile development, both team and organization needs to focus on improving the development processes.

1.2 Process Improvement

Software process improvement has its roots in general improvement philosophies like total quality management, which has been tailored to software engineering in the

Quality Improvement Paradigm (QIP) [16], and in efforts on standardisation like the Software Engineering Institute's Capability Maturity Model Integration (CMMI).

Because the field has been found to be rather dominated by the capability maturity model (CMM) [18] - now CMMI, we refer to this model when we explain what we mean by the "traditional approach" or "classical SPI". CMMI focus on software processes, standardisation and software metrics as a basis for improvement [18]. This focus on software process is based on the premises that:

- The process of producing and evolving software products can be defined, managed, measured, and progressively improved.
- The quality of a software product is largely governed by the quality of the development process [19].

This approach prescribes norms for how individuals, teams or organizations should operate, and for how processes should be standardized and improved [20].

There are several fundamental differences between traditional and agile software development regarding SPI[5]. First, while SPI in the plan driven perspective prescribes norms for how the individual, team and organization should operate, agile software development address the improvement and management of software development practices within individual teams [2]. In agile development, processes are not products, but rather practices that evolve dynamically with the team as it adapts to the particular circumstances [21]. Second, plan-driven methods, such as the waterfall model, usually adopt a top-down approach for improving the software development process [5], while the agile view has a bottom-up approach. Third, SPI in plan-driven development often emphasizes the continuous improvement of the organizational software process for future projects, while the principles of agile software development focus on iterative adaption and improvement in the on-going projects. Short development cycles provide continuous and rapid loops to iterative learning, to enhance the process and to pilot the improvement.

When doing agile development, there are typically two meetings where the team focuses on improving the process. 1) Daily meetings. In the daily meeting the team members are supposed to coordinate their work and focuses on solving problems that stop the team from working effectively. In Scrum, the Scrum-master is supposed to facilitate this meeting and making sure impediments to the process are removed 2) Retrospective [22]. At the end of each iteration, a retrospective is held. In this meeting the team focuses on what was working well and what needs to be improved. Measures are then taken.

While the conclusion of the study of Aaen et al. [23] is that there is no recognized SPI model supporting the agile approach, we found two such frameworks. Qumer and Henderson-Sellers [24], suggest a framework that can be used to create, modify or tailor situation-specific agile software processes. The model includes an agility measurement model and an agile adoption and improvement model. Salo and Abrahamsson [5] defined an iterative improvement process for conducting SPI within agile software development teams.

A more specific approach to improve teamwork is the use of the team radar by Moe et al. [25]. In the next section we will describe usage of this.

2 Research Context; Diagnosis and Action Planning

The research was conducted in two teams in different companies. The teams were selected to illustrate diverse starting points with respect to software process improvement (key information on the teams can be found in table 1 and table 2).

Table 1. Properties of the maintenance and development team

Context	"Maintenance"	"Development"
Type of system	Web-based	Back-end of large system
Technology	Primarily Java	C and C++
Project size	140.000 lines of code, and several, open-source modules	3.000.000 lines of code
Project phase	Maintenance and adding new functionality	New development
Project length	Started in 2008, handed over to customer fall of 2009.	Started in early 1990's, still on-going.
Team size	Five: One senior and four junior developers	Eight senior developers
Team composition	Almost eight months	Almost four months

The *maintenance team* was a small team doing maintenance and adding new functionality to a web-based enterprise system that is used by operators all over Norway. The team consisted of three junior developers, one service desk operator with some system and programming knowledge, and a senior developer. The team had worked together for almost eight months, located in one room.

The *development team* worked in a division of a large international corporation, adding new functionality on a large system that was over 20 years old. The team developed new functionality for administrating the software, server software, and low-level modules used by a graphical client. The company had used Scrum for more than two years. The Scrum master also worked on another development project. The team had eight team members (including the product owner) who were all senior developers with several years of software development experience. Three of the team members were external, hired from consulting companies, all working for more than two years on the system under study. The team members worked in individual offices.

Table 2. Agile practices in the two teams

Agile practice	"Maintenance"	"Development"
Iterative development	Yes	Yes
Continuous integration	Yes	No
Sprint planning	No	Yes
Sprint demo	No	Yes
Sprint retrospective	No	Yes
Daily standup	No	Yes
Self-managing team	Yes	Yes
Refactoring	Yes	Yes
Co-location	Yes	Yes
Pair-programming	2 people	No

The diagnosing means to identify the primary problems and underlying causes of the organizations desire to change [6]. In our case, the scope was limited to improving teamwork, and we used an instrument developed earlier, the team radar [25], with the factors listed in table 3. The team radar is based on a literature review and experience from case studies, which have identified the five dimensions of the instrument as playing a pivotal role in agile teamwork.

Table 3. Factors in the team radar diagnosis instrument

Factor	Description
Shared leadership	Leadership is rotated to the person with key knowledge, there is jointly shared decision authority.
Team orientation	Priority is given to team goals more than individual goals, team members respect other members' behaviour.
Redundancy	Members have multiple skills so that they can perform (parts of) each others tasks.
Learning	The team develops shared mental models, and a capacity for learning to allow operating norms and rules to change.
Autonomy	The ability to regulate the boundary conditions of the team, the influence on management (and other externals) on activity.

The diagnosing phase consisted of collecting a rich data material for analysis, through observation and semi-structured interviews. The interviews lasted on average 30 minutes, and were transcribed for analysis. The first author, observed teamwork practice in daily work, and meetings like daily meetings, iteration planning and retrospective. Field notes were taken from the observations and integrated with the interview material for analysis. In the maintenance team it was collected 4 interviews and 8 observations, and in the development team 6 interviews and 7 observations. In both teams there was a diagnosing period of two weeks each. In addition, the first author had discussions with some of the team members about the projects and work methods to gain a solid understanding of the surrounding environment.

The end-result of diagnosing was a score between zero and ten on selected team radar factors (See Figure 1). The score was given on the basis of the collected answers from all team members as well as the observed practice. In the next chapter, we show characteristic statements that form the basis of the score. Note that the diagnosing should not be seen as a precise instrument to diagnose teamwork, but the instrument enables both knowledge of important aspects and the development of a language for engaging with teamwork change and follow-up.

The action planning seeks to specify organizational actions that should relieve or improve the primary problems identified in the diagnosis [6]. In action research, the plan should be guided by a theoretical framework, in our case the theory of teamwork effectiveness underlying the team radar used in the diagnosis phase. The planning was organized as a presentation of the results of the diagnosing, with an open discussion on whether the team recognized how teamwork was portrayed in the findings. Then, we discussed which areas should be given priority to improve teamwork, and finally discussed concrete actions to form an action plan.

The scope of this article is to give a better understanding of the diagnosing and action planning phases focusing on teamwork in agile development. However, as a result of the two phases described below, a subsequent visit to the maintenance team showed that two of the suggested improvement actions, daily meetings and retrospectives, were re-implemented by the team. The development team made adjustments to their sprint planning based on the feedback. They focused on doing it more informally, using less time, and made it voluntary to attend.

3 Diagnosing Teamwork

To diagnose teamwork in the two teams, we used the team radar instrument to evaluate five aspects of teamwork. The total score on each factor is given in figure 1.

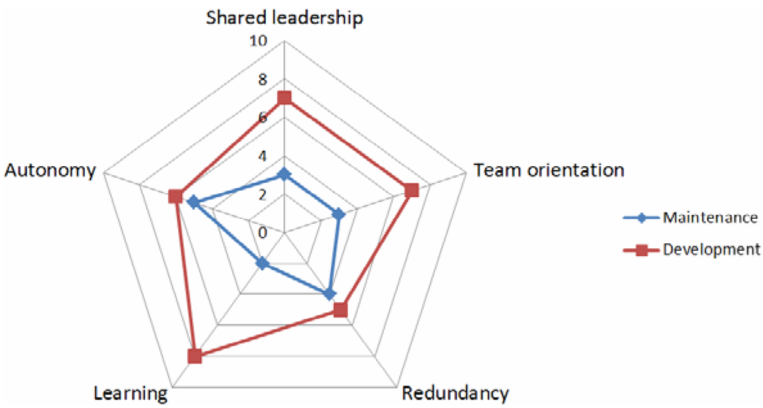


Fig. 1. A plot of teamwork characteristics of the two teams

The action planning phase involves a team-discussion to identify the right level for each factor, and the factors where both the company and researchers see a potential for improvement. Both teams chose shared leadership, team orientation and learning. As we see from figure 1, it is not necessarily the factors with the lowest score that are selected for action planning.

3.1 Shared Leadership

Shared leadership has a low score when the team-leader uses a “command and control” style of management, and when few take part in the decision-making process. A high score is given to teams which seek to engage everyone in leadership. Shared leadership implies that team members with knowledge about a certain area lead the discussions, and there is a shared decision-making process [25].

Maintenance team: The team members expressed that the team was well composed. When they felt they had knowledge about the issues discussed, the team members usually contributed to discussions and decision-making. The most important decisions, however, were made by the senior developer and the team leader in their weekly planning and status meetings. The reason for not involving the rest of the team in this meeting was the heavy workload on the rest of the developers. After these meetings, the senior developer reported back decisions, and what the team should prioritize. Some decisions, like how the customer wanted the support function organized were received negatively by the team.

Another reason for why some were not participating in the shared decision-making process was lack of knowledge. Missing knowledge resulted in some team members not being able or interested in discussing other's tasks. As one of the developers said, "We have a competency hole in the system, there are some components we don't know… and other components that only one person knows. But we have a future goal of having overlap regarding knowledge about the most important components."

With respect to the project goal, the team felt that the initial goal and release-plan was clear. However, during the first month the product had severe performance problems, and this resulted in the customer contacting the team every day with change orders. So instead of following the plan, the team focused on day-to-day work trying to solve the performance issues.

Development team: The team members were pleased with the team composition, and as one of the team members said, "we have a very strong team".

Decisions regarding work and who was supposed to solve which tasks were usually taken during daily stand-up meetings. Team members were free to pick whatever task they wanted, but sometimes the observation revealed that certain tasks were always solved by the same team members. This typically happened when one of the team-members were seen as an expert on the task.

The team members were active in discussions on topics where they felt they had enough knowledge to take part, this was evident during the sprint planning and the daily stand-ups observed. The team would discuss until they decided by consensus. We observed that the team being located in individual offices was a barrier to a shared decision-making process. One said, "It can be hard to go into another office and ask for opinions or help. Therefore, our best arenas for discussions and alternative proposals are the meetings we have".

3.2 Team Orientation

For team orientation, a low score is given when individual goals are more important than the team goals, and where team members do not respect other team member's decisions. The highest score is where the team goals are the most important, and when team members respect each other's decisions [25].

Maintenance team: Alternative proposals were not common for several reasons; the senior would often make the decisions for the team, specialization within selected components resulted in developers not discussing issues with "their" components with others, and because of a high workload the team never prioritized discussing alternative proposals. Missing a shared decision-making process resulted in individual goals becoming more important than team goals. During observation, we saw little communication between the team members in the team room, except when coordinating who should do what, and reporting status. As one of the junior developers said; "We have not had much communication lately since everyone has been so busy and overworked…. the task-assigning communication which happens quite often, is disturbing. " This situation clearly hindered team-orientation.

The team members did not show an interest in other team members work unless it was affecting their own work, and subsequently it was difficult to strengthen the importance of the team-goals. "The only person here who is interested in what the others are doing, is the senior", said one of the junior developers.

Development team: Team orientation was stronger in this team, and it was clear that alternative proposal from other team-members when planning work was appreciated. "We are very open when it comes to suggest alternative solutions", said one developer. A good example of shared leadership was during an observed sprint demo where one of the team members held the whole presentation, not the Scrum master. "We have a very professional orientation to how we work with the product and the projects", one of the team members said, pointing to the fact that they would usually have thorough discussions in the team before making decisions.

While team commitment was strong, the team members did not have a clear conception of the long-term vision of the project, even though they had clear goals for each sprint. The product owner, who got the full overview of the system, acknowledged this, realizing that he was not good enough at sharing the long-term goals with the team.

Some of the team members explained that they felt ownership to the team-plans, while others said they had ownership to the system being developed but not the project. This decreased the team-orientation.

3.3 Learning

The learning factor has the lowest score in situations where there are no feedback mechanisms. The highest score is given when there is continuous improvement of work methods based on feedback [25].

Maintenance team: Because the team stopped holding retrospectives, there were no formal arenas for learning and improving. The team members did not see the need for a common improvement and feedback meeting, since this meeting had not earlier resulted in an improved process. The team continued work in the same manner every day. The only feedback given to the team was from the weekly meetings where only the senior developer and the team leader participated.

Development team: The team had several arenas for giving feedback on other's work. The most appreciated one was the sprint retrospective. In addition, they had daily stand-up meetings and additional design meetings.

The team discussed process related problems in the sprint retrospective, which made it possible to adjust the Scrum process to make it better fit the organization, and the team. However, several of the interviewees said they were missing good discussions on how to improve the teamwork. Also we found that some process problems were not reported in the retrospectives. One example was problems related to the planning meeting. This meeting spanned over two days, and every user story was discussed in detail. Usually everyone participated in the discussions, however sometimes two or three team members could discuss a user story for a long period of time, while the others were only listening. Then team-members felt excluded from participating actively in the meeting, and subsequently the meeting was seen as less productive. “I am aware that our sprint planning is often ineffective, but I’m not sure how we can improve that”, the product owner said. This problem was not reported or discussed in the retrospective.

4 Action Planning: Measures to Improve Teamwork

To improve teamwork in the two teams, we presented the results of the diagnosis phase, and discussed priority on teamwork factors together with the teams. As a result concrete measures to be taken to improve the development processes and the teamwork were suggested.

For the maintenance team we observed challenges related to *shared leadership, team orientation, and learning*. As for leadership, the team was dominated by junior developers, there was little involvement of the team in leadership and little process in place. The team was heavily specialized, with team members working on independent modules, which again lowered team orientation. Finally, the team had no arenas for learning except for being in the same room, but observation showed little discussion and feedback on the actual work tasks the team members were involved in.

In a workshop, we presented the scores, problems and consequences to the team. The team decided to reintroduce important agile practices they had stopped doing. In prioritized order:

- Sprint retrospective to improve learning. Team members would be able to give feedback and improve both the development process as well as the product.
- Daily stand-up meetings to improve coordination of tasks, team communicating, and solve problems daily. The meeting was expected to have an effect on shared leadership, team orientation and learning.
- Code review to improve software quality, learning and increase redundancy.

The development team got higher scores on all factors compared to the maintenance team. The team prioritized to improve the problems with the highest potential for the team: inefficient sprint planning, variable ownership to project goals, and not solving process related problems in the retrospective. The following actions were suggested:

- Open space[1] sprint planning, to conduct sprint planning more efficiently. The sprint planning meetings in the team were dominated by specialists and long lasting. Using the open space process, the team members would suggest topics to discuss and then several discussions could happen in parallel in the same room. Team members are encouraged to walk between discussions. This action was expected to improve shared leadership and team orientation.
- Pair programming to improve team orientation. Making people to closely together constantly giving feedback could also improve shared decision-making and improve learning.
- Collocating the team in the same room, would improve communication and oversight, and improving team orientation.

[1] www.openspaceworld.org

5 Discussion

Now we return to our research question, "how to efficiently improve teamwork in agile software development?" We have shown results from using diagnosis with the team radar and action planning in a small and immature team and in a large and more mature team.

Both the teams perceived the diagnosing and the outcome as something they learned from, because it illuminated issues they had seen individually but not discussed within the team. It is not enough to do retrospectives if the team is not able to discuss the cause of the problems they are experiencing.

The cost associated with the improvement method reported in this article was perceived as low, with a short data collection period (interviews and observations), and little disturbance of the team. The feedback meeting where the team got concrete feedback and had the ability to discuss software process improvement measures, was the meeting taking most time. The teams stated that the radar produced a realistic and "spot on" analysis of the situation in the team. The method presented here, helped the companies improve, however, to use the team radar as a diagnosis instrument was not without challenges. Setting a score on the team radar was difficult, because the score is both subjective and imprecise. However, the main motivation for giving a score is to get a basis for discussion with the team. Also the score is discussed and verified by the team before an improvement program is suggested. Working with an instrument like the team radar should be seen as a start of a process, not as an end-mean in itself.

A question is then whether it would make more sense to have a more open approach to software process improvement, for example by basing improvement initiatives on the retrospective. There are two main differences in the approach reported in this article and an approach relying on retrospectives. First when using an external person, he or she gets more insight into the work of the team through interviews and observation. This might discover process related problems not reported in the retrospective, and give the team a better understanding of the problems. This is important to suggest the right measures to be taken. Second, since the team radar is based on the factors necessary for achieving self-management, the instrument gives more precision in identifying problems than what typically is identified in a retrospective. Redundancy for example, is a factor which is often mixed with learning, and a team might see problems but not relate them to root causes such as a lack of team orientation.

In the development team, as a larger and more mature team already experienced with process improvement, the diagnosis using a team radar led to more precise recommendations than they had experienced previously. In the maintenance team, one could argue that the results only confirmed what the team already knew. However it was not until the results from the team radar was discussed, that they were able improve their processes.

6 Conclusion and Further Work

This study indicates that process improvement, although a central concept in agile development is still hard to achieve. This study indicates that diagnosis using a specific instrument, the team radar, has an effect on action planning in teams.

This study has the following implications: The implication for theory is that there are positive indications that the team radar instrument identifies relevant challenges for agile software development teams. This form of diagnosing and action planning can be valuable in action research, and the diagnosis instrument can also be of use in case studies and ethnographic studies of teams.

The main implication for practice is that this study with two teams reveals that process improvement does not happen by itself even in agile methods, there needs to be effort invested to actively experiment with solutions.

Acknowledgements. We are grateful to participants in the two teams from the companies, who willingly shared their experience on teamwork and were willing to try out new practices. This project has been partially supported by the Research Council of Norway in the TeamIT project, through grant 193236/I40.

References

1. Aaen, I., Arent, J., Mathiassen, L., Ngwenyama, O.: A Conceptual MAP of Software Process Improvement. Scandinavian Journal of Information Systems 13, 81–101 (2001)
2. Lycett, M., Macredie, R.D., Patel, C., Paul, R.J.: Migrating agile methods to standardized development practice. Computer 36, 79–85 (2003)
3. Dybå, T., Dingsøyr, T.: Empirical Studies of Agile Software Development: A Systematic Review. Information and Software Technology 50, 833–859 (2008)
4. Nerur, S., Balijepally, V.: Theoretical Reflections on Agile Development Methodologies. Communications of the ACM 50, 79–83 (2007)
5. Salo, O., Abrahamsson, P.: An iterative improvement process for agile software development. Software Process: Improvement and Practice 12, 81–100 (2007)
6. Baskerville, R., Wood-Harper, A.T.: A critical perspective on action research as a method for information systems research. Journal of Information Technology 11, 235–246 (1996)
7. Nerur, S., Mahapatra, R., Mangalaraj, G.: Challenges of migrating to agile methodologies. Communications of the ACM 48, 72–78 (2005)
8. Okhuysen, G.A., Bechky, B.A.: Coordination in Organizations: An Integrative Perspective. Academy of Management Annals 3, 463–502 (2009)
9. Trist, E.: The evolution of socio-technical systems: a conceptual framework and an action research program, Ontario Quality of Working Life Centre, Toronto, Ontario (1981)
10. Kirkman, B.L., Rosen, B.: Beyond self-management: Antecedents and consequences of team empowerment. Academy of Management Journal 42, 58–74 (1999)
11. Morgan, G.: Images of Organizations. SAGE publications, Thousand Oaks (2006)
12. Pearce, C.L.: The future of leadership: Combining vertical and shared leadership to transform knowledge work. Academy of Management Executive 18, 47–57 (2004)
13. Hewitt, B., Walz, D.: Using Shared Leadership to Foster Knowledge Sharing in Information Systems Development Projects. In: Proceedings of the 38th Annual Hawaii International Conference on HICSS 2005 (2005)
14. Hoegl, M., Parboteeah, P.: Autonomy and teamwork in innovative projects. Human Resource Management 45, 67 (2006)
15. Emery, F., Thorsrud, E.: Democracy at work: the report of the Norwegian industrial democracy program. Martinus Nijhoff Social Sciences Division, Leiden (1976)
16. Basili, V.R.: Software development: a paradigm for the future. Presented at Computer Software and Applications Conference, COMPSAC 1989 (1989)

17. SEI, Capability Maturity Model ® Integration (CMMI) (2002)
18. Hansen, B., Rose, J., Tjørnehøj, G.: Prescription, description, reflection: the shape of the software process improvement field. International Journal of Information Management 24, 457–472 (2004)
19. Humphrey, W.S.: Managing the software process. Addison-Wesley, Reading (1989)
20. Hansen, B., Rose, J., Tjornehoj, G.: Prescription, description, reflection: the shape of the software process improvement field. International Journal of Information Management 24, 457–472 (2004)
21. Aaen, I.: Essence: Facilitating Agile Innovation. In: XP 2008, pp. 1–10. Springer, Heidelberg (2008)
22. Dingsøyr, T.: Postmortem reviews: Purpose and Approaches in Software Engineering. Information and Software Technology 47, 293–303 (2005)
23. Aaen, I., Börjesson, A., Mathiassen, L.: Navigating Software Process Improvement Projects. In: Baskerville, R., Mathiassen, L., Pries-Heje, J., DeGross, J. (eds.) Business Agility and Information Technology Diffusion. IFIP International Federation for Information Processing, vol. 180, pp. 53–71. Springer, Boston (2005)
24. Qumer, A., Henderson-Sellers, B.: A framework to support the evaluation, adoption and improvement of agile methods in practice. Journal of Systems and Software 81, 1899–1919 (2008)
25. Moe, N.B., Dingsøyr, T., Røyrvik, E.: Putting Agile Teamwork to the Test – A Preliminary Instrument for Empirically Assessing and Improving Agile Software Development. Presented at XP 2009, Pula, Italy (2009)

The Tutelkan Reference Process: A Reusable Process Model for Enabling SPI in Small Settings

Gonzalo Valdés, Marcello Visconti, and Hernán Astudillo

Departamento de Informática, Universidad Técnica Federico Santa María, Valparaíso, Chile
{gvaldes,visconti,hernan}@inf.utfsm.cl

Abstract. The adoption of international standards and models of process quality is difficult for small organizations due to several issues they face, such as inability to afford the associated costs and unawareness of SPI benefits. This article presents the *Tutelkan Reference Process* (TRP), a public software process that is conformant to CMMI-DEV v1.2, ISO 9001:2000 and Competisoft, and whose process assets can be reused as baseline for developing specific software process in small organizations. We present the methods we applied to evaluate standard-compliance of TRP, which are based on mapping techniques and methods used to appraise and audit organizations, and discuss how TRP is applied as part of an SPI framework oriented to small settings. When using TRP organizations become aware of their level of compliance with international standards, since each reusable asset contains information about the specific CMMI-DEV v1.2 practices, ISO 9001:2000 clauses and Competisoft activities it conforms to.

Keywords: software process improvement, process quality models, small settings, Tutelkan, CMMI-DEV v1.2, ISO 9001:2000, Competisoft.

1 Introduction

Software organizations implement best practices contained in standards and models of process quality, assessment and improvement (e.g., CMMI[1], ISO/IEC 15504[2], IDEAL [1]) in order to improve the quality of the software they produce, reduce production cycles, and use their resources more efficiently, among others [2] [3].

However, the adoption of international standards and models remains difficult for small organizations[3] due to several issues they face, such as lack of dedicated staff for process improvement, and inability to afford implementation, evaluation and certification costs [4] [6] [7] [8]. Therefore, small organizations often have a negative perception about models and standards, good practices are perceived as too expensive, time consuming, and over-targeted; furthermore, they find hard to relate standards with benefits and match them with their business needs, thus leading to unawareness of their actual benefits [4] [6].

[1] Capability Maturity Model Integration, http://www.sei.cmu.edu/cmmi
[2] Information technology - Process assessment.
[3] Companies or internal subunits with fewer than 50 employees (as used in [4] and [5]).

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 179–190, 2011.

Several initiatives worldwide have aimed at helping small organizations to implement software processes based on the most prominent models from ISO/IEC and SEI; mainly by (1) defining assessment methods tailored to their specific context, (2) recommending preselected subsets of processes, and/or (3) adapting and generating SPI models for small settings [4] [8] [9].

Another approach has been to develop new reference processes or standards, based on the prominent models. In Latin America, MoProSoft[4] combined and adapted to Mexican settings the recommended practices of the now-retired CMM v1.1, ISO 9001 and other specialized models such as PMBOK (Project Management Body of Knowledge); and in Brazil, the MPS.BR proposal [10] adapted several international standards to the Brazilian reality. Its main products are a reference process and an evaluation method which are conformant with ISO/IEC 15504 and ISO/IEC 12207, and compatible with CMMI. Later on, the Ibero-American project Competisoft [7] borrowed heavily from these models to develop a new region-wide reference improvement framework; indeed, its reference process may be thought of as a successor of MoProSoft.

This article briefly describes the *Tutelkan Reference Process* (TRP), a central element of the Tutelkan[5] SPI Framework (explained in detail in [11]), developed by a joint initiative[6] among Chilean universities and local software industry (companies and industry associations), aiming to facilitate the incorporation of internationally accepted process quality standards and models into small software organizations.

Unlike earlier initiatives, Tutelkan does not aim to define a new local software process standard or assessment method, but focuses on creating a mechanism to accelerate the implementation of any (acceptable) standard, through systematic reuse and composition of process assets that are already compliant with those standards. The framework provides a reference set of process assets (e.g., task and roles descriptions, workflow diagrams, templates, etc.) conformant with CMMI, ISO 9001 and Competisoft. Specifically, in this article, we described its design drivers, construction strategy, methodology by which we achieved compliance with the mentioned standards, and discuss how it helps in enabling small organizations to implement prominent international software process models and standards.

The article is organized as follows: section 2 describes the components of the Tutelkan SPI framework; section 3 presents de current version of TRP and its development strategy; section 4 specifies TRP's process pattern; section 5 maps of TRP and CMMI-DEV v1.2, section 6 maps TRP and ISO 9001:2000, and section 7 maps TRP and Competisoft; and section 9 summarizes and concludes.

2 The Tutelkan Components

Tutelkan aims to provide a framework to streamline and facilitate the adoption of prominent standards and models by small organizations. The main ideas to achieve such goal are: (1) providing a library of reusable process assets, (2) offering composition tools to describe small organizations processes using these assets, and (3)

---

[4] *Modelo de Procesos para la Industria del Software* (Process Model for Software Industry), http://www.comunidadmoprosoft.org.mx

[5] http://www.tutelkan.info

[6] Funded by CORFO (Chilean agency for entrepreneurship and innovation), www.corfo.cl

systematically training small organization focused consultants for these library and toolset. Tutelkan ultimately aims to build an informatics ecosystem of SPI tool builders and process asset maintainers (communities of experts, consultants and practitioners). Specifically, the framework includes the following elements:

- The *Tutelkan Reference Process (TRP):* defines a public software process whose process assets can be reused and modified by any organization to create its own specific software process. TRP includes practices proven by Chilean small software organizations, and is aligned with CMMI for Development v1.2 [12], ISO 9001:2000 [13] and Competisoft [7].
- The *Tutelkan Web Platform (TWP):* allows describing, accessing and composing process assets. Its service-based architecture allows building applications as "mashups" of content services; several applications have been built by internal and third party developers. It currently includes tools to explore and collaboratively build public process assets (TRP), maintenance of social content associated to them, and develop specific "private" processes for organizations (with restricted visibility). It also supports evaluations of private processes with quality models.
- The *Tutelkan Process Framework (TPF)*: a meta-model for software processes, which defines the valid types and relationships of process assets and processes. All software processes described in Tutelkan, both public and private, are described using TPF. It is based on OPF (Open Process Framework) and is compatible with SPEM (Software & Systems Process Engineering Meta-Model).
- The *Tutelkan Implementation Process (TIP):* a methodology that facilitates the implementation of software processes in small organizations. It specifies how to carry out an SPI project using Tutelkan's contents and tools.

Typical SPI projects include the phases of: *Evaluation and Diagnosis* against a standard or model, *Adaptation and Implementation* of best practices and process assets, and *Adoption and institutionalization* of the adapted practices and assets throughout the organization [11]. Tutelkan streamlines the adaptation/ implementation tasks since it makes available standard-compliant process assets ready for reuse (see Fig. 1 and it aids in the evaluation/diagnosis tasks since each reusable asset contains information about the specific CMMI practices, ISO 9001 clauses and Competisoft activities it conforms to; all of which is additionally supported by a set of Web tools (TWP) and a formalized methodology (TIP).

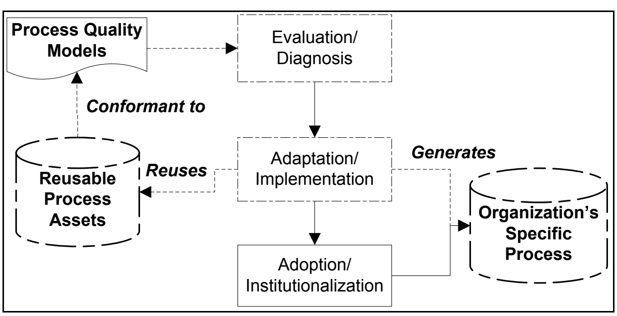


Fig. 1. Streamlined Tutelkan Processes

Several trials have being run in small Chilean software companies to evaluate the ease of application and usefulness of the Tutelkan framework and to determine the effort involved in applying it. For example, we have measured that a small company with an average number of 16 employees, invests approximately 1600 hours of total effort during 10 month to implement all CMMI ML (maturity level) 2 process areas with our framework. Additionally, Tutelkan has already been successfully piloted in conjunction with Competisoft in a Chilean small company [14] (more detail on Tutelkan validation can be found in [11]). All these experiences lead us to value as successful the main goal of the Tutelkan SPI framework, i.e., making accessible the implementation of prominent standards to small organizations.

3 The Development of TRP

TRP's main purpose is to serve as an actual exemplary software process, compliant to international standards, that organizations could use as baseline to initiate their SPI initiatives. TRP is publicly available in three ways: in the TWP platform, as printed documents, and as EPF (Eclipse Process Framework) plug-in downloadable from the Tutelkan website. The three versions are fully synchronized and have the same process content, this way TRP's process assets (e.g., task descriptions, workflows, templates, etc.) can be reused by any kind of organizations using the technological medium that best suits them.

3.1 Current Version

The current version of TRP (2011) is composed of 23 process areas grouped into seven categories, which in turn are grouped into four parts, as shown in Table 1.

Table 1. Components of TRP

Part	Category	Process Areas
Basic TRP	Project Management	Requirements Management, Project Planning, Project Monitoring and Control, Supplier Agreement Management, Measurement and Analysis, Product and Process Quality Management, Configuration Management
Advanced TRP	Software Development and Maintenance	Requirements Development, Analysis and Design, Programming, Testing, Installation
	Advanced Project Management	Risk Management, Formal Decision Evaluation, Integrated Project Management
	Process Management	Organizational Process Definition, Organizational Process Improvement, Organizational Training
High Maturity TRP	High Maturity Process Management	Organizational Process Performance, Organizational Innovation and Deployment, Causal Analysis and Resolution
	High Maturity Project Management	Quantitative Project Management
Business TRP	Business Management	Strategic Planning

Basic TRP contains recommended practices to make sure that the software projects of the organization are managed effectively. *Advanced TRP* is focused on assuring that the organization has a software development process that is defined, maintained and continuously improved. *High Maturity TRP* is concerned with incorporating quantitative analyses to process and project management. And *Business TRP* focuses on the strategic management capability of the organization.

High Maturity TRP is a superset of *Advanced TRP*, and the latter is a superset of *Basic TRP*. We recommend to organizations that are taking their first steps in the field of SPI to start addressing the process areas of *Basic TRP*, because they are related to establishing basic and effective project management practices, and allow the organization to start using best international software practices in a simple manner and with short-term results. *Business TRP* is the most independent part, but is recommended to be implemented in parallel whit *High Maturity TRP*.

3.2 Development Strategy

The TRP development process (started in 2007) took as baseline the software process of a Chilean small software company (Kepler Technology, www.kiteknology.com), who donated its process to the Tutelkan project. This company already had an ISO 9001 certification and was evaluated as CMMI v1.1 ML 2; hence it was reasonable to expect that its software process reflected a high degree of compliance to those models. The baseline process included software process best practices adapted to a small organization; and it was developed, field tested and continuously improved by its owner company through several software projects in Chile and Latin America.

Starting from this baseline process, several steps led to the current TRP version (see Table 1).

(1) A process pattern for TRP was defined, easy to understand and implement by small organizations, and compatible with the TPF process meta-model (see section 4).
(2) The baseline process was submitted to a normalization process by a group of experts, so the terminology used in the description of contents was understandable by any organization.
(3) A CMMI evaluation was carried out on this normalized process, to identify gaps in order to close them and make TRP compliant with all maturity levels of CMMI (see section 5).
(4) The ISO 9001 compliance of TRP was evaluated to identify and close any existing gap with respect to its requirements (see section 6).
(5) Finally, TRP was evaluated with Competisoft, to make it compliant with the most prominent regional process model (section 7).

In addition, during all these steps the succeeding versions of TRP were implemented in several small companies that were participating in the Tutelkan project, in order to get feedback from actual software organizations and projects.

4 TRP Process Pattern

TRP's process pattern is simple enough so small organizations can easily understand it and use it to implement their processes, and structured enough to provide the sufficient degree of formality required to implement software processes. Also the process pattern incarnates the TPF meta-model (compatible with SPEM and OPF). The idea is that the pattern allows us to easily incorporate any practice coming from any process in the future.

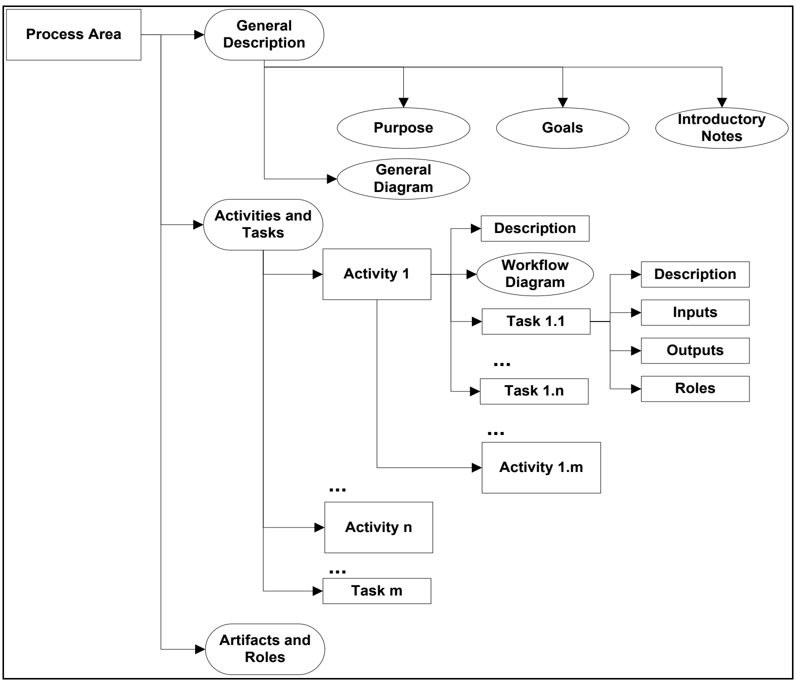


Fig. 2. TRP Process Pattern

The process pattern specifies the components of each TRP process area and their valid relationships. A process area contains the following elements (see Fig. 2):

- *General Description*: describes the expected outcome of the effective operation of the process area (*Purpose*); lists specific objectives aligned with the purpose (*Goals*); describes concepts important to understand the process area (*Intro. Notes*); and illustrates the relationships among its activities and tasks (*Gral. Diag.*)
- *Activities and Tasks*: detailed specification of:
    - *Activities*: logic groupings of *tasks* that have a common goal, they can also contain other *activities* (sub-activities). Each one contains a general narrative *Description* and a *Workflow diagram*.

- o *Tasks*: the atomic work unit, every *activity* is always composed -in the last level- of *tasks*. Each contains: a narrative text describing it (*Description*); the names of its input *artifacts* necessary to perform it (*Inputs*); the names of its output *artifacts* (*Outputs*); and the names of ones responsible for it (*Roles*).

- *Artifacts and Roles*: summary of *artifacts* and *roles* used in the process area and their relationships to each *activity* and *task*.

5 TRP and CMMI

Our goal was to evaluate whether TRP was conformant to CMMI (initially to CMMI v1.1, later to CMMI-DEV v1.2). To this end, we designed a methodology based on SCAMPI [15], which we adapted and simplified considering that the SCAMPI method is design to evaluate organizations, and our goal is to evaluate a process, i.e., instead of searching for evidence of the implementation of CMMI practices and goals in an organization's process assets, actual projects and staff, we should look for this evidence only on the process assets of the TRP.

Several aspects were analyzed to characterize the implementation of a CMMI practice in TRP:

- *Mapping*: extend to which a set of TRP tasks can be mapped to the CMMI practice.
- *Artifacts*: extend to which the set of artifacts associated to the previously mapped set of TRP tasks are compatible with the *typical work products* of the mapped CMMI practice.
- *Weaknesses*: extend to which the mapped set of TRP tasks and its set of artifacts contain deficiencies that might put at risk the proper operation of the mapped CMMI practice.

Using these attributes, we characterized each CMMI practice in TRP as either: *fully implemented* (FI) if there is a complete mapping, a compatible set of artifacts and no weaknesses; *largely implemented* (LI), same as FI but presenting some minor weaknesses; *partially implemented* (PI) if the mapping is incomplete and/or the set of artifacts is not fully compatible and/or there are weaknesses; and *not implemented* (NI) if there is no mapping.

Also, we rated each CMMI goal as *satisfied* (S) if all its practices are characterized as FI or LI, and *unsatisfied* (U) otherwise.

An example is shown in Table 2. TRP's tasks are mapped to specific practices (SP) of CMMI-DEV v1.2. The CMMI-DEV v1.2 practice *TS SP 2.1* is mapped to the tasks *T3.1* and *T3.2* of TRP, which altogether make that mapping to have an *overall rating* of FI. This method also allowed us to discover a gap, the practice *TS SP 2.4* was not covered by TRP. So we generated a new task (with its template artifacts, roles, etc.) that formalize a procedure to incorporate past experiences in order to decide if some components may be reused or purchased considering established criteria and a repository containing those past decisions.

Our baseline process already covered most of the CMMI-DEV v1.2 process areas of ML 2 and 3, and with this method we were able to close the remaining gaps.

In order to make the process conformant to ML 4 and 5 we had to incorporate many new process assets and extend many of the existing ones. We convoked the local community to collaborate and we reused, adapted and normalized material from many local companies. The final result is the mapping presented in Fig. 3, where TRP is mapped to the process areas and specific goals (SG) of CMMI-DEV v1.2.

Finally, TRP was appraised by an external party, a SEI partner (www.procesix.com), who validated TRP compliance with CMMI-DEV v1.2.

Table 2. Extract of the Mapping between TRP and CMMI

	CMMI TS - Technical Solution	**SG 1 Select Product Component Solutions**	…	**SG 2 Develop the Design**	SP 2.1 Design the Product or Product Component	…	SP 2.3 Design Interfaces Using Criteria	SP 2.4 Perform Make, Buy, or Reuse Analyses
Overall Rating				**U**	**FI**		**FI**	**NI**
TRP - Analysis and Design								
A1 Effort Estimation								
…								
A3 Design								
T3.1 Define Patterns					**X**			
T3.2 Design Components and Programs					**X**			
T3.3 Design Interfaces and Communications							**X**	

6 TRP and ISO 9001

Our goal was to evaluate if TRP was conformant to the ISO 9001:2000 standard. ISO 9001:2000 audits are designed to be performed on organizations [16], not on a set of interrelated process assets; and as we needed a concrete and precise guideline to assess if TRP was conformant to ISO 9001:2000, we decided to take advantage of the work we already did and devised a triangulation scheme in order to obtain a precise evaluation.

Specifically, besides the guidance provided in [17], we used the TRP-CMMI-DEV v1.2 mapping we already had (see section 5) and a mapping between CMMI practices and ISO 9001 requirements developed by Mutafelija and Stromberg [18]. This allowed inferring a concrete mapping between TRP and ISO 9001:2000, shown in the left side of Fig. 4.

They also defined the following values for their CMMI-ISO mappings [18]: *strong* (S) if there is a strong relationship between the ISO requirement and the CMMI practice; *medium* (M) if the match is not complete, but with some interpretation, CMMI may satisfy the ISO requirement; *weak* (W) if the statement in the ISO standard does not clearly correspond to the CMMI practice; and *not map* (N).

The right side of Fig. 4 shows an example of the triangulation scheme. Suppose we want to know if TRP satisfies the sub-clause *7.3.2 a)* of ISO 9001:2000 (concerned with functional and performance requirements), we first find its mapping to CMMI, according to [18] this clause is strongly related to the CMMI-DEV v1.2 specific practice *RD SP 2.1*. Then we look our TRP-CMMI mapping and find out that this specific practice is characterize as fully implemented in the TRP tasks *T3.1* and *T3.2* of the process area *Requirements Development*. Lastly we can infer that these TRP tasks must be related to the ISO 9001 clause.

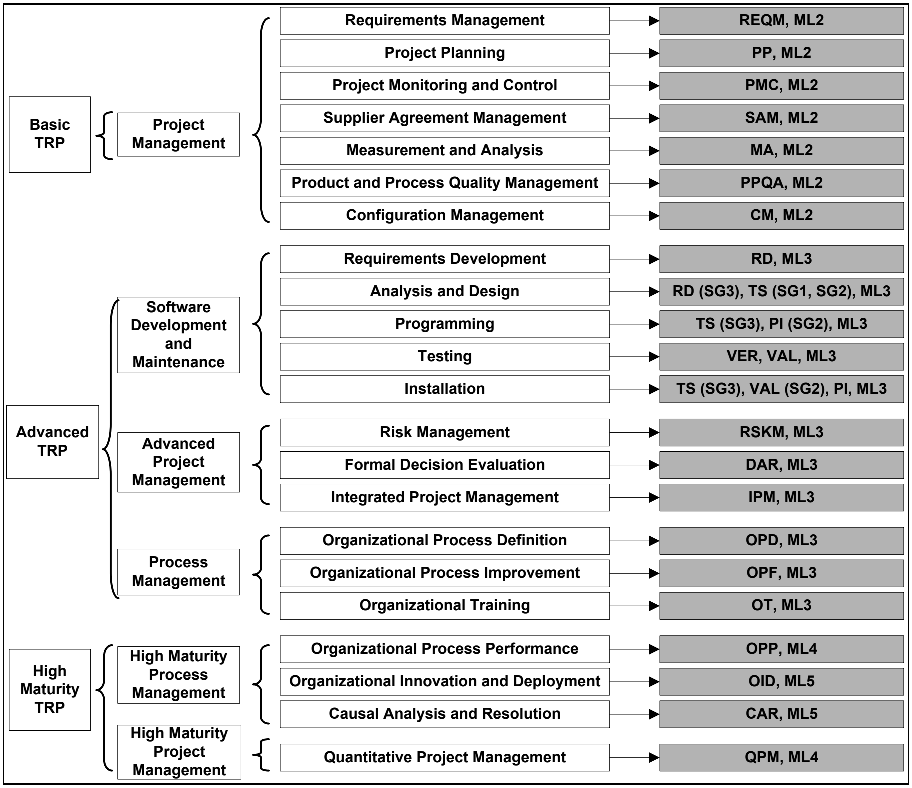


Fig. 3. High-level Mapping between TRP and CMMI

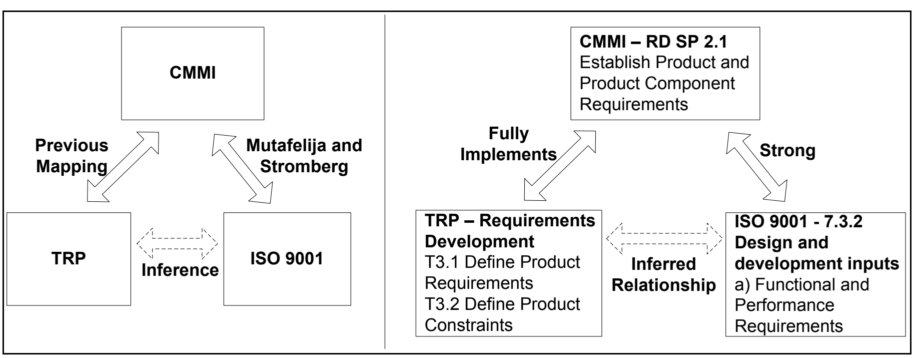


Fig. 4. Triangulation Scheme TRP, CMMI and ISO 9001

Then, with this guidance at hand we evaluated whether the mapped TRP tasks actually satisfy the mapped ISO requirement (thoroughly examining all related process assets), and we qualify the mapping. For example in Table 3 we can see that the mapping between the sub-clause *7.3.2 a)* and tasks *T3.1* and *T3.2* is qualified as strong (S) (we used the same scheme of [18]), because these tasks and their associated artifacts completely satisfy what is stated in the ISO requirement.

Table 3. Extract of the Mapping between TRP and ISO 9001

	ISO 9001 - 7 Product realization	…	**7.3.2 Design and development inputs**	…	a) Functional and performance requirements	…	c) Similar designs	d) Other requirements
Overall Rating					**S**		**N**	**S**
TRP - Requirements Development								
A1 Problem Analysis								
T1.1 Perform Preliminary Analysis of the Problem					**W**			
…								
A2 Operational Environment Specification								
T2.1 Define the Environment from Technical Requirements								**M**
…								
A3 Requirements Specification								
T3.1 Define Product Requirements					**S**			**M**
T3.2 Define Product Constraints								
T3.3 Identify Interface Requirements								**M**

Finally, we did a general evaluation to determine the *overall rating*. If there is at least one mapping that is qualified as strong, then the overall rating is strong. However there may be cases where many non-strong mappings lead a strong overall rating anyway, as those maps may complement themselves in order to satisfy the ISO requirement. This is the case of sub-clause *7.3.2 d)*, there are three mappings qualified as medium (M), because on their own none of them completely satisfies the sub-clause, but when taken together they do. Finally this method allowed us to discover gaps to close. For example, sub-clause *7.3.2 c)* was not covered by TRP at that time.

7 TRP and Competisoft

Our goal was to check if TRP was compliant with the Competisoft model. We mapped TRP tasks to Competisoft activities, and rated if those mapping as *satisfied* (S) and *not satisfied* (NS). Competisoft has 9 main processes which are grouped into three categories, namely: *Top Management*, *Middle Management* and *Operations*. TRP was compliant with all the process on the two latter categories, with some minor gaps that we were able to close. MoProSoft (the baseline process of Competisoft) includes many practices of CMMI and ISO 9001, so this was somehow expected.

For example, in Table 4 the activity *A1.3* of the Competisoft process *Human Resources Management* is satisfied by the TRP tasks *T1.2* and *T5* of the process area *Organizational Training*. A minor gap we found is that we did not have considered carrying out work environment surveys, as suggested by activity *A1.9* of Competisoft.

However, we identified a huge gap with respect to the category *Top Management*, which has only one process, called *Business Management*. Specifically, TRP did not have any practices related to strategic planning of the organization, such as definition of organizational mission, vision and values and SWOT (Strengths, Weaknesses, Opportunities, and Threats) analyses. Therefore we added an all new process area in TRP called *Strategic Planning* to include these aspects (the last one listed in Table 1).

Table 4. Extract of the Mapping Between TRP and Competisoft.

	Competisoft - Human Resources Management	**A1 Preparation**	…	A1.3 Elaborate or Update the Training Plan	…	A1.9 Prepare or Update the Form for the Survey of Work Environment
Overall Rating				**S**		**NS**
TRP - Organizational Training						
A1 Coordinate Activities for Planning						
…						
T1.2 Define Criteria and Staff Evaluation Schemes				**X**		
…						
T5 Create Training Plan				**X**		

8 Conclusions and Future Work

Tutelkan is a joint effort of Chilean academia, software industry and government, whose general objective is to create a sustainable mechanism to enable small software organizations to incorporate best international practices of prominent models and standards or process quality. One of the project's main products is the public reference process TRP, whose process assets can be reused as baseline for developing specific software process in small organizations. The Tutelkan framework and TRP has been successfully piloted with several Chilean small organizations.

We presented the methods we applied to evaluate the standard-compliance of TRP, which are based on mapping techniques and methods used to appraise and audit organizations. When using TRP organizations become aware of their level of compliance with international standards, since each reusable asset contains information about the specific CMMI practices, ISO 9001 clauses and Competisoft activities it conforms to.

Future work is intended to update our current mappings and review compliance to the newer versions of CMMI-DEV (v1.3) and ISO 9001 (2008), and to include the new ISO/IEC 29110 standard (SW Eng. - Lifecycle Profiles for Very Small Entities).

Acknowledgments. This work has been partly funded by project Fondef D09I1171 of Conicyt, Chile. The authors would like to thank Carolina Noriel, Claudio Herrera, Felipe Rodríguez and Ricardo Moreno for their work in the project.

References

1. McFeeley, B.: IDEAL: A User's Guide for Software Process Improvement. Handbook CMU/SEI-96-HB-001. Carnegie Mellon University, Pittsburgh (1996)
2. Gibson, D.L., Goldenson, D.R., Kost, K.: Performance Results of CMMI-Based Process Improvement, Technical Report CMU/SEI-2006-TR-004. Carnegie Mellon University, Pittsburgh, PA (2006)
3. Staples, M., Niazi, M.: Systematic Review of Organizational Motivations for Adopting CMM-based SPI. Information and Software Technology 50, 605–620 (2008)
4. Richardson, I., von Wangenheim, C.G.: Why are Small Software Organizations Different? IEEE Software 24(1), 18–22 (2007)
5. European Commission, New SME Definition: User Guide and Model Declaration (2005), `http://ec.europa.eu/enterprise/policies/sme/files/sme_definition/sme_user_guide_en.pdf`
6. Laporte, C.Y., Alexandre, S., O'Connor, R.V.: A Software Engineering Lifecycle Standard for Very Small Enterprises. In: O'Connor, R.V., Baddoo, N., Smolander, K., Messnarz, R. (eds.) Software Process Improvement, EuroSPI 2008. CCIS, vol. 16, pp. 129–141. Springer, Berlin (2008)
7. Oktaba, H., Garcia, F., Piattini, M., Ruiz, F., Pino, F., Alquicira, C.: Software Process Improvement: The Competisoft Project. IEEE Computer 40(10), 21–28 (2007)
8. Pino, F.J., Garcia, F., Piattini, M.: Software Process Improvement in Small and Medium Software Enterprises: a Systematic Review. Software Quality Journal 16, 237–261 (2008)
9. Pino, F.J., Garcia, F., Piattini, M.: Key Processes to Start Software Process Improvement in Small Companies. In: Proceedings of the 2009 ACM Symposium on Applied Computing, pp. 509–516. ACM, New York (2009)
10. Weber, K., Araújo, E., Rocha, A., Machado, C., Scalet, D., Salviano, C.: Brazilian software process reference model and assessment method. In: Yolum, P., Güngör, T., Gürgen, F., Özturan, C. (eds.) ISCIS 2005. LNCS, vol. 3733, pp. 402–411. Springer, Heidelberg (2005)
11. Valdes, G., Astudillo, H., Visconti, M., López, C.: The Tutelkan SPI Framework for Small Settings: A Methodology Transfer Vehicle. In: Riel, A., O'Connor, R., Tichkiewitch, S., Messnarz, R. (eds.) EuroSPI 2010. CCIS, vol. 99, pp. 142–152. Springer, Heidelberg (2010)
12. Software Engineering Institute: CMMI for Development, Version 1.2 (2006)
13. International Organization for Standardization: ISO 9001 - Quality management systems - Requirements (2000)
14. Villarroel, R., Gómez, Y., Gajardo, R., Rodríguez, O.: Implementation of an Improvement Cycle using the Competisoft Methodological Framework and the Tutelkan Platform. CLEI Electronic Journal 13(1), paper 2 (2010)
15. Software Engineering Institute: Standard CMMI Appraisal Method for Process Improvement (SCAMPI) A, Version 1.2: Method Definition Document (2006)
16. International Organization for Standardization: ISO 19011 - Guidelines for quality and/or environmental management systems auditing (2002)
17. International Organization for Standardization and International Electrotechnical Commission: ISO/IEC 90003 - Software engineering - Guidelines for the application of ISO 9001:2000 to computer software (2004)
18. Mutafelija, B., Stromberg, H.: Process Improvement with CMMI v1.2 and ISO Standards. Auerbach Publications, Boca Raton (2008)

Process Support for Product Line Application Engineering

Padraig O'Leary[1] and Ita Richardson[2]

[1] RiSE – Reuse in Software Engineering and the Federal University of Bahia, Brazil
[2] Lero – the Irish Software Engineering Research Centre, University of Limerick, Ireland
Padraig.oleary@rise.com.br, Ita.richardson@lero.ie

Abstract. The derivation of products from a software product line is a time-consuming and expensive activity. Despite recognition that an effective process could alleviate many of the difficulties associated with product derivation, existing approaches have different scope, emphasize different aspects of the derivation process and are frequently too specialized to serve as a general solution. In response to a need for methodological support, we developed Pro-PD (Process model for Product Derivation). Pro-PD was iteratively developed and evaluated through four research stages involving academic and industrial sources. This paper illustrates how Pro-PD provides systematic support by using product derivation preparation as an example.

Keywords: Software product lines, product derivation, process model.

1 Introduction

A Software Product Line (SPL) is a set of software-intensive systems that share a common, managed set of features satisfying the specific needs of a particular market segment or mission and are developed from a common set of core assets in a prescribed way [1]. The SPL approach makes a distinction between domain engineering, where a common platform for a number of products is designed and implemented, and application engineering, where a product is derived based on the platform components [2]. The separation into domain engineering and application engineering allows the development of software artefacts which are shared among the products within that domain. It is during application engineering that the individual products using the platform artefacts within a product line are constructed. The process of creating these individual products is known as product derivation.

A number of publications speak of the difficulties associated with the process. Hotz et al. [2] describe it as "slow and error prone, even if no new development is involved". Griss [3] identifies the inherent complexity and the coordination required in the derivation process by stating that "…as a product is defined by selecting a group of features, a carefully coordinated and complicated mixture of parts of different components are involved". Therefore, the derivation of individual products from shared software assets is still a time-consuming and expensive activity in many organisations [4]. Rabiser et al. [5] enforces this point when they claim that "guidance and support are needed to increase efficiency and to deal with the complexity of

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 191–202, 2011.

product derivation". Furthermore there "is a lack of methodological support for application engineering and, consequently, organisations fail to exploit the full benefits of software product families."

Due to this lack of methodological support for product derivation, the authors identified the following research objective: *To define a systematic process which will provide a structured approach to the derivation of products from a software product line based on a set of tasks, roles and artefacts.* To meet this objective, we developed Pro-PD: Process model for Product Derivation. Pro-PD was iteratively developed and evaluated through four research stages involving academic and industrial sources.

In this paper, we will focus on the development and description of how Pro-PD provides systematic support for the initial preparatory activities of product derivation. We focus on the product derivation preparation activities for two reasons. Firstly, due to space restrictions a full description of both Pro-PD and its development would be impossible. Secondly, research has demonstrated that preparing for derivation is an important activity and has to be at least closely related to product derivation [6]. We noted that a lack of support for preparing derivation is one of the main reasons that product derivation often fails in practice [7]. Furthermore through our research we observed that the task of initiating a derivation project has been overlooked by SPL research. Consequently, existing approaches to product derivation offer only partial support.

The remainder of this paper is structured as follows: section 2 discusses existing approaches to product derivation. Section 3 presents the research design. In Section 4 describes the Pro-PD approach taken to meet the defined research objective. In section 5, an overview of Pro-PD, and its support for product derivation preparation is presented. Section 7 presents our final conclusions.

2 Background

A number of models have been developed to support software product line development within organisations. These include PuLSE, FAST, ConIPF, DOPLER^{UCon} and the SEI Product Line Practice Framework.

PuLSE (**Pro**d**u**ct **L**ine **S**oftware **E**ngineering) [8] is a method engineering framework consisting of three major elements: Deployment Phases, Support Components and Technical Components. PuLSE-I activities include planning product derivation. However, the approach defines roles and tasks on a very high-level. According to Atkinson et al. [9] where a formalised process did not exist, the introduction of PuLSE in industry turned out to be problematic. The FAST application engineering process [10] greatly simplified product derivation by describing the products in the application modelling language. However, to enable automatic product derivation, system specifications must be precisely defined and specified.

A product derivation framework presented by Deelstra *et al.* [4] was developed based on two industrial case studies. This work by Deelstra *et al.* provides a framework of terminology and concepts for product derivation. However, there is no support for the early phases of product derivation or product specific development and testing. The framework focuses on product configuration and is only a high-level attempt at providing the methodological support that Deelstra *et al.* [4] and others [11-13] agree is required for product derivation.

DOPLER^{UCon} [14] is a tool-supported approach for product configuration with capabilities for adapting and augmenting variability models to guide sales people and application engineers through product derivation. DOPLER^{UCon} is focused on providing user-centred tool support for product derivation, rather than supporting the product derivation process within the approach.

The SEI Product Line Practice Framework (PLPF) [1] defines 29 software product line practical areas. However the framework is generic and does not define process support. There is a strong focus on planning product derivation with the ultimate goal to automate the derivation process.

2.1 Limitations of Current Approaches

Existing approaches and methods have very different scope and emphasise different aspects of the derivation process. Others such as FIDJI [15] (not discussed), capture only a small part of the process while others, like PuLSE-I are much broader. All of them come with different amounts of prescription and tool support. Some describe a generic process rather vaguely and others are very close to practise and prescriptive in the definition of their process steps. In particular, we identified the limitations of current approaches as a lack of:

- Lack of defined flow of artefacts
- Lack of defined roles and responsibilities
- Lack of process support

Lack of defined flow of artefacts. Product development within a SPL requires a high degree of coordination and communication. Frequently both customer-specific and platform development occur in parallel. There is a need for awareness of the artefacts and stakeholders involved in product derivation. A good starting point could be PuLSE-I [13], as it names the development items in a descriptive manner. However, it does not provide detailed description of artefacts usage within the process.

Lack of defined roles and responsibilities. Diverse people with diverse tasks, roles, and responsibilities are involved in product derivation. Current approaches do not provide sufficient support for the managing of roles and assignment of roles to tasks and artefacts within the product derivation process. FAST [10] assigns activities to one of the three defined derivation roles but this is done at a very high level and unusable in any practical setting.

Lack of process support. A well-defined process can be managed, measured and observed, and therefore improved. An emphasis on processes helps software development to become more like engineering, with predictable time and effort constraints, and less like art [16]. Clements and Northrop explain the fundamental need for documented processes within SPL [1] as follows: Defined processes set the bounds for each person’s roles and responsibilities so that the collaboration is a successful and efficient one.

3 Research Design

The goal of our research is to provide an evidence based process approach for product derivation. With this in mind, our research design was influenced by Ahlemann et al. [17] which focused on empirically grounded and valid process model construction. In an analogy with systems engineering, the overall construction process is based on a cyclic structure to allow for model corrections on preceding construction stages via feedback-loops. Although the stages are dealt with sequentially, they contain cyclic sub-processes. The research design is compatible with common suggestions for qualitative research designs in process models [18]. Stages 1 and 2 are the primary construction steps. Stage 3 is both a development and an evaluation step. Stage 4 is purely an evaluation step. An overview of the research design is presented in Figure 1.

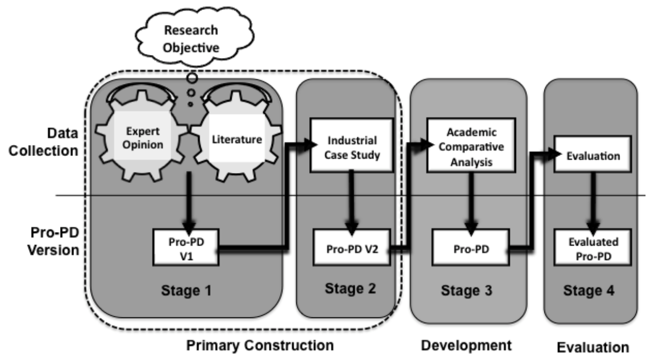


Fig. 1. Overview of Research Design

Stage 1, core construction, entailed a literature review from which a preliminary version of the model was developed. The literature review aimed to identify the fundamental practices of product derivation, through studying existing identified product derivation practices. Concurrent to the literature review, a series of iterative expert opinion workshops was organised. Participation by expert users in the core construction stage is emphasised by Rosemann and Schütte [19] and Schlagheck [20], as the users are the subject-matter experts of the problem domain. Furthermore, as the research is designed for use in both industry and academia, the selection of experts should reflect this. With this in mind, the selected participants were two academic SPL experts with 20 years experience, an industrial SPL expert with 10 years experience and a software process improvement expert.

Participants met twice per month for six months. At each workshop the model was presented to the experts and was evaluated using formal questions on model structure. The model was discussed amongst the group until a consensus was formed and the model was revised. After each workshop we returned to the literature and based upon the expert revisions and secondary research, iteratively developed Pro-PD V1.

Stage 2 was an industrial case study within Robert Bosch GmbH. This was carried out as an inductive, empirical validation [17]. We chose a case study as they are often considered to be the optimal approach for researching practice based problems, where the aim is to represent the case authentically "in its own terms" [21]. Pro-PD V1 was mapped and compared to product derivation within the company. Robert Bosch GmbH was chosen for the case study because previous SPL efforts have been judged a success by their peers [22]. The case study was carried out in conjunction with the corporate research division. The case study was dual-purpose. In the first instance, we modelled the Bosch product derivation process for their internal use and then we updated Pro-PD V1 based on our observations.

In conducting the case study, we analysed internal company documentation, which illustrated the existing process through completed projects. We then organised an onsite visit including a two-day workshop with the corporate research division of Robert Bosch GmbH. Attendees included selected product architects and developers from product line business units within the company. The primary researcher (O'Leary) was accompanied by two other researchers, one of whom had published extensively on case study research. After the workshop a technical report on the company's product derivation process was created and validated through feedback with Bosch SPL experts. Both the documentation analysis and the workshop output were used to identify what components should be included in Pro-PD V2.

Stage 3 of the research, an academic comparative analysis, was carried out during a research collaboration with JKU (Johannes Kepler University Linz, Austria). JKU had previously developed the DOPLERP^{UCon} (**D**ecision-**O**riented **P**roduct **L**ine **E**ngineering for effective **R**euse: **U**ser-centered **Con**figuration) approach. Based on initial discussions and existing documentation of our two approaches, a high-level mapping was created. This was done in a distributed manner using spreadsheets to visualize commonalities and differences between the two approaches. Using this mapping, the authors of this paper met to analyse the first results, discuss open issues, and detail the comparison. We then conducted several telephone conferences with JKU researchers to work on the details of the comparison. Pro-PD was compared to the activities identified by DOPLER for Siemens VAI. Based on this comparison the final version of the model, Pro-PD, was developed.

Pro-PD was evaluated during stage 4 of the research in two steps. The first was an inter-model evaluation with the SEI PLPF during which Pro-PD was reverse engineered and compared to the PLPF. According to Ahlemann et al. [17] process models that are compatible with such standards and norms can be are regarded as high quality.

Then, we systematically evaluated Pro-PD by analyzing support for its activities in three independently developed, published and highly-cited approaches: COVAMOF [23], FAST [10], and PuLSE-I [13]. The approaches have been developed with different goals, for different purposes, and in different domains. Furthermore, in our literature review we identified that these three approaches were influential through their frequent citations.

Although a framework for evaluating product derivation approaches does not exist, we adapted a framework[1] developed for the purpose of evaluating software product line architecture design methods [24]. We adapted the questions regarding the category context proposed by Matinlassi [24] from "product line architecture design method" to "product derivation approach". We adopted only one element for the category user (target group) as our focus is on evaluating the contents (support for key activities) and not the user support. For the category contents, we adopted the first two elements activities and artefacts. This evaluation was subsequently published [25].

---

[1] This work was a result of a collaboration with Dr. Rick Rabiser.

4 Approach

In order to achieve the objective defined above, this research has developed the Pro-PD process for product derivation. Pro-PD is a process reference model for product derivation that is minimal, complete, and adaptable:

- Minimal – only content that is seen as essential for product derivation is included
- Complete – it can be manifested as an entire process to build a system
- Adaptable – it can be adapted to different process types

Pro-PD is a minimally complete process reference model for product derivation. This means that only fundamental product derivation process content is included. Domain and discipline specific content is not included in Pro-PD and Pro-PD is independent of the methods and techniques used to derive a product. Pro-PD focuses instead on the essential tasks, roles and artifacts used to derive products from a software product line.

Pro-PD is adaptable, it can be used as a foundation from which company specific product derivation process content can be developed. The process structure is based on the waterfall model; however, to demonstrate its flexibility, it is adapted to fit the characteristics of an iterative process model.

5 Pro-PD

Pro-PD focuses on the activities, roles and work artefacts used to derive products from a software product line, these elements represent the process building blocks of Pro-PD. Roles represent a set of related skills and responsibilities. Work products are artefacts that are produced, modified or used by tasks. Tasks are assignable units of work that usually consume or produce one or more products. Phases are collections of related tasks that share common goals and allow the process be presented at a high level. Figure 1 gives on overview of these Pro-PD activities and the iterative nature of the Pro-PD process.

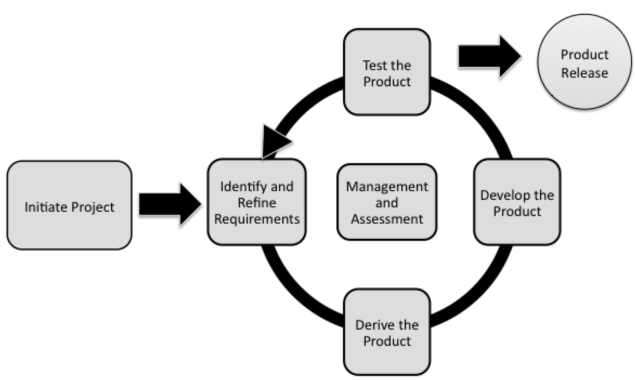


Fig. 2. Overview of Pro-PD Activities

5.1 Units of Work: Activities and Tasks

Pro-PD contains the following activities:

- **Initiate Project** - the preparatory tasks required to establish a product derivation project.
- **Identify and Refine Requirements** – the preparatory tasks required to commence a new iteration of the product derivation project.
- **Derive the Product** - creates an integrated product configuration that makes maximum use of the platform and minimises the amount of product specific development required.
- **Develop the Product** - facilitates requirements that could not be satisfied by a configuration of the existing assets through component development or adaptation.
- **Test the Product** - validates the current product build.
- **Management and Assessment** - provides feedback to the platform team and monitor progress of derivation project.

Table 1 lists the tasks performed for each of these activities:

Table 1. Pro-PD activities and tasks

Activity	Tasks performed in this activity
Initiate Project	Translate Customer Requirements; Coverage Analysis; Customer Negotiation; Create the Product Requirements
Identify and Refine Requirements	Find and Outline Requirements; Create the Product Test Cases; Allocate Requirements; Create Guidance for Decision Makers
Derive the Product	Select Closest Matching Configuration; Derive New Configuration; Evaluate Product Architecture; Select Platform Components; Product Integration; Integration Testing; Identify Required Product Development
Develop the Product	Component Development; Component Testing; Product Integration; Integration Testing
Test the Product	Run Acceptance Tests
Management and Assessment	Provide Feedback to Platform Team, Monitor Project

5.2 Roles

Despite attempts to automate product derivation, it remains a human activity in which tasks are performed through collaboration and the exchange of work. In Pro-PD there are several roles that represent the different responsibilities, which occur during product derivation. These roles are: Customer, Platform Manager, Product Architect, Product Developer, Product Manager and Product Tester. These roles are assigned to specific tasks, which create and modify the different work products.

Table 2. Pro-PD Roles

Role	Responsibility
Customer	Represents the 'work' in the project. They are responsible for defining what product to build and determining the priority of features.
Platform Manager	Represents the interests of the platform during the derivation project. The role should have a degree of understanding on the demands of the product team.
Product Architect	Responsible for the major technical decision making within the derivation project. The role requires a good knowledge of the platform and an understanding of the demands on the platform team.
Product Developer	Responsible for Component Development and Component Testing. The Product Developer needs to be able to understand and conform to the product architecture.
Product Manager	Responsible for customer relationship management, negotiation of product features with the customer and project planning.
Product Tester	Responsible for the main testing effort within the project. The Product Tester should co-ordinate with the platform testing team to reuse Platform Test Artefacts.

5.3 Artefacts

In Pro-PD, an artefact is produced, modified or used by a task within the derivation process (see Table 3).

Table 3. Pro-PD Roles

Software Artefact	Platform Test Artefacts, Product Build, Product Test Cases, New Platform Release, Platform Architecture, Platform Components, Developed or Adapted Components, Existing Platform Configurations, Base Product Configuration, Integrated Product Configuration.
Documentation	Required Product Development, Translated Customer Requirements, Product Specific Platform Requirements, Product Requirements, Platform Feedback, Platform Requirements, Customer Requirements, Customer Specific Product Requirements, Negotiated Customer Requirements, Glossary.

In the following sections, the Initiate Project and Identify and Refine Requirements activities are described in detail.

5.4 Initiate Project

Derivation does not start "from scratch", i.e., by just selecting features or taking decisions described in a variability model. The *Initiate Project* activity contains the preparatory tasks required to establish a product derivation project. Table 2 describes the *Initiate Project* tasks and their purpose.

Table 4. Initiate Project Tasks

Task	Purpose
Translate Customer Requirements	To translate the *Customer Requirements* into the internal organisational language.
Coverage Analysis	To perform a comparison between the *Translated Customer Requirements* and the *Platform Requirements*. The *Translated Customer Requirements,* which are within the scope of the platform, are identified. Requirements outside the scope of the platform are contained in the *Customer Specific Product Requirements*.
Customer Negotiation	Negotiate customer requirements, which fall outside scope of the product line. Requirements are allocated to specific development iterations based on customer priority.
Create the Product Requirements	To form the *Product Requirements* using the *Negotiated Customer Requirements* and the *Translated Customer Requirements,* which were within the scope of the platform. The *Platform Requirements* can be used as a baseline.

5.5 Identify and Refine Requirements

The *Identify and Refine Requirements* activity contains the preparatory tasks required to commence a new product derivation iteration. Table 3 describes the *Identify and Refine Requirements* tasks and their purpose.

Table 5. Identify and Refine Requirements Tasks

Task	Purpose
Find and Outline Requirements	The functional and non-functional requirements for the system are specified and scoped by the *Product Architect*. With every requirement, it must be decided whether to integrate it into the platform or into an individual product [26].
Create the Product Test Cases	Design the *Product Test Cases* for requirements in the *Product Requirements*. Typically, the *Product Tester* uses the *Platform Test Artefacts* as a basis for the creation.
Allocate Requirements	The *Product Requirements* are allocated to the relevant organisational disciplines, roles and personal. The goal is to define who is responsible for resolving what remaining variability to fulfil the product requirements.
Create Guidance for Decision Makers	Guidance can be linked into the *Product Requirements*, often to external sources to provide information on the background to a particular decision. Guidance is essential, especially for domain experts like customers and sales people, who are confronted with many, often technical, decisions.

5.6 Threats to Validity

Firstly, all qualitative research suffers from the risk of bias and multiple interpretations of data. Data collected during the various research stages was analysed objectively in order to ensure minimisation of bias. Despite this, results taken from the data will be influenced by the inclusion of the Robert Bosch GmbH case study.

A second threat to validity is handling model refinements. Each stage of the research provided the basis for the revision or refinement of Pro-PD. A major challenge when making iterations was the evaluation of different suggestions with respect to each other. For example, before a correction was integrated it had to be determined whether the proposal could be characterized as being universally valid or whether it was tied to a specific context and therefore not suitable for model refinement.

6 Conclusion

In response to a need for methodological support for product derivation, the authors identified the following research objective: *To define a systematic process which will provide a structured approach to the derivation of products from a software product line based on a set of tasks, roles and work artefacts.* To meet this objective, we developed Pro-PD (Process model for Product Derivation). Pro-PD was iteratively developed and evaluated through four research stages involving academic and industrial sources. When commencing the research, we identified three limitations to current approaches, and our research, through the development of Pro-PD, has addressed each of these.

To overcome the limitation, lack of defined flow of artefacts, Pro-PD describes the usage and flow of specific artefacts through the product derivation process. This was observed in the Robert Bosch GmbH industrial case study where documentation was used to drive the product derivation process. These and other observations on artefact flow were modelled in Pro-PD.

It was clear in the early stages of our research that the variety of roles and responsibilities for product derivation could not be undertaken by a single professional group – the engineers (as in [4]). This was highlighted in particular during the Robert Bosch GmbH case study where the provision of different views, according to the role, can help reduce the lower the complexity of large decision spaces. Pro-PD defines different roles and their responsibilities.

The third limitation was the lack of process support. Pro-PD is a process model defining tasks, artefacts and roles. It is evidence-based, having being developed through industry input. In addition, it is inline with product derivation practice as defined by the Software Engineering Institute's PLPF. In particular, Pro-PD provides systematic support for product derivation preparation. We focus on this aspect of Pro-PD as we have experienced that a lack of support for preparing derivation is one of the main reasons that product derivation often fails in practice [7].

In this paper, we have described Pro-PD and outlined in more detail the tasks for two activities. The tasks we present are generic and in some situations domain-specific tasks will be required. Therefore, further research is needed to support the definition of when and how tasks are tailored to specific contexts, domains or organization. Also, validation is necessary with regard to the usefulness of the tasks in practice.

Acknowledgements. This work is supported by IRCSET under grant number RS/06/167 and by Science Foundation Ireland through Lero – the Irish Software Engineering Research Centre under grant number 03/CE2/I303_1.

References

1. Clements, P., Northrop, L.: Software Product Lines: Practices and Patterns. Addison-Wesley Longman Publishing Co., Inc., Boston (2001)
2. Hotz, L., Gunter, A., Krebs, T.: A Knowledge-based Product Derivation Process and some Ideas how to Integrate Product Development. In: Proc. of Software Variability Management Workshop 2003, Groningen, The Netherlands (2003)
3. Griss, M.L.: Implementing Product-Line Features with Component Reuse. In: ICSR 2006: Proceedings of the 6th International Conference on Software Reuse, pp. 137–152. Springer, London (2000)
4. Deelstra, S., Sinnema, M., Bosch, J.: Product Derivation in Software Product Families: A Case Study. J. Sys. Soft. 74(2), 173–194 (2005)
5. Rabiser, R., Grünbacher, P., Dhungana, D.: Supporting Product Derivation by Adapting and Augmenting Variability Models. In: 11th International Software Product Line Conference 2007, Kyoto, Japan (2007)
6. Rabiser, R., Dhungana, D.: Integrated Support for Product Configuration and Requirements Engineering in Product Derivation. In: 33rd EUROMICRO Conference on Software Engineering and Advanced Applications (2007)
7. O'Leary, P., et al.: Important Issues and Key Activities in Product Derivation: Experiences from Two Independent Research Projects. In: Proc. of the 13th International Software Product Line Conference (SPLC 2009), San Francisco, CA (2009)
8. Bayer, J., et al.: PuLSE: a Methodology to Develop Software Product Lines. In: Proceedings of the 1999 Symposium on Software Reusability. ACM, Los Angeles (1999)
9. Atkinson, C., Bayer, J., Muthig, D.: Component-based product line development: the KobrA approach. In: Proceedings of the First Conference on Software Product Lines: Experience and Research Directions. Kluwer Academic Publishers, Denver (2000)
10. Weiss, D.M., Lai, C.T.R.: Software product line engineering: A Family-based Software Development Process, 1st edn. Addison-Wesley Professional, Reading (1999)
11. Sinnema, M., et al.: Modeling Dependencies in Product Families with COVAMOF. In: 13th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS 2006), Potsdam, Germany (2006)
12. McGregor, J.D.: Preparing for Automated Derivation of Products in a Software Product Line, CMU/SEI-2005-TR-017
13. Bayer, J., et al.: PuLSE-I: Deriving Instances from a Product Line Infrastructure. In: 7th IEEE International Conference and Workshop on the Engineering of Computer Based Systems, Edinburgh, UK (2000)
14. Rabiser, R.: A User-Centered Approach to Product Configuration in Software Product Line Engineering. In: Christian Doppler Laboratory for Automated Software Engineering, Institute for Systems Engineering and Automation. Johannes Kepler University, Linz (2009)
15. Guelfi, N., Perrouin, G.: A Flexible Requirements Analysis Approach for Software Product Lines. In: Sawyer, P., Heymans, P. (eds.) REFSQ 2007. LNCS, vol. 4542, pp. 78–92. Springer, Heidelberg (2007)
16. Rombach, D.: Fraunhofer: the German model for Applied Research and Technology Transfer. In: Proceedings of the 22nd international Conference on Software Engineering (ICSE 2000). ACM, Limerick (2000)
17. Ahlemann, F., Gastl, H.: Process Model for an Empirically Grounded Reference Model Construction. In: Fettke, P., Loos, P. (eds.) Reference Modeling for Business Systems Analysis. IGI Publishing (2006)
18. Fettke, P., Loos, P.: Reference Modeling for Business Systems Analysis. IGI Publishing (2006)

19. Rosemann, M., Schütte, R.: Multi-Perspective Reference Modelling. In: Becker, J., Rosemann, M., Schütte, R. (eds.) Referenzmodellierung. State-of-the-art und entwicklungsperpektiven, pp. 22–44. Physica-Verlag, Heidelberg (1999)
20. Schlagheck, B.: Object-oriented Reference Models for Process and Project Controlling - Foundation Construction Fields of Application. Deutscher Univ. Verlag, Wiesbaden (2000)
21. Hammersley, M., Gomm, R., Foster, P.: Case Study Method: Key Issues, Key Texts. Sage Publications, London (2000)
22. The SPLC Product Line Hall of Fame, `http://www.splc.net/fame.html` (cited 3/02/2009)
23. Sinnema, M., Deelstra, S., Hoekstra, P.: The COVAMOF Derivation Process. In: Morisio, M. (ed.) ICSR 2006. LNCS, vol. 4039, pp. 101–114. Springer, Heidelberg (2006)
24. Matinlassi, M.: Comparison of Software Product Line Architecture Design Methods: COPA, FAST, FORM, KobrA and QADA. In: Proceedings of 26th International Conference on Software Engineering, ICSE 2004. EICC, Scotland (2004)
25. Rabiser, R., O'Leary, P., Richardson, I.: Key Activities for Product Derivation in Software Product Lines. J. Sys. Softw. 84(2), 285–300 (2010)
26. Birk, A., et al.: Product Line Engineering: The State of the Practice. IEEE Software 20(6), 52–60 (2003)

Software Product Lines – An Agile Success Factor?

Andrea Leitner and Christian Kreiner

Institute for Technical Informatics, Graz University of Technology, Graz, Austria

Abstract. Introducing agility into the systems and software engineering process important, as customers demand more flexibility. For companies it is essential to react on changing requirements as well as on changing market demands.

A catalog of Agile Systems Engineering success factors also lists "flexible product (line) architecture". Being counterintuitive at first sight, the relation between these agile success factors and Software Product Lines (SPL) is discussed. Equally well, the concepts of SPL are mapped to a top-down Systems Engineering (SE) approach as possible SE implementation approach.

A closer look at these mappings reveals what we call an agile continuum. It represents a time line of binding times in an SPL from domain definition time to runtime. Binding of variability can occur at any of these times. It has been named a continuum, as there is no clear point in time, when the product line process is finished and the product life cycle starts. In this respect, the concepts of "flexible product lines" (processes) and "flexible products" form a continuous spectrum.

1 Introduction

Flexibility in the development process is getting more and more necessary, since customer requirements tend to change until late development phases. As shown in Fig. 1 flexibility can be increased by delaying decisions until an as late as possible point in time. In the scenario on the left-hand side variability is removed early in the development process. The advantage is the faster development of such a system. In the right-hand side scenario, variability is left open until the latest possible point in time. Such a system is harder to develop, but has more potential for reuse and flexibility. Software Product Lines (SPL) are an example for this strategy.

Agility provides an additional qualifier on flexible systems and Systems Engineering (SE) [HdW05]. Systematically regular feedback and learning cycles help following moving and even fuzzy targets in early stages of development.

An industry practice study [SKS+10] lists success factors for Agile Systems Engineering. (see also Tab. 1). Interestingly, besides typical "lightweight" agile practices, one of these success factors reads as "flexible product (line) architecture". As product lines are deliberately planned, and intended to be long-lived, this sounds like a contradiction at first. In this paper we investigate and try to

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 203–214, 2011.

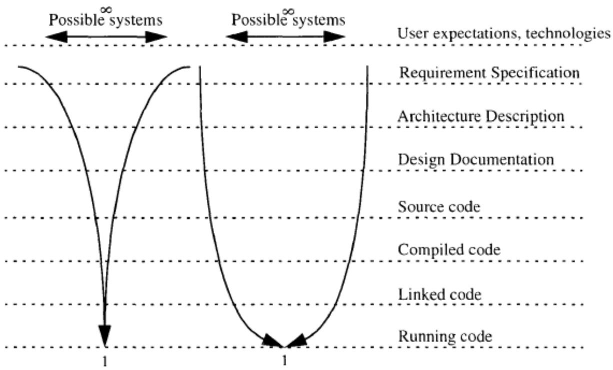


Fig. 1. Impact of early and delayed variability on flexibility [GBS01]

clarify the relation between SPL and Agile Systems Engineering, especially in the area of multi-disciplinary embedded systems development.

Another, practical motivation is the investigation of the applicability of a generic software architecture as an agile method for control software development for a broad range of hybrid electrical vehicle types [EGE10]. The goal of the HybConS[1] project is to build a software base for the fast and flexible reaction to different customer requirements.

In Sec. 2, general concepts of Agility, Software Product Lines and Systems Engineering are described. Sec. 3 gives an overview of related literature. Sec. 4 takes a closer look on the Agile SE success factors and their relation to SPL. Different aspects of SPLs and Systems Engineering are compared in Sec. 5. In Sec. 6 we introduce the idea of an Agile Continuum and conclude in Sec. 7.

2 General Concepts

2.1 Agility

Agility "*is a time based strategy for operational success ... A project is agile if it is able to execute its reorienting and action-taking cycle faster than the changes occurring in its environment*" [Ado06]. Such fast (enough) feedback and learning cycles are typical for all agile methods.

The "Agile Manifesto" [Man10] is the landmark of the agile software development community. Together with the Agile Manifesto, a collection of twelve agile principles is suggested. The mission statement of the Agile Manifesto reads as follows:

> *"Individuals and interactions over processes and tools*
> *Working software over comprehensive documentation*
> *Customer collaboration over contract negotiation*
> *Responding to change over following a plan"*

[1] http://www.iti.tugraz.at/hybcons

It is interesting to see that each of these statements seems to be contradicting the ideas of software product lines, where preplanned, systematic reuse is exploited. Nevertheless both concepts are often combined. Taking a closer look, they loose much of this contradiction. Time-boxed development and evolution of reuse artifacts using an agile method seems to be straightforward. Less obvious, also on the overall methodology common characteristics can be found. E.g. "*Maximize the work not done*" is one of those agile principles mentioned above. McGregor [McG08] states that this is valid for both, agile methods and software product lines.

2.2 Software Product Lines

An often used definition from [CN02] describes a Software Product Line as *"a set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way."*

Basically, SPL development consists of two fundamentals:

– The differentiation of domain and application engineering and
– the separation of commonalities and variabilities in domain engineering.

Characteristic for SPL is the ability to efficiently build many variants of basically the same product (*mass customization*). To realize this, variability has to be planned and managed effectively.

Variability is made explicit in variation points. A variation point can be regarded as a delayed design decision [BFG+02]. The binding time is defined as the point in time when the decision upon selection of a variant must be made [vdLSR07]. In a software engineering process it describes the step where fully or partially instantiated products are created from software artifacts that contain variability [Kru03].

The choice of the binding time has an important influence on the flexibility of the system. If the variation point is bound too early, flexibility of the product line artifacts is lost. On the other hand, late binding is costly and if the point of variant resolution is chosen to late this will unnecessarily increase costs. Many classifications for variability binding times can be found in literature. A simple schema e.g. distinguishes compile-time, link-time and start-up time [vdLSR07].

2.3 Systems Engineering

The recommended definition for the term "Systems Engineering" is proposed by the INCOSE board as *"an interdisciplinary approach and means to enable the realization of successful systems"* [INC04].

A "system" in this context is defined as *"an integrated set of elements that accomplish a defined objective. These elements include products (hardware, software, firmware), processes, people, information, techniques, facilities, services, and other support elements"* [INC04].

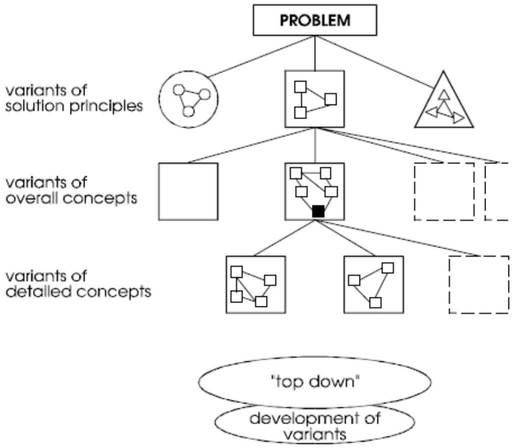


Fig. 2. The Hall-ETH system engineering approach [HdW05]

Haberfellner and de Weck [HdW05] describe several systems engineering processes. One of them is the Hall-ETH-approach. We base further investigations on this systems engineering approach. One of the basic ideas of this approach is to proceed from the general to the particular ("Top down approach") as shown in Fig. 2. In this top down approach it is proposed to consider different alternatives in each development stage. Objectives have to be formulated, possible solutions have to be identified, and, in the last step, a selection has to be taken. They discuss the possibility to install agility into the systems engineering process. For that they define three grades of agility for systems engineering:

Apply agility to existing SE-process-model – using iterative loops, and explicitly addressing flexibility.

Piecemeal engineering means to learn from detailed concept studies, and to iteratively refine the more coarse approaches.

Set-based design means to choose the realization path in a way that decisions are kept open as long as possible by maximizing the set of variants that can be dealt with in a common step.

3 Related Work

Most of the papers concerning SPL and agility, describe how to introduce agile concepts in PLE. In most cases this is described on the component layer (e.g. [ABTP09]). In this paper the view on the two topics is reverted.

McGregor [McG08] compares in his article the most important characteristics of SPL and agile methods. He states that both approaches operate within a scope. In PL activities this scope is made explicit, while it remains implicit in agile projects. Further work is postponed until needed in an agile setting. In a PL this need is systematically anticipated. Another advantage are the variation points introduced in SPL architectures. These are the spots for planned change.

Hansson and Faegri [HF08] compare agile software development and software product line engineering. Their observations are based on experience within a company they studied for some years. In that company both agile development and software product line engineering was adopted. They highlight differences between the two approaches due to their different objectives. E.g. PL advocates the importance of an architecture, whereas agile methods count on emerging architecture out of a self-organizing team. At the same time they uncover common traces on different scale factors. As an example, they also see the one of the principles in the Agile Manifesto [Man10] *Simplicity–the art of maximizing the amount of work not done–is essential* reflected in SPLE where the amount of work not done is maximized by systematic reuse. They describe a system of three integrated, iterative, customer driven processes executing in parallel with different cycle times: a strategic, technological innovation SPLE process (1-2 years), a tactical, agile software development process with 2wk-4months cycles (based on Evo [Gil05]), and a operation, request driven, day-to-day cycle.

The SPLC Hall of Fame [Hal] maintains a list of successful and instructional SPL introduction case studies. Out of 18 organizations listed, there are 8 embedded/control/SCADA product lines (automotive, aerospace, naval, defence and energy), 3 telecom systems SPLs, 4 consumer electronics and computer hardware SPLs, and only 3 classical, software-only product line approaches. Apparently, SPL is particularly well suited for complex, multi-discipline engineering domains – thus stressing the systems and systems engineering perspective.

Hoda et.al.[HKNM10] advocate to always view a software project in its context in order to individually choose the right set of methods and practices for it. Their work touches similar topics as some of the agile success factors (project setup and management) mentioned before [SKS+10]. They go into much more detail as they give guidance on how to decide which practices to use depending on the system's context; e.g. whether to adopt agility for a project or not. The context attributes described are: size, criticality, age of system, rate of change, business model, stable architecture, team distribution, and governance.

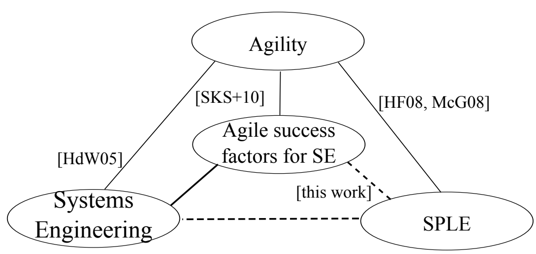


Fig. 3. Context and contribution of this paper

Fig. 3 shows the context of this paper with respect to the concepts Agility, SE, SPL, and the Agile Systems Engineering success factors. Relations between Agility, SE, and the success factors have been discussed in [HdW05, SKS+10]. [HF08, McG08] relate Agility and SPL. This paper tries to characterize the remaining relations (i) between SE and SPL as well as (ii) between the practical Agile SE success factors and SPL, in particular for multi-disciplinary and embedded systems engineering.

4 Success Factor: Flexible Product (Line) Architecture

Stelzmann et.al. [SKS+10] suggested a list of success factors for Agile Systems Engineering identified in industry. Topics addressed are: agile methods and paradigms for system specification, development and verification; project setup and management; solution finding; organizational and communication aspects; flexible products and product lines (see Tab. 1).

Table 1. Agile Systems Engineering Success Factors (from [SKS+10])

SF1	Agile project setup (alias project launch meeting)
SF2	Change response strategies
SF3	Direct customer communication
SF4	mandatory requirements and priorities
SF5	Software and hardware development coordination and collaboration
SF6/7	Flexible product (line) architecture and systematic reuse
SF8	Effectively linking requirements and tests
SF9	Know your agile method
SF10	Team-work, -thinking, -responsibility
SF11	Synchronise sprint cycles with general organisation control cycles
SF12/13	Team and inter-team organisation
SF14	Ensure minimal requirements documentation
SF15	Generic requirements

One of these factors (SF6/7) is called *Flexible product (line) architecture*. A look at this success factor seems to be confusing first, since there are two concepts mixed in one argument. In our opinion this success factor implies that a flexible product architecture is the same as a flexible product line architecture. The notation of a flexible product architecture as a success factor seems to be realistic. The more flexible the architecture of a product, the easier are changes even in later development phases. The description of a flexible product line architecture as a success factor seems to be somehow less tangible.

In this paper we are trying to find some argumentation whether a product line architecture is a success factor by itself or if it is more a mean to realize Agile Systems Engineering. An indicator for this observation is the fact that there are a few more success factors which can be realized by means of SPL. In the following we give an overview of these success factors and how they are related to SPLs.

One success factor concerns the *Handling of generic requirements* (SF15). In general, generic requirements are just one more systematically reusable artifact. This also applies to *Effectively linking requirements and tests* (SF8), because tests can be handled as reusable artifacts as well. In PL settings, this linking should be guaranteed via the domain model.

One important success factor is the *inter- and intra team communication* (SF10/12/13). Knowledge is often available only through an expert. This can lead to many problems. Especially in an agile project team it can be advantageous to make the knowledge explicit – and the project less dependent on single key persons in turn. SPLs provide powerful means for the explicit representation of domain knowledge. For systematic reuse and consistent configuration of products in an SPL, knowledge has to be explicit in some form (e.g. a domain model). This also supports the realization of the *direct customer communication* success factor (SF3). The domain model can be used to introduce a common terminology and/or link different terminologies, respectively.

Agility means to delay decisions as long as possible. As a result, it is easier to react faster to changing requirements. In SPLE, the domain engineering process leaves variability decisions open for the application engineering process. Within the boundaries of the domain scope, an SPL can react quickly to changing customer requirements. A balance between *customer/market oriented vs. mandatory requirements priorities* (SF4) is easily realizable in a SPL. Mandatory requirements can be represented as commonality, whereas customer and market oriented requirements describe variabilities. As stated later in this paper, it is not only necessary to be agile on the product level. Process agility has to be supported as well. This is included in *Change response strategies* (SF2) and *Know your agile method* (SF9). The first refers to process level agility, whereas the second more describes product level agility in context of an SPL.

Our findings are summarized in Tab. 2.

5 Software Product Line vs. Systems Engineering

5.1 Variants of Solution Principles

Prior to the start of a software-intensive project, the available coarse solution principles have to be assessed and decided upon. Typically, and depending on the context, there are three coarse scenarios described below:

Isolated solution development. Development of isolated solutions for each customer is a common, straightforward method without much consideration of reuse especially of customer visible features.

"Clone-and-own" reuse solutions. When realizing a number of similar products, a primitive, albeit popular way of reuse is to simply copy an available project, and adapt it for the new project. While being very simple, the reused substance tends to degenerate. Systematic configuration management and bug tracing is not possible across projects.

Product line solutions. Based on a centrally maintained repository of artifacts and production plans for a deliberately scoped domain, application variants can be generated within the domain's variability space. This scenario must be supported by a common architecture capable of postponing variability decisions.

These principal solution options can be systematically and quantitatively assessed using e.g. Multi-Attribute Utility Theory (MAUT) [Mau09]. Using a set of defined and weighted comparison criteria, a utility value can be calculated for each option. Typical comparison criteria can be: initial costs, maintenance costs (regular release changes), support people needed, knowledge reuse (availability of experts and reuse of training), etc. Software Product Line cost models are readily available in the literature, e.g. [CMC05, NdAM08]. A case study [LK10] showed that – when taking into account additional criteria beyond cost – the product line approach scores best even for very small numbers (3 or more) of reuse projects.

5.2 Development of Variants with SPL

Fig. 4 shows a mapping from the top down Hall-ETH approach (see Sec. 2.3). One of the basic ideas of the Hall-ETH-approach, as mentioned, is to proceed from the general to the particular. In this "top down approach" it is proposed to consider different alternatives in each development layer. Objectives have to be formulated, possible solutions have to be identified and in the last step a selection has to be taken. The concept of a SPL is similar. In the domain engineering process, generic applications are described. The decision for a concrete product is delayed until the application engineering process.

The systems engineering approach starts with a problem. Before a solution can be found, the problem has to be defined in a detailed way. Therefore the specific goals have to be specified. The same applies to PLE. First the domain scope has to be defined based on the business goals. The scope defines the boundaries of the domain. It specifies what is inside and what is outside of the domain. This is the first decision which has to be taken.

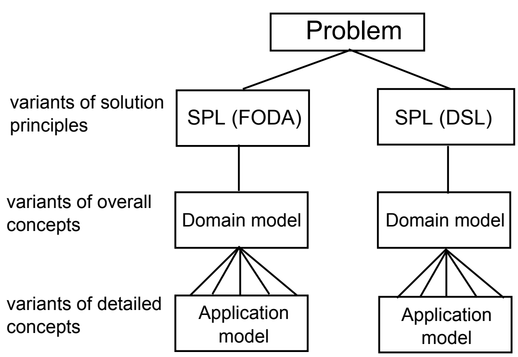


Fig. 4. Mapping Hall-ETH approach for systems engineering to SPL

After the problem definition different solution principles have to be evaluated. We split the selection of the solution principle in two main steps for better readability. Although this is more related to the process definition we think that this is a major step. Our opinion is underpinned by [HKNM10] as described in Sec. 3. In the original work of [HdW05] this seems to be a more product centric view. In the first step a basic decision for a single system project, a template solution or a SPL solution has to be taken. This step is supported by the method described in Sec. 5.1.

Table 2. Agile Systems Engineering and Software Product Lines comparison overview (only SPL/SE relevant SFs, see text)

Software product line engineering (SPLE)	Systems Engineering (Hall-ETH [HdW05])	SF3	SF4	SF6/7	SF8	SF9	SF10	SF14	SF15
Domain scope (based on business goals)	SE problem		X						
Product architecture allowing flexibility	Variants of detailed concepts (binding at start-up, run-time			X		X			
Postponing binding time	Set based design (SE agility, grade 3)			X					
Domain model with explicit variability	Variants of overall concepts	X	X	X	X	X	X	X	X
Reuse of development artifacts (requirements, components, tests, ...)				X	X			X	X
Concrete product model (derivation time binding)	Variant of detailed concept	X		X					
Concrete product derived from SPL	Detailed concept			X					

In case of a SPL there has to be a second step where different domain modeling paradigm have to be evaluated according their usability for the given domain. Typical options here are to use Feature Oriented Domain Analysis (FODA) or a Domain Specific Modeling (DSM) approach.

Once the modeling paradigm has been selected, the overall concept has to be defined. In an SPL this corresponds to domain engineering activities. The resulting domain model is an overall concept of the domain. The different domain model alternatives describing realizations with different primary views on the domain. In contrast to the Hall-ETH-approach there are not several alternatives which have to be evaluated against each other, but an entire set of possible systems described in a domain model. All the alternatives are valid and the decision for a concrete system is directed by a customer's requirements. This relates to the concept of set-based design [SM89], which is based on the philosophy of working on different solutions in parallel until one is forced to take a decision for a smaller set of solutions. Haberfellner [HdW05] defines grades of agility for systems engineering processes. Set-based design a is a highly agile concept having the highest grade three.

The detailed concepts or products in this case are described in the application engineering process, which is the next level of detail. In this stage, decisions on the actual shape of the product are taken within the boundaries of the domain.

6 Agile Continuum

In their paper [HdW05] state that *agility in the design process and agility in the product itself, are not mutually exclusive*. From our experience it is also often difficult to draw a definitive line between agility in the process and agility in the product. In the following we will describe our thoughts in more detail and introduce the term *"agile continuum"*.

In an Agile Systems Engineering approach we delay design decisions as long as possible. In other words, the binding time [GBS01] is moved to a later point in time. The possible binding times can be described on a timescale in Fig. 5. The scale starts with *domain definition time* and ends with *runtime*. With *domain definition time* we define the point in time when the domain scope is decided. Anywhere between these two points, the decision has to be bound. It is also not necessary to have only one point in time where all decisions are taken. For each point where variability occurs, an individual decision has to be taken at its defined binding time. This is exactly the notation and concept used in SPLs.

If we now take this timescale, we can divide it according the two main PL activities, domain engineering and application engineering, followed by the product life cycle. Investigating this three phases regarding the responsibility for variation binding, the following can be stated. In a delivered product the customer is responsible for the configuration (i.e. he triggers the configuration). The responsibility in the domain engineering process is clear as well. Decisions regarding the domain are taken by the domain engineers and the management, respectively. We disregard the fact that the customer has definitely influence on these decisions as he dictates requirements and is the major driver for market demands. Basically, it can be distinguished between responsibility of engineers in the domain engineering process and responsibility of the user for product configurations. The situation gets a bit more complex in the application engineering process. In this process both, the customer as well as the engineers, are responsible for decisions during product derivation. So we are not able to identify a clear boundary for responsibility.

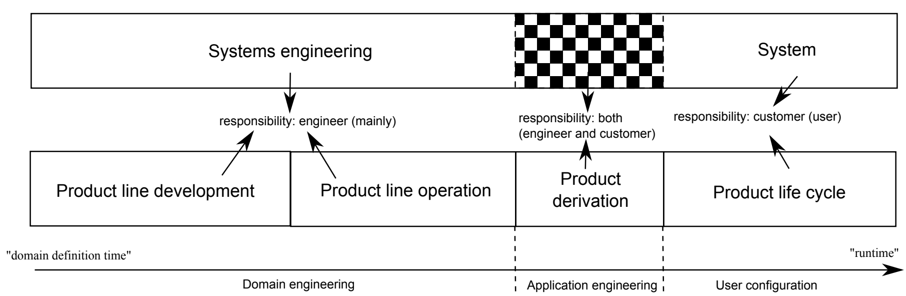


Fig. 5. The agile continuum describes a smooth transition between product line and flexible product

So the question when we still have a product line and when we have a configurable product is still unanswered? Since we could not find a clear boundary and there seem to be no criteria for differentiation, we define the term "*agile continuum*", which describes the smooth transition from a product line to a flexible or agile product. Equally well, within one system, several variability binding times can also be distributed across this continuum.

To return to our question whether a product line architecture might not be a success factor, but more a method for realization, we now can see that it can be both. Because of the - originally binding time driven - agile continuum, in real systems there also might be no clear separation between a flexible product architecture and a flexible product line architecture. Since no clear boundary between these two concepts could be found, it seems to be perfectly legitimate to formulate the success factor as it was collected from industry practice.

7 Conclusion

As stated in [HdW05], systems engineering (SE) approaches with the intent to reduce "time-to-market" have to scale down the number of possible variants at an early stage. By using Software Product Lines (SPL) as implementation approach, this situation can be eased substantially. Within the SPL domain boundaries, systems can be developed with a comparably short "time-to-market", while at the same time being able to react on changing requirements rapidly.

In this paper we investigated relation of SPL and several Agile Systems Engineering success factors proposed earlier, and how these success factors can be supported using SPL. We also discussed SPL as one solution principle for systems engineering. In a first step, a method is proposed which supports the selection between single system development, a template approach and a SPL approach. For SPLs there is a further differentiation step, which distinguishes between different domain modeling approaches. With this approach the delay of design decisions is possible. The main advantage is the fact that alternatives are not "lost", but they are provided as alternatives for further systems.

We introduced the term "agile continuum" - linked to the notion of SPL variability binding times - to show the potentially continuous spectrum between agile processes (systems engineering) and flexible products (systems).

References

[ABTP09] Babar, M.A., Tuomas, I., Pikkarainen, M.: An industrial case of exploiting product line architectures in agile software development. In: Proc. of the 13th Int. Software Product Line Conf., pp. 171–179 (2009)

[Ado06] Adolph, S.: What Lessons Can the Agile Community Learn from A Maverick Fighter Pilot? In: AGILE Conference, pp. 94–99 (2006)

[BFG+02] Bosch, J., Florijn, G., Greefhorst, D., Kuusela, J., Henk Obbink, J., Pohl, K.: Variability Issues in Software Product Lines. In: PFE 2001: Revised Papers from the 4th International Workshop on Software Product-Family Engineering, London, UK, pp. 13–21. Springer, Heidelberg (2002)

[CMC05] Clements, P.C., McGregor, J.D., Cohen, S.G.: The Structured Intuitive Model for Product Line Economics (SIMPLE). Technical report, SEI at CMU (February 2005)

[CN02] Clements, P., Northrop, L.M.: Software Product Lines: Practices and Patterns. Addison-Wesley, Reading (2002)

[EGE10] Ehsani, M., Gao, Y., Emadi, A.: Modern Electric, Hybrid Electric, and Fuel Cell Vehicles: Fundamentals, Theory, and Design, 2nd edn. CRC Press, Boca Raton (2010)

[GBS01] Van Gurp, J., Bosch, J., Svahnberg, M.: On the Notion of Variability in Software Product Lines. In: WICSA 2001: Proceedings of the Working IEEE/IFIP Conference on Software Architecture, page 45. IEEE Computer Society, Washington, DC, USA (2001)

[Gil05] Gilb, T.: Competitive Engineering: A Handbook For Systems Engineering, Requirements Engineering, and Software Engineering Using Planguage. Butterworth-Heinemann, Newton (2005)

[Hal] Product Line Hall of Fame, http://www.sei.cmu.edu/productlines/plp_hof.html (05.07.2010)

[HdW05] Haberfellner, R., de Weck, O.: Agile Systems Engineering Versus Agile System engineering. In: Proc. 15th Ann. Intl. Symposium of the Intl. Council on Systems Engineering, INCOSE (2005)

[HF08] Hanssen, G.K., Fægri, T.E.: Process fusion: An industrial case study on agile software product line engineering. J. Syst. Softw. 81, 843–854 (2008)

[HKNM10] Hoda, R., Kruchten, P., Noble, J., Marshall, S.: Agility in context. In: ACM Int. Conf. on Object Oriented Programming Systems Languages and Applications, OOPSLA 2010, pp. 74–88. ACM, New York (2010)

[INC04] Systems Engineering Handbook. Technical report, Technical Board International Council on Systems Engineering, INCOSE (2004)

[Kru03] Krueger, C.W.: Towards a Taxonomy for Software Product Lines. In: van der Linden, F.J. (ed.) PFE 2003. LNCS, vol. 3014, pp. 323–331. Springer, Heidelberg (2004)

[LK10] Leitner, A., Kreiner, C.: Managing ERP Configuration Variants: An Experience Report. In: Proc. of the 1st Workshop on Knowledge Oriented Product Line Engineering (2010)

[Man10] Manifesto for Agile Software Development (2010), http://agilemanifesto.org/

[Mau09] Multiattribute utility theory (2009), http://ddl.me.cmu.edu/ddwiki/index.php/Multiattribute_utility_theory

[McG08] McGregor, J.D.: Agile Software Product Lines, Deconstructed. Journal of Object Technology 7(8), 7–19 (2008)

[NdAM08] Nobrega, J.P., de Almeida, E.S., Meira, S.: InCoME: Integrated Cost Model for Product Line Engineering. In: Software Engineering and Advanced Applications, Euromicro Conf., pp. 27–34 (2008)

[SKS+10] Stelzmann, E., Kreiner, C., Spork, G., Messnarz, R., Koenig, F.: Agility meets systems engineering: A catalogue of success factors from industry practice. In: Systems, Software and Services Process Improvement. CCIS, vol. 99, pp. 245–256. Springer, Heidelberg (2010)

[SM89] Shigley, J.E., Mischke, C.R.: Mechanical Engineering Design, 5th edn. McGraw-Hill, New York (1989)

[vdLSR07] van der Linden, F.J., Schmid, K., Rommes, E.: Software Product Lines in Action: The Best Industrial Practice in Product Line Engineering. Springer, Berlin (2007)

Introducing Scrum in a Very Small Enterprise: A Productivity and Quality Analysis

Edgar Caballero, Jose A. Calvo-Manzano, and Tomás San Feliu

Departamento Lenguajes y Sistemas Informáticos e Ingeniería del Software,
Facultad de Informática, 28660 Boadilla del Monte, Madrid, España
ecaballero@bolesfactory.com,
{josentonio.calvomanzano,tomas.sanfeliu}@upm.es

Abstract. The very small enterprises are seriously affected by the actual global economic crisis. At this scenario the competitiveness is a key factor and productivity enhancements are needed. This article shows an experience about introducing an agile methodology in a very small enterprise. The organization needed to improve their projects productivity without compromising the quality and decided to introduce Scrum in order to verify its efficiency. The goal of this paper is to analyze the productivity and quality by comparing the Scrum pilot project with a previous similar experience based in TSPi.

Keywords: Scrum, Software Quality, Software Productivity, Process Improvement, Very Small Enterprise, TSPi.

1 Introduction

Due to the global economic crisis, organizations are forced to adapt their business strategies in order to stay in the market [1]. Spain and particularly the software industry are not the exception.

Moreover, an inappropriate quality and project management in software organizations generates cost overruns, low quality and cancelled projects [2].

Organizations must improve their competitiveness through improvements in productivity in order to survive in a market weakened by the crisis [1].

At this scenario, it is no sufficient delivering a quality product. Organizations need a more efficient resources management. Producing more in less time with the same or less cost is a key factor. A process improvement is necessary to achieve this goal [3], but a very small organization does not have enough time or resources to invest in it [4].

Several process improvements have been proposed for small organizations based on defined process control [5] and empirical process control [6]. The agile methods are based on empirical process control and have a good performance in small teams [6], therefore they are suitable for process improvements in a very small organization.

This article shows the impact of introducing an agile methodology [7] in a very small enterprise by analyzing the productivity and the quality. After an internal process selection, the organization decided to test Scrum [8]. In order to verify the organizational goals, a previous TSPi [9] experience will be compared to.

In the following sections it will be described the organization, the pilot project, the analysis of results and the conclusions.

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 215–224, 2011.

2 Context

2.1 The Organization

Bolesfactory is a very small enterprise of software development with a staff of 14 people (www.bolesfactory.com). In a previous experience, the organization was able to improve the quality and reduce the over cost thanks to a customized process based on TSPi [10].

Despite the quality improvement, the productivity has not been increased and the organization started to get worried about this competitiveness key factor at the actual crisis.

In order to improve the productivity, the organization is interested in agile methodologies and decided to test some of them in a pilot project. After evaluating several alternatives, the management decided to use Scrum.

2.2 Why Scrum?

Scrum is an agile process framework that allows organizations focus on real business value through the frequent and regular delivery of high quality software [11].

Scrum is based on an empirical process control model rather than the traditional defined process control model, which regularly inspects activities to see what is occurring and adapts them to produce desired and predictable outcomes [12].

The main criteria for selecting Scrum were:

- Several characteristics and principles similar to TSPi (see Table 1).
- Hypothesis: Scrum increases productivity and reduces time to benefits [12].
- Scrum leads as the most adopted agile methodology [13].

A comparative analysis between Scrum and TSPI was conducted to identify the main similarities and differences between them (see Table 1).

Table 1. Comparative analysis between Scrum and TSPi

Scrum	**TSPi**
Empirical process control	Defined process control
Daily project monitoring	Weekly Project monitoring
Project management	Project and quality management
Implicit quality management	Explicit quality management
Project monitoring: Burn down chart	Project monitoring: Earned value method
Impediments management	Risk management
Iterative and Incremental	Iterative and Incremental
Iterative Sprints (2-4 weeks)	Iterative Cycles (5-8 weeks)
Small and self-organized team	Small and self-organized team

Table 1. (*Continued*)

Scrum	**TSPi**
Based in communication	Based in discipline
Light measurement process	Complete measurement process
Requires Agile concepts previous training	Requires PSP previous training
Previous: Sprint 0	Previous: Launch and strategy Cycle 1
Process: Sprint planning meeting Sprint Process (custom process) Sprint review Sprint retrospective	Process: Strategy and Planning Cycle Process (defined process) No equivalent Postmortem
Sprint Process (custom process)	Cycle Process (defined process) Requirements, Design (High-level), Implementation (Detailed design, Code, Compile, Unit Test), Test
Appraisal Quality Activities: Sprint planning Daily meetings	Appraisal Quality Activities: Requirements inspection, High-level design inspection Detailed design review Detailed design inspection Code review, Code inspections
Failure Quality Activities: Sprint review	Failure Quality Activities: Compile Unit test, Build and Integration, System test

In addition, the organization identified the following main risks associated with the adoption of Scrum in the pilot project (see Table 2):

Table 2. Risks identified associated to Scrum

Risk	**Action plan**
The Scrum performance depends largely on the capability of involved team members [14]	The team consists of 4 engineers with 6 experience years. One of them has been working for 15 months with Scrum
The weakness of Scrum is about Quality management. It leaves too many things open about verification and testing [15]	Code inspections and unit test were inherited from the previous process
Scrum should be combined with another agile methodology like XP in order to improve the verification practices [15]	The organization assumed the risk
Self-managed team	Supported by the Scrum Master

2.3 The Project

The project "PRO Scrum" is a pilot experience developed for validating the Scrum productivity and quality on a very small enterprise. The project has had 4 sprints and every one had a 4 weeks duration.

At the end, the team worked one additional week in order to fix the defects detected in the Sprint 4. The strategy used for fixing defects was taken from Kniberg [16].

The team consists of 4 full time engineers with a mean of 6 experience years. One of them had been working for 15 months with Scrum. The Scrum Master and Product Owner effort has been ignored in the analysis.

The architecture and the project scope were analyzed at the Sprint 0. Besides, the team defined the following tasks to be executed for each user story: design, data base, code, code inspections and test.

The project schedule was established by the organization around in 16 weeks. The team goal was to complete all the functionality as possible in this time frame.

The results for the direct metrics are shown in Table 3. The formula applied to calculate the deviations is: % Deviation = [(Estimation – Actual) / Estimation] * 100

Table 3. Estimation vs. Actual Measures

Metrics	Estimation	Actual	Deviation
Schedule [Week]	15.0	17.0	-13.3 %
Effort [Hour]	1832.0	1985.0	-8.3 %
Size [KLOC]	13375.0	36728.0	-174.6 %

It is necessary to take in mind the following considerations about the results showed in Table 3:

- Schedule estimation: time frame was established by the organization.
- Effort estimation: incremental team estimation during each Sprint. The initial project estimation was discarded because the team was only able to estimate reliably since the Sprint 2.
- Size estimation: based in the effort estimation and the mean organization productivity (7.3 loc/hour).

3 Analysis of Results

3.1 Initial Considerations

In order to analyze the study goals, the project "PRO Scrum" will be compared with the project "PRO TSPi", which was developed by the same organization in a previous experienced based in TSPi [10].

Both projects are similar in schedule or programming language, but different in other variables (see Table 4). The indirect effect of this variability in measurements is beyond the scope of the study.

Table 4. "PRO Scrum" vs."PRO TPSi" (variables)

Variable	"PRO Scrum"	"PRO TSPi"
Project Type	Web application	Web application
Team size	4 engineers	3 engineers
Schedule	16 weeks	14 weeks
Programming language	Visual basic.net (vb.net)	Visual basic.net (vb.net)
Environment	Visual Studio 2008	Visual Studio 2005
Team member experience	6 years	4 years
Management tool	Target Process v.2.0	Ms Excel 2007
Iterations	4 Sprints	2 Cycles
Iteration length	4 weeks	7 weeks
Hardware	Processor Intel Core2 Duo RAM memory: 4 GB	Processor Intel Dual Core RAM memory: 2 GB

In order to support the analysis, both projects were divided in three phases (see Table 5)

Table 5. Project phases

Phase	Scrum scope	TSPi scope
Development	All task except test and fixing bugs	From launch to unit test
Test	All test tasks and fixing bugs	Integration and system test
Operation	From the final release until three months up	

The direct and derived metrics used in order to verify the study goals are showed in Table 6. The formula applied to calculate the difference is: % Difference = [(PRO TSPi – PRO Scrum) / PRO TSPi] * 100%

Table 6. "PRO Scrum" vs."PRO TPSi" (direct and derived metrics)

Metric	"PRO Scrum"	"PRO TSPi"	Difference
Size [loc][1]	36728.0	8500.0	-332.1%
Cycle Time [hour] (Dev + Test)	1985.0	1121.0	-77.1%
Development (Dev) Effort [%]	75.8%	90.0%	15.7%
Test Effort [%]	24.2%	10.0%	-141.6%
Effort deviation [%]	-8.3%	-18.0%	53.7%
Process Productivity [loc/hour]	18.5	7.6	-144.0%
Dev. Productivity [hour/KLOC]	41.0	118.7	65.5%
Test Productivity [hour/KLOC]	13.1	13.2	1.0%
Total Mayor Defects	300.0	266.0	-12.8%
Development Defects [%]	37.3%	83.1%	55.1%
Test Defects [%]	53.3%	13.2%	-305.3%

[1] Physical source lines of code (asp.net, vb.net, jscript and css) obtained using the "Unified CodeCount" tool [19].

Table 6. (*Continued*)

Metric	"PRO Scrum"	"PRO TSPi"	Difference
Operation Defects [%]	9.3%	3.8%	-148.3%
Defects detected (Dd.) [%]			
Dd. before test [%]	37.3%	83.1%	55.1%
Dd. before operation [%]	90.7%	96.2%	5.8%
Defect Density (DD)	8.2	31.3	73.9%
Dev DD [defect/KLOC]	3.0	26.0	88.3%
Test DD [defect/KLOC]	4.4	4.1	-5.8%
Operation DD [defect/KLOC]	0.8	1.2	35.2%
Total COQ [%]	41.2%	35.5%	-16.0%
Appraisal COQ [%]	41.3%	63.4%	34.8%
Failure COQ [%]	58.7%	36.6%	-60.2%

3.2 Productivity Analysis

The product size and the effort allow to know the productivity. The team "PRO Scrum" has written 18.5 logical code lines by hour vs. the 7.6 achieved by the team "PRO TSPi". The main difference lies in the development phase (see Figure 1)

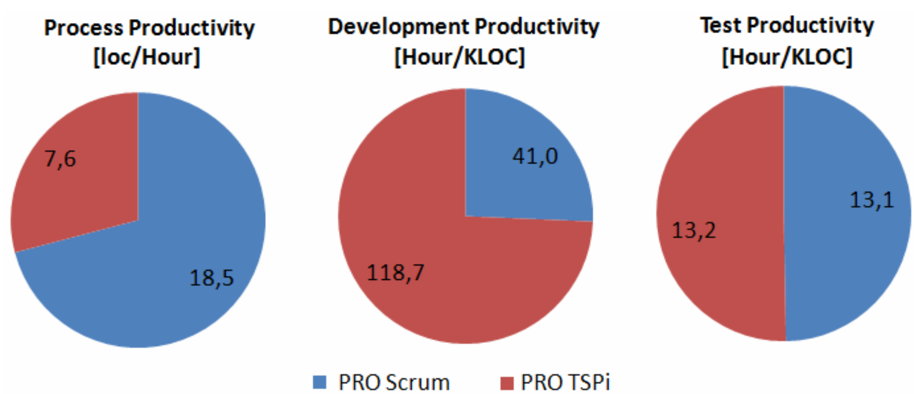


Fig. 1. "PRO Scrum" vs. "PRO TSPi" (Productivity)

Some causes that can have influenced these results are the following:

- Scrum does not define reviews or formal inspections. These activities add effort to the project and do not produce code.
- The requirements specification was made by the Scrum Master, but his effort has been excluded for this study.
- The Scrum iterations (sprints) are shorter than TSPi iterations (Cycles).
- The monitoring is daily and not weekly.

– The PSP training absence in the project "PRO TSPi" could had influenced negatively in its productivity.

In order to validate this improvement, the quality must be analyzed. If the product quality has decreased, then the process should increase the effort in quality activities, which would reduce its productivity.

3.3 Quality Analysis

The quality can have the opposite effect. A high quality product could require a lot of quality effort, and the productivity could decrease because it does not increase the product size. An early defect detection strategy increases the quality [9].

Scrum and TSPi have a different approach for quality management. TSPi defines explicit quality activities in its process, however Scrum uses daily meetings for adapting the project empirically according the business value and lets the team defines their own quality strategy.

The defects ratios found in each phase are shown in Figure 2. Because "PRO TSPi" is based on early defects detection strategy, the remaining defects before the testing phase are very low compared with 62.6% of the "PRO Scrum".

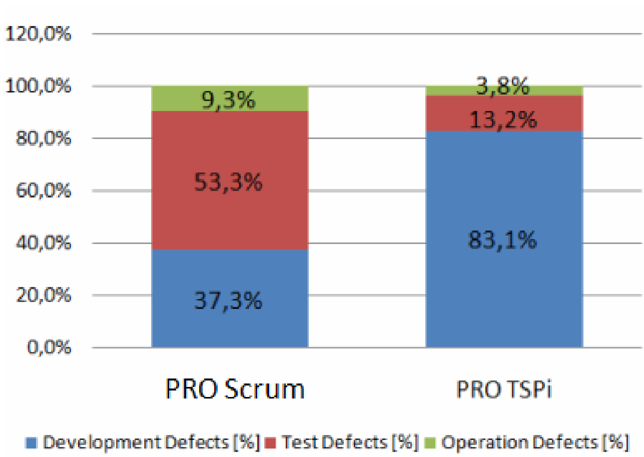


Fig. 2. "PRO Scrum" vs. "PRO TSPi" (Defect detection by phase)

Figure 2 shows that "PRO TSPI" apparently has better quality than "PRO Scrum". Only the 3.8% of all defects have been detected in the operation phase. This ratio shows the quality profile of the process, but does not determines the product quality level.

The metric that lets to know the product quality and effectiveness of each phase is the defect density. A process is considered effective when every phase has less or equal defect density than the last one [9].

The defect density measures the number of defects per 1000 lines of code. The results for both projects are shown in Figure 3.

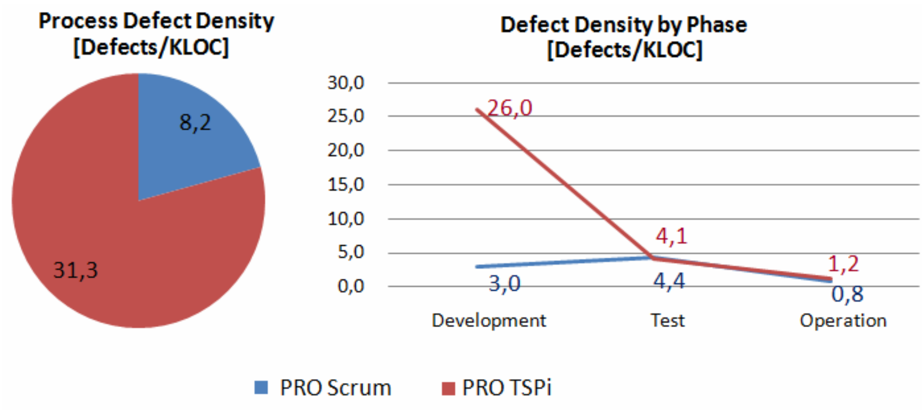


Fig. 3. "PRO Scrum" vs. "PRO TSPi" (Defect density)

The "PRO Scrum" has a better defect density in the operation phase, but the "PRO TSPi" has a better efficiency in development phase.

Some conclusions from these results are the following:

- The "PRO Scrum" did not use reviews or formal inspections in the development phase. This may have caused the low efficiency of this phase.
- Introducing reviews and formal inspections could improve the efficiency of the development phase. But if these activities are not effective, the productivity could be reduced.
- The testing phase in "PRO Scrum" represents 25.2% compared to 10.0% in "PRO TSPi" (see Table 6), nevertheless their efficiency are similar because the "PRO Scrum" is 4 times larger.
- In order to get a 0.8 value in the defect density, the "PRO TSPi" should have found 3 more defects in the test phase, which means 9 additional testing hours.

Therefore, the product quality was similar in both projects. The difference was the effort and the approach used by each one.

If the final quality of the products is similar, which of the quality approaches is more expensive? The Cost of Quality (COQ) is a measure that allows quantifying the size of the quality [17]. It has three components, but TSPi only works with two (COQ = Appraisal Costs + Failure Costs):

- Appraisal Costs are the cost of evaluating the product to determine its quality level (reviews and inspections).
- Failure Costs are the cost of diagnosing a failure, making necessary fixes, and getting back into operation (compilation and test).

In order to compare the COQ components between Scrum and TSPi, it was necessary to establish an analogy between them (see Table 7).

Code inspections and unit test were introduced to "PRO Scrum". These were inherited from the previous process based on TSPi..

Table 7. "PRO Scrum" vs."PRO TPSi" (COQ Components)

COQ component	**"PRO Scrum"**	**"PRO TSPi"**
Appraisal activities	Sprint planning Daily meetings Code inspections	Requirements inspection, High-level design inspection Detailed design review Detailed design inspection Code review, Code inspections
Failure activities	Sprint review Unit test	Compile Unit test, Build and Integration, System test

The COQ results are shown in Figure 4.

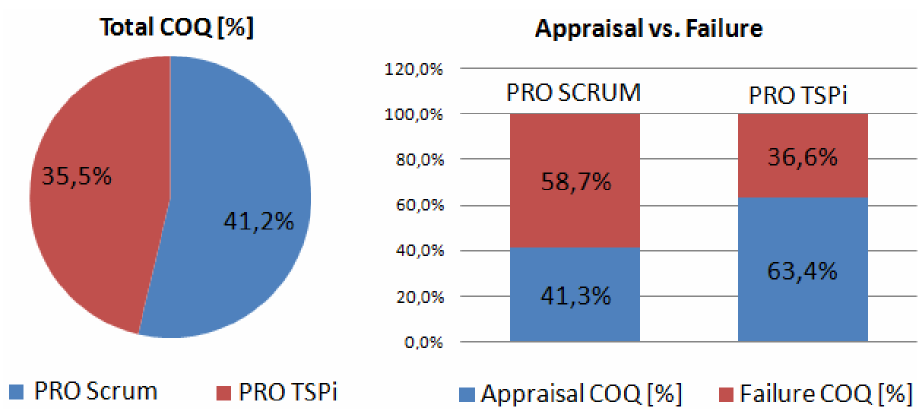


Fig. 4. "PRO Scrum" vs."PRO TPSi" (Cost of Quality)

The "PRO Scrum" has a more expensive quality process because its failure COQ component is greater. Fix defects in the test phase is more expensive.

4 Conclusions

The study shows that Scrum can enhance the process productivity without decrease the product quality in a very small enterprise.

The introduction of Scrum has been made quickly and easily, and therefore it is a good alternative for process improvement in an organization with very limited resources.

It should be emphasized that the PSP training absence in the project "PRO TSPi" could have influenced negatively its productivity.

Scrum has no explicit quality management. In this study, the team decided to introduce code inspections and unit test, but in future experiences and depending on the requirements of each project, the team could introduce new quality activities such as reviews and inspections, or combine another agile methodology as XP to improve their verification practices.

Scrum focus on project management and gives the team the possibility for organizing the development work based on a self-management criteria. For example, the team could decide to use the TSPi process, from requirements definition until the system test for every sprint. The top productivity and quality levels can be achieved combining TSP and Scrum [18].

References

1. Schwab, K.: The Global Competitiveness Report 2010–2011. World economic forum (2010), http://www3.weforum.org
2. Standish group: CHAOS Report (2009), http://www.standishgroup.com
3. Glazer, H.: Time to Market vs. Process Discipline. In: SPEG Conference, Nashville, TN, USA (2006)
4. Richardson, I., Wangenheim, C.: Why are Small Software Organizations Different? Guest editors introduction. IEEE Software 24(1), 18–22 (2007)
5. Pino, J., García, F., Piattini, M.: Software process improvement in small and medium software enterprises: a systematic review. Software Quality Journal 16(2), 237–261 (2007)
6. Tore, D., Torgeir, D.: Empirical studies of agile software development: A systematic review. Information and Software Technology 50(9-10), 833–859 (2008)
7. Agile Alliance (2011), http://www.agilealiiance.org
8. Scrum Alliance (2011), http://www.scrumalliance.org
9. Humphrey, W.: Introduction to the Team Software Process. Addison-Wesley Publishing Company, Reading (1999)
10. Caballero, E., Calvo-Manzano, J., Cuevas, G., San Feliu, T.: Implementation of Software Process Improvement through TSPi in Very Small Enterprises. In: Riel, A., O'Connor, R., Tichkiewitch, S., Messnarz, R. (eds.) EuroSPI 2010. Communications in Computer and Information Science, vol. 99, pp. 280–289. Springer, Heidelberg (2010)
11. Schwaber, K.: Agile Project Management with Scrum. Microsoft Press, Redmond (2003)
12. Schwaber, K., Beedle, M.: Agile Software Development with Scrum. Prentice Hall, Englewood Cliffs (2002)
13. West, D., Grant, T.: Agile Development: Mainstream Adoption Has Changed Agility. Forrester Research (2010)
14. Rong, G., Saho, D., Zhang, H.: SCRUM-PSP: Embracing Process Agility and Discipline. In: 2010 Asia Pacific Software Engineering Conference (2010)
15. Timperi, O.: An Overview of Quality Assurance Practices in Agile Methodologies. SoberIT, T-76.650 Seminar in software engineering, Finalandia (2004)
16. Kniberg, H.: Scrum and XP from the trenches. InfoQ (2007)
17. Humphrey, W.: A Discipline for Software Engineering. Addison-Wesley, Reading (1995)
18. Jones, C.: Applied Software Measurement, global analysis of Productivity and Quality. McGraw Hill, New York (2008)
19. Center for Systems and Software Engineering: Unified Code Count Tool (2010), http://sunset.usc.edu/research/CODECOUNT/index.html

Using ISO/IEC 29110 to Harness Process Improvement in Very Small Entities

Rory V. O'Connor[1,2] and Claude Y. Laporte[3]

[1] Lero, the Irish Software Engineering Research Centre, Ireland
[2] Dublin City University, Dublin, Ireland
[3] École de technologie supérieure, Montréal, Canada
Rory.OConnor@computing.dcu.ie, Claude.Y.Laporte@etsmtl.ca

Abstract. The recently published ISO/IEC 29110 standard Lifecycle profiles for Very Small Entities has at its core a Management and Engineering Guide [1] which are targeted at very small entities (enterprises, organizations, departments or projects) having up to 25 people [2], to assist them unlock the potential benefits of using standards which are specifically designed to address their needs. This paper will outline this new standard and the implementation of a series of pilot project initiative harnessing a set of detailed guidelines known as "Deployment Packages" to assist very small entities in understanding the potential usage of this new software process standard.

Keywords: VSE, ISO/IEC 29110, ISO, Standards.

1 Introduction

For many small and very small software companies, implementing controls and structures to properly manage their software development activity is a major challenge. Administering software development in this way is usually achieved through the introduction of a software process. All software companies are not the same and vary according to factors including size, market sector, time in business, management style, product range and geographical location. For example, a software company operating in India may have a completely different set of operational problems to contend with to a software company in Israel or Ireland. Even within a single geographical area such as Ireland, the range of operational issues faced by a small local Irish-owned firm can be radically different to those affecting a multinational subsidiary. The fact that all companies are not the same, raises important questions for those who develop software process and process improvement models. To be widely adopted by the software industry, any process or process improvement model should be capable of handling the differences in the operational contexts of the companies making up that industry. But process improvement models, though highly publicized and marketed, are far from being extensively deployed and their influence in the software industry therefore remains more at a theoretical than practical level [3].

In a time when software quality is a key to competitive advantage, the use of ISO/IEC systems and software engineering standards remains limited to a few of the

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 225–235, 2011.

most popular ones. Research shows that small and very small companies can find it difficult to relate ISO/IEC standards to their business needs and to justify the application of the standards to their business practices [2, 3, 4]. Most of these companies don't have the expertise or can't afford the resources - in number of employees, cost, and time - or see a net benefit in establishing software life-cycle processes. There is sometimes a disconnect between the short-term vision of the company, looking at what will keep it in business for another six months or so, and the long-term or mid-term benefits of gradually improving the ways the company can manage its software development and maintenance. A primary reason cited by many small software companies for this lack of adoption of software engineering standards, is the perception that they have been developed for large software companies and not with the small organization in mind [3]. To date VSEs have no or very limited ways to be recognized, by large organizations, as enterprises that produce quality software systems within budget and calendar in their domain and may therefore be cut off from some economic activities.

Accordingly there is a need to help such organizations understand and use the concepts, processes and practices proposed in the ISO/IEC JTC1/SC7's international software engineering standards. The recently published ISO/IEC 29110 standard "Lifecycle profiles for Very Small Entities" [1] is aimed at addressing the issues identified above and addresses the specific needs of VSEs.

2 Background

This section will present a brief overview of the motivation behind the adoption of standards by software development organizations and discuss the particular issues faced by small and very small companies in the adoption of software process standards. It will also introduce the specific needs of one specific category of very small organization, known as a Very Small Entity (VSE).

2.1 Standards and Benefits

Quality orientated process approaches and standards are maturing and gaining acceptance in many organizations. Standards emphasize communication and shared understanding more than anything. Examples are: any documentation is consistent and what is needed to meet the needs of the organization; all users understand the same meaning of words used - if one person says, '*Testing is completed* 'all affected bodies understand what those words mean. This kind of understanding is not only important in a global development environment; even a small group working in the same office might have difficulties in communication and understanding of issues shared by all. Standards can help in these and other areas to make the business more profitable because less time is spent on non-productive work.

There are many potential benefits of using standards. From the a VSE perspective, the benefits that certification can provide include: increased competitiveness, greater customer confidence and satisfaction, greater software product quality, increased sponsorship for process improvement, decreased development risk, facilitation of marketing, and higher potential to export. While good internal software management might help meet the first five claims; the last two can only be the benefits of using a widely recognized standard.

Many Software Process Improvement (SPI) models have been developed to assist companies in this regard and purport to represent beacons of 'best practice'. Contained within the scope of these models, according to their supporters, lies the road to budgetary and schedule adherence, better product quality and improved customer satisfaction. Some large software organizations have used SPI 'best practice' models, such as the Capability Maturity Model Integration (CMMI) [5] and the ISO 9000 series [6]. More recently, agile methodologies have been used in SPI programmes as a way of improving delivery time and increasing customer satisfaction, and these agile approaches have been widely embraced by software organizations.

2.2 Problems with Standards

Although commercial SPI models (such as CMMI) have been highly publicized and marketed, they are not being widely adopted and their influence in the software industry therefore remains more at a theoretical than practical level [4]. In the case of CMMI, evidence for this lack of adoption can be seen by examining the SEI (Software Engineering Institute) CMMI data for the three year period March 2008 to March 2011[7], which shows that worldwide during that period less than 3,500 individual appraisals were reported, which includes many divisions of the same company. It is clear that this represents a very small proportion of the world's software companies and company in-house developers. In addition, there is evidence that the majority of small and very small software organizations are not adopting standards such as CMMI. For example, an Australian study [8] found that small organizations considered that adopting CMMI "would be infeasible".

Further investigation of the SEI CMMI appraisal data reveals that in the case of Ireland – a country whose indigenous software industry is primarily made of small to medium sized organizations (SME) - fewer than 10 CMMI appraisals were conducted during the ten year period 2001 - 2011, from a population of more than 900 software companies. Therefore it is also clear that the Irish software industry is largely ignoring the most highly-publicized SPI models. In the case of CMMI (and its predecessor CMM), Staples and Niazi [9] discovered, after systematically reviewing 600 papers, that there has been little published evidence about those organizations who have decided not to adopt CMMI.

Though it is not new to claim that SPI has an associated cost, many companies are deterred from investigating SPI models because of a perceived cost. Managers' perceptions are that SPI means increased documentation and bureaucracy [3]. Such a perception is widespread and is seen as a 'feature' of standards such as CMMI. Whether or not this is true is a debatable point. The fact that managers associate CMMI with increased overhead means that most small companies do not see the model as being a viable solution or even worthy of investigation.

There is evidence [2, 3, 4] that the majority of small and very small software organizations are not adopting existing standards / proven best practice models because they perceive the standards as being developed by large organizations and orientated towards large organizations, thus provoking the debate the in terms of number of employees, size does actually matter. Studies have shown that small firms' negative perceptions of process model standards are primarily driven by negative views of cost, documentation and bureaucracy. In addition, it has been reported that SMEs find it difficult to relate standards to their business needs and to justify the application of the international standards in their operations. Most SMEs cannot afford the resources for, or see a net benefit in, establishing software processes as defined by current standards (e.g. ISO/IEC 12207) and maturity models (e.g CMMI).

2.3 Very Small Entities

The definition of "*Small*" and "*Very Small*" Entities is challengingly ambiguous, as there is no commonly accepted definition of the terms. For example, the participants of the 1995 Capability Maturity Model (CMM) tailoring workshop [10] could not even agree on what "small" really meant. Subsequently in 1998 SEPG conference panel on the CMM and small projects small was defined as "*3-4 months in duration with 5 or fewer staff*". Johnson and Brodman [11] define a small organization as "*fewer than 50 software developers and a small project as fewer than 20 software developers*".

To take a legalistic perspective the European Commission [12] defines three levels of small to medium-sized enterprise (SME) as being: **Small to medium** - "*employ fewer than 250 persons and which have an annual turnover not exceeding 50 million Euro, and/or an annual balance sheet total not exceeding 43 million Euro*"; **Small** - "*which employ fewer than 50 persons, and whose annual turnover and/or annual balance sheet total does not exceed 10 million Euro*" and **Micro** - "*which employ fewer than 10 persons and whose annual turnover does not exceed 2 million euro*".

To better understand the dichotomy between the definitions above it is necessary to examine the size of software companies operating in the market today. In Europe, for instance, 85% of the Information Technology (IT) sector's companies have 1 to 10 employees. In the context of indigenous Irish software firms 1.9% (10 companies), out of a total of 630 employed more than 100 people whilst 61% of the total employed 10 or fewer, with the average size of indigenous Irish software firms being about 16 employees [4]. In Canada, the Montreal area was surveyed, it was found that 78% of software development enterprises have less than 25 employees and 50% have fewer than 10 employees [2]. In Brazil, small IT companies (less than 50 employees) represent about 70% of the total number of companies [13].

The term "*very small entity*" had been defined by the ISO/IEC JTC1/SC7 Working Group (WG) 24 and subsequently adopted for use in the new ISO/IEC 29110 software process lifecycle standard as being "*an entity* (*enterprise, organization, department or project*) *having up to 25 people*" [2].

2.4 VSE and Standards

In a time when software quality is a key to competitive advantage, the use of ISO/IEC systems and software engineering standards remains limited to a few of the most popular ones, such as ISO 9000. Research shows that VSEs can find it difficult to relate ISO/IEC standards to their business needs and to justify the application of the standards to their business practices. Most of these VSEs can't afford the resources - in number of employees, expertise, cost, and time - or see a net benefit in establishing software life-cycle processes. There is sometimes a disconnect between the short-term vision of the organization, looking at what will keep it in business for another six months or so, and the long-term benefits of gradually improving the ways the company can manage its software development and maintenance. A primary reason cited by many small software organizations for this lack of adoption of such ISO standards, is the perception that they have been developed by and for large multi-national software companies and not with the small organisation in mind [4]. Subsequently, VSEs have no or very limited ways to be recognized as enterprises that produce quality software systems in their domain and may therefore be cut off from some economic activities.

Small software organizations, in the first instance, focus exclusively on survival. This, in part, explains the success of agile methodologies whose ‘light’, non-bureaucratic techniques support companies in survival mode attempting to establish good, fundamental software development practices. Though CMMI is firmly anchored in the belief that better processes mean better products, many small Irish software product companies are merely concerned about getting a product released to the market as quickly as possible. Development models, such as those within the agile approach, rather than CMMI or ISO 9000, are perceived as supporting this objective. This clearly poses questions for CMMI and ISO 9000 researchers. However, if SPI models are to be more widely deployed by early stage (start-ups) companies, existing models may have to be broadened to take account of the necessity for these companies to meet their development targets and ‘walk before they can run’ [4]

3 The ISO/IEC 29110 Standard

Accordingly there is a need to help such organizations understand and use the concepts, processes and practices proposed in the ISO/IEC JTC1/SC7’s international software engineering standards. The ISO/IEC 29110 standard “Lifecycle profiles for Very Small Entities” [1] is aimed at addressing the issues identified above and addresses the specific needs of VSEs [2]. The approach [2] used to develop ISO/IEC 29110 started with the pre-existing international standard ISO/IEC 12207 [14] dedicated to software process lifecycles. The overall approach consisted of three steps: (1) Selecting ISO/IEC 12207 process subset applicable to VSEs of up to 25 employees; (2) Tailor the subset to fit VSE needs; and (3) Develop guidelines for VSEs.

The core characteristic of the entities targeted by ISO/IEC 29110 is size, however there are other aspects and characteristics of VSEs that may affect profile preparation or selection, such as: Business Models (commercial, contracting, in-house development, etc.); Situational factors (such as criticality, uncertainty environment, etc.); and Risk Levels. Creating one profile for each possible combination of values of the various dimensions introduced above would result in an unmanageable set of profiles. Accordingly VSE’s profiles are grouped in such a way as to be applicable to more than one category. Table 1 illustrates a Profile Group which contains three profiles (labeled A, B and C) that are mapped to nine combinations of business models and situational factors.

Table 1. Allocating VSE characteristics to profile groups

	Profile Situational Factors		
Business Models	**Critical**	**User Uncertainty**	**Environment Change**
Contract	*Profile A*	*Profile A*	*Profile A*
In-House	*Profile C*	*Profile B*	*Profile A*
Commercial	*Profile B*	*Profile A*	*Profile A*

Profile Groups are a collection of profiles which are related either by composition of processes (i.e. activities, tasks), or by capability level, or both. The "Generic" profile group has been defined [15] as applicable to a vast majority of VSEs that do not develop critical software and have typical situational factors. This profile group does not imply any specific application domain, however, it is envisaged that in the future new domain-specific sub-profiles may be developed in the future. Table 2 illustrates this profile group as a collection of four profiles, providing a progressive approach to satisfying the requirements of profile group.

Table 2. Graduated profile of the Generic profile group

	Generic Profile Group		
Entry	Basic	Intermediate	Advanced

To date the Basic Profile [1] has been pupblished, the purpose of which is to define a software development and project management guide for a subset of processes and outcomes appropriate for characteristics and needs of VSEs.

3.1 Engineering and Management Guide

At the core of this standard is a Management and Engineering Guide (ISO/IEC 29110-5) [1] focusing on *Project Management* and *Software Implementation* as illustrated in figure 1. The purpose of the *Project Management* process is to establish and carry out in a systematic way the tasks of a software implementation project, which complies with the project's objectives in terms of quality, time and cost. *Project Management* generates a *Project Plan* to direct the software project. During the execution of the project *Change Requests* may cause revisions to the *Project Plan*. The project is the subject of *Project Assessment and Control* during the lifetimes of the project until the *Software Implementation* is complete and *Project Closure* occurs.

Software Implementation (SI) produces a specified software system implemented as a software product or service. This process starts with the establishment of

Software Requirements, after which *Architectural and Detailed Design* are produced. Software is the *Constructed* and verified using *Integration and Test* procedures. The final staged being *product d*elivery to the customer.

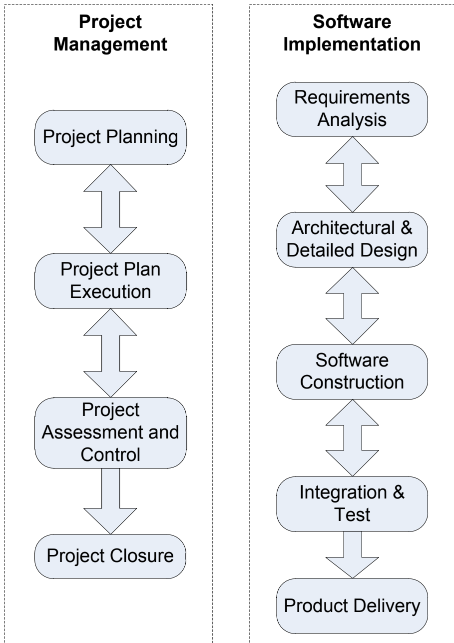


Fig 1. ISO/IEC 29110 Process Diagrams

4 Deployment Assistance

In order to assist with the deployment of ISO/IEC 29110 and to provide guidance on the actual implementation of ISO/IEC 29110-5 in VSEs a series of *Deployment Packages* and *Implementation Guides* have been developed to define guidelines and explain in more detail the processes defined in the ISO/IEC 29110 profiles.

The issues of assistance to VSEs in understanding and adopting standards, as outlined above, must be addressed. To this end, some members of the ISO/IEC JTC1/SC7 WG 24 have produced a set of "Deployment Packages" (DP) which are freely available from [16]. A DP is a set of artifacts developed to facilitate the implementation of a set of practices, of the selected framework, in a VSE. A DP is not a process reference model (i.e. it is not prescriptive). The elements of a typical DP are: description of processes, activities, tasks, roles and products, template, checklist, example, reference and mapping to standards and models, and a list of tools. The mapping is only given as information to show that a deployment package has explicit links to standards, such as ISO/IEC 12207, or models, such as the CMMI for Development, hence by deploying and implementing the package, a VSE can see its concrete step to achieve or demonstrate coverage. Packages are designed such that a VSE can implement its content, without having to implement the complete framework at the same time. The table of content of a deployment package is illustrated in figure 2.

1. Technical Description
Purpose of this document
Why this Topic is important
2. Definitions (Generic and Specific Definitions)
3. Relationships with ISO/IEC 29110
4. Detailed Description of Processes, Activities, Tasks, Steps, Roles and Products
Role Description
Product Description
Artefact Description
5. Templates
6. Examples
7. Checklists
8. Tools
9. Reference to Other Standards and Models (ISO/IEC 12207, ISO 9001, CMMI for Development)
10. References
11. Deployment Package Evaluation Form

Fig. 2. Table of Content of a deployment package

In addition a series of *Implementation Guides* have been developed to help implement a specific process supported by a tool and are freely available from [16]. To date a small number of implementation guides have been developed. These include:

- Version Control with CVS
- Version Control with SVN
- Project Management with GForge
- Issue tracking with GForge
- Software Process Improvement with OpenOffice Calc.

5 Pilot Projects

The working group (ISO/IEC JTC1/SC7 WG 24) behind the development of this standard is advocating the use of pilot projects as a mean to accelerate the adoption and utilization of ISO/IEC 29110 by VSEs around the world. Pilot projects are an important mean of reducing risks and learning more about the organizational and technical issues associated with the deployment of new software engineering practices. A successful pilot project is also an effective means of building adoption of new practices by members of a VSE. Pilot projects are based on the ISO/IEC 29110-5 Management and engineering guide [1] and the deployment package(s). In particular these are aimed to collect, as a minimum, the following data:

- Effort and time to deploy by the VSE
- Usefulness for the VSE
- Verification of the understanding of the VSE
- Self-assessments data - A self-assessment at the beginning of the pilot and at the end of the pilot project DP

To date a series of pilot projects have been completed in several countries utilizing some of the deployment packages developed. For example in Canada a pilot study has been conducted with an IT department with a staff of 4: 1 analyst and 3 developers, who were involved in the translation and implemented 3 DPs: Software Requirements, Version Control, Project Management. In Belgium a VSE of 25 people started with a process assessment phase aiming to identify strengths and weaknesses in development related processes. This company is now working on improvement actions mainly based on the following Deployment Packages: Requirement Analysis, Version Control, and Project Management. In France, a pilot study [17] was conducted with a 14-people VSE that builds and sells counting systems about the frequenting of natural spaces and public sites. In addition a further series of pilot projects are currently underway in Canada, Ireland, Belgium and France, with further pilot projects planned in the near future.

5.1 Trials to Date

To date we have published [17] the final conclusions and results of one pilot project that conducted with a 14-person VSE based in France, which successfully implemented ISO/IEC 29110 processes practices utilising the available Deployment Packages. From which we have identified some potential additional infrastructure and support process activities and suggestions for future evolution of ISO/IEC 29110 Process Profiles. A further series of pilot projects are currently underway in research laboratories and enterprises in Canada, Ireland, Belgium and France, with further pilot projects planned in the near future.

6 Discussion

As ISO/IEC 29110 is an emerging standard there is much work yet to be completed. The main remaining work item is to finalize the development of the remaining three profiles: (a) Entry – a six person-months effort project or a start-up VSEs; (b) Intermediate - Management of more than one project and (c) Advanced - business management and portfolio management practices. In addition the development of additional Profile Groups for other domains such as critical software, game industry, scientific software development are being studied.

With any new initiative there is much to be learnt from conducting pilot projects. One issue of major importance to VSEs which is emerging from these pilot projects and similar work by the ISO working group is the need for a light-weight flexible approach to process assessment. Whilst work is currently underway on an assessment mechanism for ISO/IEC 29110 [18], a clear niche market need is emerging which may force the process assessment community to change their views on how process assessments are carried out for VSEs. In particular there is a strong need to ensure that VSEs are not required to invest the anything similar in terms of time, money and other resources on process assessments, as may be expected from their larger SMEs (small and medium enterprises), or even MNC (multinational corporations) counterparts. Indeed some form of self-assessment, possibly supported by Internet based tools, along with periodic spot-checks may be suitable alternative to meet the unique needs of VSEs. It is clear that the process assessment community will have to rethink process assessment, new methods and ideas for assessing processes in VSEs.

Acknowledgments. This work is supported, in part, by Science Foundation Ireland grant 03/CE2/I303_1 to Lero, the Irish Software Engineering Research Centre (www.lero.ie).

References

1. International Organization for Standardization (ISO): ISO/IEC TR 29110-5-1-2 Software Engineering - Lifecycle Profiles for Very Small Entities (VSEs) Part 5-1-2: Management and engineering guide: Generic profile group: Basic profile, Geneva (2011)
2. Laporte, C.Y., Alexandre, S., O'Connor, R.: A Software Engineering Lifecycle Standard for Very Small Enterprises. In: O'Connor, R., et al. (eds.) Proceedings of EuroSPI. CCIS, vol. 16, pp. 129–141. Springer, Heidelberg (2008)
3. Coleman, G., O'Connor, R.: Investigating Software Process in Practice: A Grounded Theory Perspective. Journal of Systems and Software 81(5), 772–784 (2008)
4. O'Connor, R., Coleman, G.: Ignoring 'Best Practice': Why Irish Software SMEs are rejecting CMMI and ISO 9000. Australasian Journal of Information Systems 16(1) (2009)
5. Ahern, D.M., Clouse, A., Turner, R.: CMMI Distilled: A Practical Introduction to Integrated Process Improvement, 2nd edn. Addison Wesley, Boston (2004)
6. ISO 9000:2005, Quality management system. International Organization for Standardization, Geneva, Switzerland (2005)
7. CMMI Published Appraisal Results List, Software Engineering Institute, `http://www.sas.sei.cmu.edu/pars/`
8. Staples, M., Niazi, M., Jeffery, R., Abrahams, A., Byatt, P., Murphy, R.: An exploratory study of why organizations do not adopt CMMI. The Journal of Systems and Software 80(6), 883–895 (2007)
9. Staples, M., Niazi, M.: Systematic Review of Organizational Motivations for Adopting CMM-based SPI. Technical Report PA005957, National ICT Australia (2006)
10. Ginsberg, M., Quinn, L.: Process Tailoring and the Software Capability Maturity Model, Software Engineering Institute, CMU/SEI-94-TR-024 (November 1995)
11. Johnson, D., Brodman, J.: Applying the CMM to Small Organizations and Small Projects. In: Proceedings of Software Engineering Process Group Conference, Chicago (1998)

12. European Commission, The New SME Definition: User Guide and Model Declaration (2005), http://europa.eu.int/comm/enterprise/enterprise_policy/sme_definition/sme_user_guide.pdf
13. Anacleto, A., von Wangenheim, C.G., Salviano, C.F., Savi, R.: Experiences gained from applying ISO/IEC 15504 to small software companies in Brazil. In: 4th International SPICE Conference on Process Assessment and Improvement, Lisbon, Portugal (April 2004)
14. ISO/IEC 12207:2008, Information technology – Software life cycle processes. International Organization for Standardization/International Electrotechnical Commission, Geneva, Switzerland (2008)
15. O'Connor, R., Laporte, C.Y.: Towards the provision of assistance for very small entities in deploying software lifecycle standards. In: Proceedings of the 11th International Conference on Product Focused Software (PROFES 2010), ACM, New York (2010)
16. ISO/IEC JCT1/SC7 Working Group 24 Deployment Packages repository, http://profs.logti.etsmtl.ca/claporte/English/VSE/index.html
17. Ribaud, V., Saliou, P., O'Connor, R., Laporte, C.: Software Engineering Support Activities for Very Small Entities. In: Riel, et al. (eds.) Systems, Software and Services Process Improvement. CCIS, vol. 99, pp. 165–176. Springer, Heidelberg (2010)
18. International Organization for Standardization (ISO). ISO/IEC TR 29110-3, Software Engineering - Lifecycle Profiles for Very Small Entities (VSE) - Part 3: Assessment Guide, Geneva (2011)
19. O'Connor, R., Laporte, C.: Deploying Lifecycle profiles for Very Small Entities: An Early Stage Industry View. In: O'Connor, et al. (eds.) Software Process Improvement and Capability Determination. CCIS, vol. 155, pp. 227–230. Springer, Heidelberg (2011)

A Software Tool to Support the Integrated Management of Software Projects in Mature SMEs

Antonia Mas and Antoni Lluís Mesquida

University of the Balearic Islands,
Ctra. de Valldemossa, Km. 7.5. 07122 Palma de Mallorca, Spain
{antonia.mas,antoni.mesquida}@uib.es

Abstract. This paper presents *BizzProject*, a software tool to support the integrated process management according to the ISO/IEC 15504 standard. This tool has been developed by *Bizzit*, a Spanish small software development company. The current version of *BizzProject* covers the majority of the ISO/IEC 15504-7 maturity level 2 processes. The tool, which has been internally used in real projects of the company, is planned to be commercialized in the near future.

Keywords: Software Process Improvement (SPI), ISO/IEC 15504 (SPICE), Software tool, Integrated Project Management, ISO/IEC 12207, ISO/IEC 15504-7.

1 Introduction

In an environment where software development is increasingly competitive, resource optimization and quality assurance has become a priority for organizations. The need for a mature Information and Communications Technology (ICT) sector requires companies to formalize their methods and products. Certification according to internationally recognised standards and models increases the competitiveness of companies. As stated by the Spanish National Institute for Communication Technologies (INTECO), *"software quality certification has gained a lot of significance through the years. New models and initiatives that seek to provide organizations with new methodologies to improve the quality and excellence in its products are continually emerging* [1]*"*.

The implementation of a standard or a process-oriented model, such as ISO/IEC 15504, helps the organization to establish a structured working method, identifying best practices for each process in order to facilitate its deployment in the organization. Once the organization has achieved a certain level of maturity, it is able to obtain a clear and effective representation of their processes and to use agile and standardised procedures, understandable by all the involved stakeholders and independent of different ways of working. At this stage of maturity, it is possible to increase assets and provide the management team with a new source of information, allowing it to know the status of each process in each project at any time.

On the other hand, organizing processes according to a standard requires a considerable organizational effort in order to manage these processes and their resulting

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 236–246, 2011.

work products, and to control its efficiency and their continual improvement. So, if the goal is to achieve a determined level of maturity in software development, it is necessary to have a support tool that allows process implementation and management in an integrated manner in order to facilitate the deployment of the standard on all the projects performed by the company.

The weight of SMEs in the Spanish ICT sector is substantial. Indeed, they represent 99.8 per cent of the companies of this sector, of which more than 85% are micro enterprises with less than 10 employees. SMEs generate more than 60% of the added value in the sector and 70% of jobs. Micro enterprises and SMEs are potential users of the best practices recommended by ISO/IEC 15504 [2] and also of a software tool to support integrated process management. According to current data, 51% of companies know this international standard. However, it has been implemented by only 8% of them.

In the last 4 years, the weight of ICT in the Balearic GDP has doubled, from 2.57% in 2004 to 5.6% in 2008. Hence, it can be deduced that there has been an increase in the number of IT projects, in its size or in both at once. The increases in the number of projects and in its complexity, together with the growing need for efficient management, make it essential to have a platform that offers a solution to all these aspects.

In the current market it is quite difficult to find tools to support the integrated process management that can be adopted by a SME. However, there are some tools, normally ERP systems, which are used in large companies after being parameterized. These tools are hardly adaptable and adoptable for SMEs because of oversized functionality and costs. Thus, given the difficulties encountered by SMEs to find a useful tool, small enterprises which have implemented a standard have automated the support to the lifecycle processes in different ways. Two tendencies have been observed:

- The company has adopted proprietary or open source tools to support process deployment. In this case, many independent tools should be used during the development of a software project: a tool for time tracking and project planning tool, another one for accounting and budgeting, another one for checking the resulting work products, another one for controlling process performance indicators, etc.
- The company has developed its own tool, but usually it only covers some of the processes it performs.

This paper presents a tool to support the integrated process management according to the ISO/IEC 15504 standard which has been developed by a software development company of the Balearic Islands, *Bizzit*, both for internal use and to be commercialised in the future. The current version of this tool, called *BizzProject*, covers maturity level 2 processes of ISO/IEC 15504-7 [3]. This part of the standard describes an Organizational Maturity Model that may be used as a framework for determining organizational maturity.

This paper is structured as follows: Section 2 presents the company profile of *Bizzit* and summarizes its history. Section 3 describes the objectives, key aspects and steps followed during the development of *BizzProject*. Section 4 defines the functionality of the tool and details the particularities of its modules. Finally, in Section 5 the conclusions are presented.

2 Company's Background

Bizzit is a Spanish software development company which began its activity in 2004. The company has performed many projects for both the Balearic Islands civil service and other private organizations of all kind of sectors, especially for the tourism sector. Moreover, it has developed some of its own information systems for internal management. *Bizzit* has completed more than 400 projects during the last five years.

The company is structured in four business units: a software factory, an innovation department, an expansion department and a management department. It has a young team of 30 employees with different professional profiles of the IT sector.

Bizzit is involved in a software process improvement programme according to ISO/IEC 15504. All the processes ISO/IEC 15504-7 maturity level 2 have been implemented so far. Some of these processes have capability level 2 while some others are in process of achieving this level.

3 *BizzProject* Development

3.1 Objectives

The project, which is based on the development of a software tool to support the implementation of ISO/IEC 15504 to facilitate the integrated management and standardization of internal processes of the company, came up with two objectives: to organize the internal processes of *Bizzit* and to exploit a new niche market by commercializing a product which is easily applicable to any software development company interested in implementing the ISO/IEC 15504 standard.

The new integrated management support tool would enable *Bizzit* to:

- Optimize the use of resources owing to an efficient process management.
- Reach a particular capability level in the implemented processes and to achieve a higher organizational maturity level.
- Create a new product line aimed at a broad market segment. Increase the pool of potential customers.
- Expand the company into new markets, outside the Balearic autonomous region. Export the product to companies in other regions or countries, anywhere in the world. Internationally promote the company brand.

In order to give visibility to the project, *BizzProject* is expected to be published in recognized free software web sites and in software quality forums.

3.2 Processes Covered

It is expected that *BizzProject* covers all software lifecycle processes defined in the ISO/IEC 12207 [4] international standard in order to facilitate the achievement of the different organizational maturity levels described by ISO/IEC 15504-7. That is, an integral project management tool that enables the implementation of ISO/IEC 12207 processes according to the requirements of ISO/IEC 15504 is being developed.

Since, at the moment, *Bizzit* has only implemented ISO/IEC 15504-7 maturity levels 1 and 2, the tool only covers some processes of these levels. Figure 1 shows the thirteen ISO/IEC 12207 processes covered by *BizzProject*.

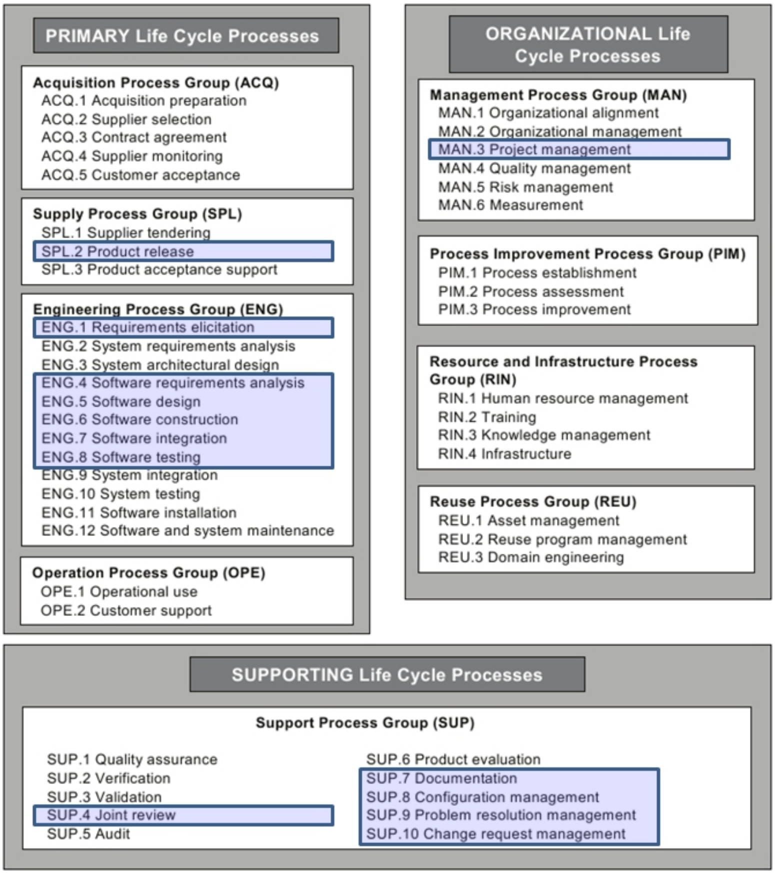


Fig. 1. ISO/IEC 12207 processes covered by *BizzProject*

3.3 Business Model

The tool is being developed under the open source business model. It is planned that software, source code and documents will be freely accessible and can be downloaded and modified by third parties. Thus, any company interested in implementing these processes may use *BizzProject* for free. *Bizzit* commits to serve companies that require adaptation of the tool to its own production process. *Bizzit* will support companies in evaluating and developing their own platform for a successful process deployment. Each new installation of the tool will be considered as a new project, similar to the implementation and parameterization of an ERP or CRM.

3.4 Technology

Regarding to technology aspects, some consolidated techniques and tools such as SOA and web services have been used for developing *BizzProject*. The main feature of the tool is the use of this architecture for project management. No other project management tool meets the objectives proposed by *BizzProject*, which are:

- A modular project management accessible via SOA or Web services. The modules will be independent of each other, accessible via Web services. For a proper access to the features of the tool, all the required WDSL files will be published.
- A scalable tool. The tool will be scalable in two aspects:
  - Functionality. Incorporating functionality will be as simple as access to new common web services, while respecting the standards and formats. In that sense, the tool operates as a common platform in which the required modules can be connected.
  - Amount of stored information. Increasing information generated by project management may be distributed as needed.
- Portability to the cloud. The project aims at a future migration to the cloud (cloud computing). In this way, a progressive and unlimited growth of required resources will be feasible.
- Relating integrated project management to the implementation of the ISO/IEC 15504 standard.

3.5 Development Phases

The development of *BizzProject* was carried out incrementally in four phases. For every single increment the Waterfall model was followed.

During the first phase, the requirements to be met by the tool in order to cover the ISO/IEC 12207 process outputs and the ISO/IEC 15504 requirements were defined. The members of our research group, MiProSoft [5], took part in this task. All the different aspects of the processes to deploy in each increment were analysed in detail. Moreover, the better way to integrate these processes was also determined.

During the second phase, after carefully analysing the related standards, the design process was initiated. A modular architecture was designed, with a specific module for each process in the standard. This architecture will facilitate any future extension of the tool. Therefore, a company wishing to implement the ISO/IEC 15504 standard should only install the modules required to achieve the target maturity level. It is important to note that not all companies must implement the same processes. Depending on the activities of the company the set of processes may vary. For example, as *Bizzit* does not have any supplier, the module corresponding to the processes related to suppliers was not developed. In this second phase the design of the technological platform, the definition of the data and the design of the screens were also undertaken.

During the third phase, the tool was constructed and unit tests for all modules were performed. Then, all modules were integrated and full application testing was conducted.

During the fourth and final phase, the tool was used in a real environment, as a pilot project of the company. All new software development projects were managed and supported by the tool. All incidents and possible improvements were recorded in order to be used for developing a new version of *BizzProject*.

3.6 Promotion and Publication of Results

From the beginning, the company's goal was twofold: on the one hand to have their own support tool and, on the other hand, to open a new line of business by commercializing the tool. Therefore, the promotion of the tool is part of *Bizzit*'s business strategy. Actions planned to promote *BizzProject* are:

- Publish the tool on the main free software web sites, both domestic and international. *Bizzit* will publish the source code and associated documentation.
- Publish the results of the project on the corporate web site. *Bizzit*'s web site has about 1,000 visits a day (more than 215,000 visits during the last year).

4 *BizzProject* Modules

The tool is accessible from the company's Intranet. The tool will have different modes of access depending on the profiles of the users: project manager, software engineer, programmer, etc. The current version of *BizzProject* divides its functionality into four main modules: a Management module, an Analysis module, a Development module and a Support module.

4.1 Management Module

The Management module only covers ISO/IEC 15504-5 MAN.3 *Project Management* process. Moreover, *BizzProject* addresses all best practices proposed by the following ISO/IEC 12207:2008 [6] processes: *Project Planning process*, *Project Assessment and control process*, *Life Cycle Model Management process* and *Project portfolio Management process*.

In order to meet ISO/IEC 15504-7 maturity level 2, the tool should cover MAN.5 *Risk Management process* too. However, the current version does not still address the best practices of this process. Next version will include MAN.5 process.

Fig. 2. *BizzProject* Management module

As it can be seen in Figure 2, the MAN.3 *Project Management* base practices are included in the different Management module's options. These options let the user identify the project's objectives and motivation and define the work to be undertaken by the project.

- By clicking on the Project Data item (1), the life cycle and strategy for the project is defined. After entering all the project information the feasibility of achieving the goals with the available resources and constraints is evaluated.
- The project manager will use the Tasks item (2) to define the project activities and tasks and the dependencies between them.
- The Project Plan item (3) enables the user to identify the experience, knowledge and skills requirements and to apply them to the selection of individuals.
- The project schedule is established using the Timing item (4).
- Once the project is planned and scheduled, project activities are implemented. Then it is necessary to record the status of the progress, to monitor the project scope, the budget, the cost, the resources and other necessary attributes. Using the Project Management item (5) the user can review the progress of the project, act to correct deviations and document them against the project baseline.
- Finally, and in order to report the current status to the affected parties, different reports are available: opening report (6), monitoring report (7), closing report (8), initial activity report (9) and final activities report (10).

4.2 Analysis Module

The Analysis module covers the ENG.1 *Requirements Elicitation*, the ENG.4 *Software Requirements Analysis* and the ENG.5 *Software Design* processes. Figure 3 shows the main window of this second module. It can be seen that its three items are directly related to the three covered processes.

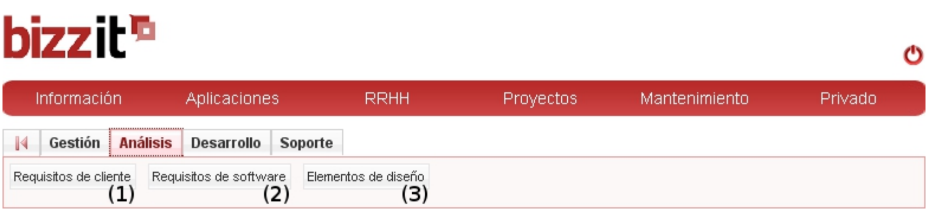


Fig. 3. *BizzProject* Analysis module

The ENG.1 *Requirements Elicitation process* is used to gather the customer needs, to establish a requirements baseline that serves as the basis for defining the needed work products and to manage the customer requirements changes. The Customer Requirements item (1) makes possible to establish links with the different types of customer agreed documents.

The ENG.4 *Software Requirements Analysis process* allows to specify and prioritize functional and non functional requirements of the software elements and to document them in a complete software requirements specification. It is possible to store the details of the performed analysis and the impact they will have in the operational environment. Consistency of customer requirements analysis to software requirements analysis is supported in *BizzProject* by establishing and maintaining traceability between them. Moreover, the tool facilitates the use of the Software Requirements item (2) to define acceptance criteria for the software product tests that should demonstrate compliance with the software requirements. All the ENG.4 base practices related to the evaluation of the requirements with the customer, to the changes approval or rejection, and to the establishment of communication mechanisms for dissemination of software requirements to all parties are also considered by this version of the tool. As in the previous item, and as in the majority of *BizzProject* functions, the tool lets the user link, for each project, all the documents related to this process.

The ENG.5 *Software Design process* is supported by the Design Elements item (3). Through this option it is possible to establish links with all the software design related products: software architecture describing the top-level structure and identifying its major software elements, detailed design for each software element describing all software units to be produced and tested, external and internal interfaces between the software elements. As in the previous process, consistency of software requirements and software design (maintaining traceability between them) is also supported by *BizzProject*.

4.3 Development Module

The Development module covers the ENG.6 *Software Construction*, the ENG.7 *Software Integration*, the ENG.8 *Software Testing* and the SPL.2 *Product Release* processes. Figure 4 shows the main window of the Development module. As the Analysis module, this module has a specific item for each covered process.

Fig. 4. *BizzProject* Development module

The ENG.6 *Software Construction process*, Software Units item (1), holds a complete description of each software unit associated to the project. This description includes: the identification, the name, a classification, the author, the state, the version, dates, the software elements that it covers, the associated software requirements and the state of each procedure and criteria. With this function the user can verify that the software unit satisfies its design requirements, including unit test cases, unit test data and code review. Links to executable representations of each software unit can also be recorded.

The ENG.7 *Software Integration process* is represented by the Software Elements item (2). In this process the strategy for integrating software units considering the software requirements is described and the integration to form a software item is performed according it. Each software unit to be integrated is selected from a list box. Moreover, using the option, the tests to be run against each integrated software item can be described. These tests include the verification of the interfaces, indicating software requirements to be checked, input data and verification criteria. *BizzProject* also records the results of these tests. The possibility of making regression tests for re-testing the integrated software items when changes are made to software units, designs or requirements, is not yet implemented in the current version. As in the previous processes, consistency of software design to software integration is also supported by *BizzProject*.

The ENG.8 *Software Testing process* is supported by the Testing item (3). This item is used to describe the tests to be run against the integrated software product, indicating software requirements being checked, input data, and verification criteria. Moreover, it can also be used to store or link all the information related to the tests and to the results of the tests. The regression test strategy is not implemented yet in this version of the tool.

Finally, the SPL.2 *Product Release process* defines the products associated with the release, and prepares the product for delivery. By using Release item (4) all the products of the release are selected from a list and the release is built ensuring integrity. The software product release documentation: type, level, duration and delivery media type for the release is also defined, produced approved and stored. Moreover, this option can be also used to prevent problems arising during delivery.

4.4 Support Module

The Support module covers the SUP.4 *Joint Review*, SUP.7 *Documentation*, SUP.8 *Configuration Management*, SUP.9 *Problem Resolution Management* and SUP.10 *Change Request Management* processes. Figure 5 shows the main window of the Support module and its five items, one for each covered process.

Fig. 5. *BizzProject* Support module

The SUP.4 *Joint Review process*, represented by the Meetings option (1) lets the user register the reviews of the project. Moreover, it can be used to store the schedule, scope and participants of management and technical reviews, depending on the needs of the project. It is also possible to identify the material to be distributed and that should be made available to all interested parties. This function supports the recording of the review results, the problems detected during the review and the proposed resolution(s) changes to work products and processes.

The Documents option (2) contains the documentation management strategy according to the SUP.7 *Documentation process*. Through this item it is possible to register standards for all kind of documents and to specify requirements for documents such as: format, title, date, identifier, version history, author(s), reviewer, authorizer, outline of contents, purpose and distribution list. Moreover, this function enables the access to all the documents generated in a project. The document reviewing before distribution and the delivery in order to make them available, are also controlled by this item. The document maintenance according to the documentation strategy is also registered by this function.

The Configuration Management item (3) is used to establish, maintain the integrity of the work products/items of a process or project and make them available to concerned parties according to the SUP.8 *Configuration management process* best practices.

The Problem Record item (4) is used to ensure that all discovered problems are identified, analyzed, managed and controlled to resolution according to the SUP.9 *Problem resolution management process*.

Finally, the Change Request Record (5) lets the user ensure that change requests are managed, tracked and controlled according to the SUP.10 *Change request management process*.

4.5 Conformity of ISO/IEC 15504-7 Organizational Maturity Model

According to the Organizational Maturity Model defined in ISO/IEC 15504-7, 15 processes define the minimum set of processes required to achieve maturity level 2. *BizzProject* covers twelve of these processes. The three maturity level 2 processes not covered by the current version are: SUP.1 *Quality Assurance process*, SUP.2 *Verification process* and MAN.5 *Risk Management process*. SUP.1 and SUP.2 are not yet implemented. MAN.5 is currently being developed but it has not yet been integrated into the existing version.

On the other hand, ISO/IEC 15504-7 defines, for each maturity level, a set of additional processes that must be implemented, or not, depending on the particular circumstances of the organization. *BizzProject* covers one additional process: the SUP.4 *Joint review process*, which is optional where the project involves agreements with stakeholders.

Next version of *BizzProject* will also support MAN.5, SUP.1 and SUP.2. Moreover, due to the workload of the company has recently increased and some developments are being outsourced, some supplier related processes will be also considered for next version. These processes are ACQ.3 *Contract Agreement process*, ACQ.4 *Supplier Monitoring process* and ACQ.5 *Customer Acceptance process*.

Finally, and with the intention to make *BizzProject* compliant with a new Spanish Certification Model developed by AENOR (Normalization Spanish Association), *Bizzit* will add some new processes contained in ISO/IEC 12207:2008 and not implemented yet in *BizzProject*. To complete the maturity level 2 of the new Spanish model the *Life Cycle Model Management process* and the *Measurement process* will be added. In the same way, the following processes will be added to complete maturity level 3: *Software Architectural Design process*, *System Architectural Design process*, *Infrastructure Management process*, *Human Resource Management process*, *Decision Management process* and *Software Validation process*.

5 Conclusion

For a small software development company, it is very important to have a simple but powerful tool for integrated project management. Moreover, if the company is involved in a software process improvement programme in order to achieve a certain level of maturity, the use of a supporting tool is strongly recommended.

In today's market, it is quite difficult to find complete tools affordable for this kind of organizations. For this reason, companies usually develop their own tools to help them in process implementation and project management. Development of such tools often involves a major effort for these SMEs.

This paper describes *BizzProject*, an integrated project management tool which supports ISO/IEC 15504 best practices. This tool has been developed by a Spanish small software development company, *Bizzit*, in order to be used internally but also with the intention to become commercialized.

BizzProject is still at a very early stage. At the moment, it only covers the software life cycle processes of ISO/IEC 15504-7 maturity level 2. Both functionality and user interface aspects need to be improved in order to be considered a marketable product. However, it is quite certain that the development of this tool has been a major effort for *Bizzit*, and has also meant a significant change in internal management structure and culture.

Acknowledgments. This research has been supported by CICYT-TIN2010-20057-C03-03 "Simulación aplicada a la gestión de equipos, procesos y servicios", Sim4Gest.

The authors wish to thank *Bizzit* for its availability and kindness and for allowing us to show this preliminary version of the tool.

References

1. Instituto Nacional de Tecnologías de la Comunicación (INTECO): Estudio sobre la certificación de la calidad en España (2008), `http://www.inteco.es/calidad_TIC/estudios_e_indicadores/publicaciones/calidad_TIC_estudios_e_informes/Calidad_software_32`
2. ISO/IEC: ISO/IEC 15504-5:2006 Information technology - Process Assessment - Part 5: An exemplar Process Assessment Model (2006)
3. ISO/IEC: ISO/IEC TR 15504-7:2008 Information technology - Process assessment - Part 7: Assessment of organizational maturity (2008)
4. ISO/IEC: ISO/IEC 12207:1995/Amd 1:2002/Amd 2:2004 Information technology - Software life cycle processes (1995)
5. Mas, A., Amengual, E., Mesquida, A.L.: Application of ISO/IEC 15504 in Very Small Enterprises. In: Riel, A., O'Connor, R., Tichkiewitch, S., Messnarz, R. (eds.) EuroSPI 2010. CCIS, vol. 99, pp. 290–301. Springer, Heidelberg (2010)
6. ISO/IEC: ISO/IEC 12207:2008 Systems and software engineering - Software life cycle processes (2008)

How Can Software SMEs Become Medical Device Software SMEs

Fergal McCaffery, Valentine Casey, and Martin McHugh

Regulated Software Research Group,
Dundalk Institute of Technology & Lero, Dundalk, Co. Louth, Ireland
{fergal.mccaffery,val.casey,Martin.McHugh}@dkit.ie

Abstract. The amount of software content within medical devices has grown considerably over recent years and will continue to do so as the level of complexity of medical devices increase. This is driven by the fact that software is introduced to produce sophisticated medical devices that would not be possible using only hardware. This therefore presents opportunities for software development SMEs to become medical device software development organisations. However, some obstacles need to be addressed and overcome in order to make the transition from being a generic software development organisation to becoming a medical device software development organisation. This paper describes these obstacles and how research that is currently being performed within the Regulated Software Research Group in Dundalk Institute of Technology may be used to assist with this transition.

Keywords: Medical device software, Software development, Software process improvement and assessment, Medi SPICE, Medical device standards, SME's.

1 Introduction - Background to Medical Device Software Development

Today software is an increasingly important component of medical devices, as it enables often complex functional changes to be implemented without necessitating changes to the hardware [1]. As a consequence of the increasing demands for greater functionally within medical devices, the complexity of medical device software development also continues to increase [2]. This has resulted in increased demand for appropriate traceability and risk management processes and tools.

It is very important that highly effective software development practices are in place within medical device companies due to the safety-critical nature of medical device software. In order to sell their products medical device companies must comply with the regulatory requirements of the countries in which they wish to market their devices [3]. Governments have put in place regulatory bodies to tackle these issues whose role is to define regulatory systems for medical devices and to ensure that only safe medical devices are placed on the market [4].

While regulatory bodies offer some guidance on what software activities must be performed, no specific method for performing these activities is provided or enforced

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 247–258, 2011.

[5]. In this context in the United States (US), the Food and Drug Administration (FDA) Center for Devices and Radiological Health (CDRH) has published guidance papers which include risk-based activities to be performed during software validation [6], pre-market submission [7] and when using off-the-shelf software in a medical device [8]. Although the CDRH guidance documents provide information on which software activities should be performed, they do not enforce any specific method for performing these activities. This can result in medical device software companies failing to comply with the expected requirements.

To help address this situation a decision was taken by the medical device industry to recognize ISO/IEC 12207:1995 [9] (a general software engineering lifecycle process standard) as suitable for general medical device software development. Subsequently the Association for the Advancement of Medical Instrumentation (AAMI) software committee carefully reviewed ISO/IEC 12207:1995 and identified a number of shortcomings due to the fact that it was a generic standard. This resulted in the decision to create a new standard which was domain specific to medical device software development. The AAMI did not discard the work done with the ISO/IEC 12207:1995 and used it as the foundation for their new standard "AAMI SW68, Medical device software – Software lifecycle processes" [10]. In 2006, a new standard IEC 62304 [11] was released that was based on the AAMI SW68 standard.

The Council of the European Communities published in 1993 the Directive 93/42/EEC (1993) [12], the "Medical Device Directive" (MDD), on medical devices. The MDD is intended to ensure the safety of medical devices placed on the market in the European Union (EU), and has the backing of national legislation in member states. Amendments to this directive occurred via Directives 2000/70/EC (2000) [13], 2001/104/EC (2001) [14], 2003/32/EC (2003) [15], and 2007/47/EC (2007) [16].

Whenever we mention medical device guidelines within this paper we refer to the following medical device standards and guidelines: IEC 62304, FDA, the MDD, ISO 14971 [17], EN 60601-4 [18], TIR 32 [19], IEC TR 80002-1[20], IEC 62366 [21],GAMP 5 [22], IEC/TR 61508 [23], ISO 13485[24] and IEC 60812 [25].

The remainder of this paper is structured in the following way. Section 2 considers the reasons why software Small and Medium sized Enterprises (SMEs) are interested in becoming medical device SMEs and the challenges they face. In section 3 recent changes made by the EU and FDA regarding medical device software are discussed. Section 4 outlines the research currently being undertaken by the Regulated Software Research Group (RSRG) in Dundalk Institute of Technology which is of particular value to assist SMEs to develop medical device software. Section 5 discusses what initial steps can be taken by SMEs when embarking on medical device software development. A conclusion is provided in section 6 and future work is outlined.

2 Why Software SMEs Are Interested in Becoming Medical Device SMEs

Software development SMEs are currently becoming medical device software development SMEs for two main reasons. The first of these is medical devices are becoming increasingly more complex and software facilitates increased functionality without the necessity to replace hardware. Therefore, due to growth in this area for this type of software there is an opportunity for software development SMEs to become medical device software development SMEs. The second reason is that software development SMEs currently developing health related software applications may now be actually developing medical device software. This is due to the most recent revision of the of Medical Device Directive [16], which states that standalone software may now be defined as a medical device.

The challenge both types of organizations face is that to become a medical device software SME requires that software be developed in a regulatory compliant manner. This is essential as in order for a medical device to be marketed it is first necessary to achieve regulatory approval for the device in the region where it will be sold. This therefore means that such organizations need to become aware of what developing regulatory compliant software means and how they will have to change their current software development processes in order to fulfill this requirement. The starting point for such organizations is to become aware of the relevant regulations.

3 Changes Impacting Medical Device Software Development

As medical devices are safety critical, they are subject to stringent regulations before they can be approved for use. Within the US medical devices must be approved by the FDA and likewise, medical devices for use within the EU must carry a CE conformance mark. This is awarded by notified bodies in each country. The most recent changes in the regulations, with regard to medical device software development, is the European MDD (2007/47/EC) [16] and the FDA Final Rule on Medical Device Data Systems [26]

On March 21st 2010, the MDD (2007/47/EC) came into force in the EU. This directive amends the MDD (93/42/EEC) [12], the Active Implantable Medical Device directive (90/385/EEC) [27] and the Biocides directive (98/8/EC) [28]. The most significant amendment within the MDD (2007/47/EC) is the provision for standalone software to be used as an active medical device. The MDD (2007/47/EC) Annex IX Section 1.4 states: *"stand-alone software is considered to be an active medical device"*. It defines an active medical device as *"any medical device operation which depends on a source of electrical energy or any source of power other than that generated by the human body or gravity"*[16].

Consequently, standalone medical device software is now subject to regulation, independent of the hardware on which it resides. As software can now be seen as the only component of a medical device, improved guidance is needed in the development of this type of software to achieve regulatory compliance. While the MDD provides medical device manufacturers with a list of harmonized standards which can be used during development to aid in achieving regulatory compliance. However, no harmonized standards exist that can provide full guidance in the development of standalone software as an active medical device. The FDA does not provide software or standalone software specific regulations. They regulate medical devices and the elements that are included as part of that device, with software being regulated in this context. To help address this the FDA provide relevant guidance documents to assist with medical device software development [7], [6], [8].

On April 18th 2011, the FDA Final Rule reclassifying Medical Device Data systems (MDDS) as Class I medical devices became effective [26]. MDDS's are off-the-shelf or custom software or hardware products used alone, or in combination, that display unaltered medical device data or transfer, store or convert medical device data for future use, in accordance with a pre-set specification. Prior to the release of this final rule, MDDS were classified as Class III devices or as an accessory to the parent device, requiring the greatest amount of scrutiny before approval could be awarded.

Whilst this ruling will make it simpler for companies developing MDDS software, it will also ring fence those which were previously beyond the net of regulatory requirements and make them subject to regulation. In addition, as part of this final rule a caveat was added to exclude devices that are used to actively monitor patients from being included as a MDDS. These devices remain classified as accessories and must undergo the same amount of regulatory conformance as the parent device to which they are connected [29]. Also standalone software such as Electronic Health Records (EHR) and Computerized Physician Order Entry (CPOE) fall outside of the scope of being defined as MDDS.

4 Research to Help SMEs Develop Medical Device Software

The RSRG was established in Dundalk Institute of Technology in February 2008. The research undertaken by the RSRG is focused on the development of an international Software Process Improvement (SPI) framework for the medical device industry. The objective of this framework is to provide a key enabler of best practice for the sector.

The RSRG have undertaken a multi-faceted approach to establishing this framework including the examination of best practice from other safety-critical domains and determining how best practice SPI models can be successfully mapped onto regulatory frameworks. This has taken place through close cooperation with the medical device industry, relevant international standards bodies and the software process improvement community. A key element of this research is the use of empirical studies of industrial practice which has been utilized to inform theory. To undertake this work the RSRG have adopted a range of quantitative and qualitative research methodologies including experiments, quantitative analysis of data sets, case studies, action research and grounded theory to provide a rich analysis of the domain.

The main focus of the RSRG is to support the growth of the medical device software industry within Ireland. However, this does not mean that their research has been restricted to Ireland, in fact the RSRG collaborates closely with international medical device organizations and SPI researchers. In particular, the RSRG is working as part of an international working group to revise the International Standard for Software Medical Device Software Lifecycle Processes (IEC 62304) and also closely with the Spice User Group to develop Medi SPICE [30]. Whilst the main deliverable for the RSRG is Medi SPICE they have also developed a number of lightweight assessment methods to assist software SMEs to become medical device software SMEs. The following subsections describe Medi SPICE and the lightweight assessment methods Med-Trace, Med-Adept and Medi SPICE-Adept.

4.1 Medi SPICE

Medi SPICE [30], [5] is a process assessment and improvement model which is domain specific to medical device software development and incorporates regulatory compliance. The results of a Medi SPICE assessment may be used to indicate the state of a medical device suppliers software practices in relation to the regulatory requirements of the industry, and identify areas for process improvement. The results of an assessment may also be used as a criterion for supplier selection.

Medi SPICE is based upon ISO/IEC 15504-5 [31] and provides coverage of the medical device software regulations. Like ISO/IEC 15504-5 and Automotive SPICE [32] it contains both a Process Reference Model (PRM) and Process Assessment Model (PAM) containing processes that provide comprehensive coverage of the FDA and European Council directives, and associated standards (e.g. ISO 14971, ISO 13485, IEC TR 80002-1, IEC 62304) for the complete software development lifecycle.

The overall objective of Medi SPICE is to provide a conformity assessment scheme to support first, second or third party assessment results that may be recognized by the regulatory bodies. The PRM and PAM of the Medi SPICE assessment model is derived from relevant ISO/IEC 15504-5 processes as they are all applicable to the development of safety-critical medical device software. As the IEC 62304 standard contains the medical device software lifecycle processes that have to be adhered to in order to achieve medical device regulatory compliance, a key objective is to provide coverage of all processes that are either included in or referenced from IEC 62304 and its associated standards.

The Medi SPICE PRM and PAM is being released in phases and consists of a defined set of software processes that contain a comprehensive set of specific practices which when utilized assist medical device software development organizations to fulfill the regulatory guidelines and standards of the medical device industry. It also addresses the requirements for process assessment and can be utilized to facilitate process improvement.

4.2 Med-Trace

As traceability is central to the development of regulatory compliant software the RSRG decided to develop an assessment method specifically to assist companies to adhere to the traceability aspects of the medical device software standards. Emanating from the Adept method [33], previously developed by the authors, and based upon CMMI® [34] and ISO/IEC 15504-5 software process reference models, Med-Trace is a lightweight assessment method that provides a means of assessing the capability of an organization in relation to medical device software traceability. Med-Trace enables software development organizations to gain an appreciation of the fundamental traceability best practices based on the software engineering traceability literature, software engineering process models (CMMI®, ISO/IEC 15504-5), and the medical device software guidelines and standards. Med-Trace may be used to diagnose an organization's weaknesses and strengths with relation to their medical device software development traceability practices.

Med-Trace is composed of 8 stages. The assessment team typically consists of two assessors who conduct the assessment between them. Stage 1, involves a preliminary meeting between the assessment team and the company wishing to undertake a Med-Trace assessment. At stage 2, the lead assessor presents an overview of the Med-Trace assessment to members of the organization who will be involved in subsequent stages. Stage 3, provides a brief insight into project documentation. The first 3 stages are normally performed on the company's premises, but the sample documentation collected in stage 3 is sometimes taken off-site as it can then be used to assist with the generation of additional questions for stage 4.

During stage 4, the assessment team return onsite and key staff members from the organization are interviewed. A set of scripted questions are utilized as a basis for these interviews. These questions are based upon the software traceability literature, traceability practices within the CMMI® and ISO/IEC 15504-5 models, and traceability practices that are required by the medical device industry. Additional questions may be asked based on the review of the documentation outlined in stage 3.

Stage 5, is a collaborative exercise between the assessors to develop the findings report using interview notes. Stage 6, involves presenting the findings report to participating staff in the organization. Stage 7, entails collaborating with staff to develop a pathway towards achieving highly effective and regulatory compliant traceability practices. The findings report provides guidance to the assessed company and focuses on practices that provide the greatest benefit in terms of the company's business goals, in addition to their quality and compliance needs.

At Stage 8, the assessed company is revisited approximately 3 months after the completion of stage 7 and their progress is reviewed against the recommended improvement path. The outcome of this stage is an updated improvement path and a final report detailing the progress that has been accomplished along with additional recommendations.

4.3 Med-Adept

The Adept method [33] was previously developed to provide a lightweight assessment of software processes from CMMI® and ISO/IEC 15504-5 and was not domain specific. The Adept method was then integrated with practices specified in medical device regulatory guidelines and standards to produce Med-Adept. Med-Adept [35] is an assessment method that provides a means of assessing the software engineering capability for processes in relation to medical device software (both application and embedded software).

Med-Adept enables software development organisations to gain an appreciation of the fundamental processes from CMMI®, ISO/IEC 15504-5 and IEC 62304 (including additional practices required by other medical device guidelines and standards) through diagnosing strengths and weaknesses in their software development practices. Med-Adept was designed to adhere to 8 of the 10 criteria outlined by Anacleto et al. [36], for the development of lightweight assessment methods: low cost, detailed description of the assessment process, guidance for process selection, detailed definition of the assessment model, support for identification of risks and improvement suggestions, conformity with ISO/IEC 15504-5, no specific software engineering knowledge required from companies' representatives, and tool support is provided. The two exceptions to the criteria outlined by Anacleto et al. [36] are that no support is provided for high-level process modeling and only the authors currently have access to the method. Med-Adept also inherits the following requirements from Adept: improvement is more important than certification, a rating is not required, preparation time required by the company is minimised; assessment time is minimized, and companies should be enabled to select assessment in the process areas that are most relevant to their business goals.

The main aims of Med-Adept are to either encourage non-medical device software development organisations to develop software for the medical device industry or to improve the software development processes within existing medical device software development organizations. However, the Med-Adept method also provides an ideal opportunity to educate software development organisations in terms of generic SPI. Therefore, the assessment would still have value even if the assessed software development company did not intend becoming a medical device software development company in the future. Consequently, Med-Adept provides medical device specific and non-medical device specific recommendations. Assessed companies are also supplied with feedback in relation to both CMMI® and ISO/IEC 15504-5 which enables such companies to decide whether they wish to follow a CMMI® or an ISO/IEC 15504-5 improvement path. Med-Adept provides the assessed company with a findings document presented in terms of processes from CMMI®, ISO/IEC 15504-5 and practices required by medical device software standards and regulations (with a particular focus on IEC 62304).

Med-Adept is composed of 8 stages which are similar to those in Adept and Med-Trace. The assessment team consists of two assessors who conduct the assessment between them. Stage 1 involves a preliminary meeting between the assessment team and the software company wishing to undergo a software process assessment. The assessment team discuss the main drivers for the company embarking upon a Med-Adept assessment and establish whether the company is interested in developing software for the medical device industry. During stage 2 the lead assessor provides an overview of Med-Adept for members of the assessed organisation who will be involved in subsequent stages. This session is used to remove any concerns that individuals may have.

Stage 3 provides a brief review of project documentation. However, the primary source of data for Med-Adept is through a series of process interviews conducted during stage 4. In this stage key staff members from the assessed organisation are interviewed. There is an interview for each process. Each interview is scheduled to last approximately 1.5 hours. To enable stage 4 to be completed within 1 day the scope of a single Med-Adept assessment is restricted to 4 processes. Each interview involves two assessors and at least one representative from the company. Stage 5 is a collaborative exercise between the assessors to develop the findings report using interview notes for each of the assessed processes. The resultant findings report consists of a list of strengths, issues and suggested actions for each of the assessed processes.

Stage 6, involves presenting the findings report to participating staff in the organisation. Stage 7, involves collaborating with staff to develop a roadmap. This provides guidance to the assessed company presenting practices that can provide the greatest benefit in terms of the company's business goals. Companies wishing to develop software for the medical device industry are recommended to focus upon establishing working practices that will assist them to fulfil the medical device regulations. Stage 8, involves revisiting the assessed company approximately 3 to 6 months after the completion of stage 7 and reviewing progress against the SPI path. The outcome of this stage is an updated SPI path and a final report detailing the progress that has been accomplished along with additional recommendations. This stage provides feedback and assistance to the assessed company after a period of time and also assists in compiling research material in terms of SPI experiences.

As Med-Adept is based upon the Adept method, existing Adept questions were used as the foundation for the Med-Adept method. Questions were added to enable coverage of medical device regulations. Even though each assessment component adopts a CMMI® process area name, it also contains questions providing coverage of relevant ISO/IEC 15504-5 processes and medical device standards and regulations.

Of the 12 Adept processes, 11 had medical device regulatory questions added for Med-Adept the exception being Measurement and Analysis which cannot be mapped against the processes of IEC 62304. Additionally, the existing Adept processes (which Med-Adept is founded upon) did not provide coverage of 5 IEC 62304 processes (Software Release, Software Maintenance, Software Problem Solution, Documentation, Software Safety Classification). Therefore, the current version of Med-Adept also does not provide coverage of these processes. Additionally, it should also be noted that the current release of Med-Adept does not include a Risk Management process.

Therefore Med-Adept does not provide complete coverage of all the medical device regulations. However, the main aim of Med-Adept is not to provide comprehensive coverage of medical device regulations, but rather to assist organisations to improve their software practices and to encourage organizations to develop medical device software. To encourage the uptake of the Med-Adept assessment by software SMEs, on-site interviewing is restricted to one day thus minimizing the time and cost associated with the assessment.

4.4 Medi SPICE-Adept

The RSRG are currently developing a new assessment method (Medi SPICE-Adept) that will enable lightweight assessments to be performed against the Medi SPICE PAM. Medi SPICE-Adept follows the same structure as Med-Adept but with several key differences:

- **Different processes then Med-Adept** - Due to changes in the configuration of the processes that makeup Medi SPICE, Med Adept does not provide coverage of all Medi SPICE processes. Medi SPICE-Adept will provide coverage of all Medi SPICE processes but currently it consists of only 10 processes: Configuration Management, Change Request Management, Software Requirements Elicitation, Systems Requirements Analysis, Software Requirements Analysis, Software Construction, Software Integration, Software Testing, Verification, Validation

- **Not based on CMMI®** – Whereas, Med-Adept was founded upon CMMI® questions, ISO/IEC 15504-5 and regulatory processes Medi SPICE-Adept is not based upon CMMI® and does not include CMMI® questions. Medi SPICE-Adept is instead based upon Medi SPICE processes.

- **Includes agile and lean questions** - Medi SPICE-Adept, unlike Med-Adept contains questions that refer to agile and lean practices. The output from an assessment is a set of strengths, issues and recommendations. As companies wish to increase the efficiency of their software development practices it was therefore considered essential to include agile and lean practices as part of the assessment.

- **Reflects changes in latest directives' and standards** - Contains processes that are in-line with the latest changes in the MDD and the FDA rules. Med-Adept was initially developed to provide assessment of Medi SPICE processes that were aligned with ISO 12207:1995, however, both IEC 62304 and ISO/IEC 15504-5 are now being revised to align with ISO 12207:2008. It is therefore important that Medi SPICE is also revised to align with ISO 12207:2008. Therefore, Medi SPICE-Adept provides coverage of the revised Medi SPICE processes.

- **May be performed over a number of days** - Whereas a Med-Adept assessment typically involved only one day of on-site interviews, a Medi SPICE-Adept assessment currently consists of 2 days of on-site interviews and provides coverage of 10 process areas. However, in the future we plan to extend the scope of the assessment to include all 38 Medi SPICE processes and 10 subprocesses. Although, it will still be possible for the organization to be assessed in as few processes as they wish and in such cases the on-site interviews may be completed in one day.

5 What Initial Steps May Be Taken by SMEs

The first step for a company is to discover if their software comes under the umbrella of a medical device as defined by either the revision of the European Council's MDD or the FDA's MDDS. The second step will then be to become aware of what standards need to be adhered to. This can be quite a time-consuming task for organizations as there are many different standards and it is difficult to determine the complete list of appropriate standards and the relationship between the different standards.

The third step is for an organization to benchmark the state of their current software development processes against recommended best practices for medical device software development. In relation to this the author's recommend adopting a Med-Trace assessment to gain an understanding of how the traceability aspects of their software development processes could be improved to assist them to achieve regulatory compliance. Additionally, the author's also recommend implementing a Medi SPICE-Adept assessment to gain an understanding as to the set of strengths and issues of an organization's current software development processes in relation to developing regulatory compliant medical device software. The output from implementing both Med-Trace and Medi SPICE-Adept will be a set of recommendations that will provide a roadmap to put an organization on the correct path towards developing medical device software using compliant processes. It should be noted that if an organization decides to implement a Medi SPICE-Adept assessment this would remove the need to perform step 2 as this assessment is based upon Medi SPICE which provides seamless alignment across all the required medical device software standards. It will then be important for the organization to prioritize the recommendations and start to implement them within their organization – this should be performed by implementing a few recommendations at a time as opposed to everything at once.

After implementing the recommendations over a period of time the organization should then embark upon another Medi SPICE-Adept and/or Med-Trace assessment to determine the level of improvement in relation to their software development processes. The output from this follow-up assessment will either be another set of recommendations or an indication that adequate processes are now in place for the organization to contact the national notified body for an official certification.

6 Conclusion

The growing requirement for medical device software and the opportunities this provides for software SMEs continues to increase. The necessity for regulatory compliance and the lack of clear guidance are often cited as barriers to entry to this expanding market. Organizations that are currently developing medical device software are mainly focused on achieving regulatory compliance rather than also improving their development process and increasing their efficiency. For both of these situations the research undertaken by the RSRG is of particular value. Medi SPICE is a process assessment and improvement model which is domain specific to medical device software development. It has been developed to address the specific requirements of medical device software companies. Med-Trace and Medi SPICE-Adept (based on Medi SPICE and Med-Adept) are lightweight assessment methods which provide effective mechanisms to assist software SMEs to enable them to put processes in place that will enable them to perform medical device software development. The RSRG plan to continue to build on its extensive experience and knowledge in this area. It is envisaged this will include the development of additional assessment methods and tools in the future as opportunities for innovation and improvement arise.

Acknowledgements. This research is supported by the Science Foundation Ireland (SFI) Stokes Lectureship Programme, grant number 07/SK/I1299, the SFI Principal Investigator Programme, grant number 08/IN.1/I2030 (the funding of this project was awarded by Science Foundation Ireland under a co-funding initiative by the Irish Government and European Regional Development Fund), and supported in part by Lero - the Irish Software Engineering Research Centre (http://www.lero.ie) grant 03/CE2/I303_1.

References

1. Lee, I., Pappas, G., Cleaveland, R., Hatcliff, J., Krogh, B., Lee, P., Rubin, H., Sha, L.: High-Confidence Medical Device Software And Systems. Computer 39(4), 33–38 (2006)
2. Rakitin, R.: Coping with defective software in medical devices. Computer 39(4), 40–45 (2006)
3. Burton, J., Mc Caffery, F., Richardson, I.: A risk management capability model for use in medical device companies. In: International Workshop on Software quality (WoSQ 2006). ACM, Shanghai (2006)

4. Mc Caffrey, F., Burton, J., Casey, V., Doring, A.: Software Process Improvement in the Medical Device Industry. In: Laplante, P. (ed.) Encyclopedia of Software Engineering, pp. 528–540. CRC Press Francis Taylor Group, New York (2010)
5. McCaffery, F., Dorling, A., Casey, V.: Medi SPICE: An Update. In: International Conference on Software Process Improvement and Capability Determinations (SPICE). Edizioni ETS, Pisa (2010)
6. US FDA Center for Devices and Radiological Health, General Principles of Software Validation, Final Guidance for Industry and FDA Staff. 2002. CDRH, Rockville (2002)
7. US FDA Center for Devices and Radiological Health, Guidance for the Content of Premarket Submissions for Software Contained in Medical Devices. CDRH, Rockville (2005)
8. US FDA Center for Devices and Radiological Health, Off-The-Shelf Software Use in Medical Devices; Guidance for Industry, medical device Reviewers and Compliance. CDRH, Rockville (1999)
9. ISO/IEC 12207:1995, Information Technology — Software life Cycle Processes. ISO: Geneva, Switzerland (1995)
10. ANSI/AAMI SW68:2001, Medical device software - Software life cycle processes. AMMI, Arlington (2001)
11. IEC 62304:2006, Medical device software—Software life cycle processes. IEC, Geneva, Switzerland (2006)
12. European Council, Council Directive 93/42/EEC Concerning Medical Devices. Official Journal of The European Communities, Luxembourg (1993)
13. European Council, Council Directive 2000/70/EC (Amendment), Official Journal of The European Union, Luxembourg (2000)
14. European Council, Council Directive 2001/104/EC (Amendment). Official Journal of The European Union, Luxembourg (2001)
15. European Council, Council Directive 2003/32/EC (Amendment). Official Journal of The European Union, Luxembourg (2003)
16. European Council, Council Directive 2007/47/EC (Amendment). Official Journal of The European Union, Luxembourg (2007)
17. ISO 14971:2007, Medical Devices — Application of risk management to medical devices. ISO, Geneva (2007)
18. BS EN 60601-1-4:2000, Medical Electrical Equipment, Part 1 - General requirements for safety. BSI, London (2000)
19. AAMI TIR32:2004, Medical device software risk management. AAMI, Arlington (2005)
20. IEC/TR 80002-1:2009, Medical device software Part 1: Guidance on the application of ISO 14971 to medical device software. BSI, London (2009)
21. IEC 62366:2007, Medical devices - Application of usability engineering to medical devices. IEC: Geneva, Switzerland (2007)
22. GAMP 5:2008, A Risk-Based Approach to Compliant GxP Computerized System. ISPE, Florida (2008)
23. IEC/TR 61508:2005, Functional safety of electrical/electronic/ programmable electronic safety related systems. BSI, London (2005)
24. ISO 13485:2003, Medical devices — Quality management systems — Requirements for regulatory purposes. ISO, Geneva, Switzerland (2003)
25. IEC 60812:2006, Analysis technique for system reliability - Procedure for failure modes and effects analysis (FMEA). IEC: Geneva, Switzerland (2006)
26. US FDA, 21 CFR Part 880 Medical Devices; Medical Device Data Systems Final Rule. Federal Register 76(31), 8637–8649 (2011)

27. European Council, Council Directive 90/385/EEC On the approximation of the laws of the Members States relating to active implantable medical devices. Official Journal of The European Communities, Luxembourg (1990)
28. European Council, Council Directive 98/8/EC Concerning the placing of biocidal products on the market. Official Journal of The European Communities, Luxembourg (1998)
29. Thompson, B., FDA Regulations of Mobile Health. In: Mobi Health News, p. 51. Chester Street Publishing (2010)
30. McCaffery, F., Dorling, A.: Medi SPICE: An Overview. In: International Conference on Software Process Improvement and Capability Determinations (SPICE), Turku, Finland (2009)
31. ISO/IEC 15504-5:2006, Information technology — Process Assessment — Part 5: An Exemplar Process Assessment Model. ISO, Geneva, Switzerland (2006)
32. Automotive SPICE Process Assessment. SIG (August 21, 2005)
33. Mc Caffery, F., Richardson, I., Coleman, G.: Adept – A Software Process Appraisal Method for Small to Medium-sized Irish Software Development Organisations. In: European Systems & Software Process Improvement and Innovation (EuroSPI 2006), Joensuu, Finland (2006)
34. CMMI Product Team, Capability Maturity Model ® Integration for Development Version 1.2, Software Engineering Institute (2006)
35. Mc Caffery, F., Casey, V.: Med-Adept: A Lightweight Assessment Method for the Irish Medical Device Software Industry. In: European Systems & Software Process Improvement and Innovation Conference (EuroSPI), Grenoble, France (2010)
36. Anacleto, A., von Wangenheim, C.G., Salviano, C.F., Savi, R.: Experiences gained from applying ISO/IEC 15504 to small software companies in Brazil. In: 4th International SPICE Conference on Process Assessment and Improvement, Lisbon, Portugal (2004)

Towards Innovative Software Projects – Creating Environments Supporting Innovation and Improvement

Kouichi Kishida

Key Technology Laboratory, SRA Inc., Japan,
2-32-8 Minami-Ikebukuro, Toshima, Tokyo 171-8513, Japan
Tel.: +81-3-5979-2688
k2@sra.co.jp

Abstract. Immaterial labour, which is a sociological concept proposed by Maurizio Lazzarato and others for understanding today's post-Fordism industry, refers to the process of producing the informational and cultural contents of a commodity. Through examining software development and the software-intensive society with the lens of immaterial labour, this paper aims to consider about establishing a new conceptual framework to promote innovation in actual software development projects.

Keywords: Immaterial labor, software development paradigm, innovation.

1 Introduction

The concept of software engineering was born as the result of discussion at the two NATO workshops [1]. We have looked up to established engineering fields to borrow concepts, theories and ideas to guide the understanding and development of the software. Much of research in software engineering has been done strongly influenced by the efficiency of Fordism, the modern industrial production that is built upon the Taylor's criteria: serialization of work, division of labor, and insignificance of individual differences.

The engineering framework has demonstrated success as long as software are the solution tool for a well-defined problem, for instance, to calculate the trajectory of a missile faster, or to simulate air dynamics more accurately, or to search a phrase within a large body of text more efficiently.

However, most of software systems we create today are no longer confined to solving well-known problems. Rather, they are cultural and informational products that redefine the way we work, learn, communicate, and entertain. When such personal, cultural and social elements become essential in software systems, the traditional engineering framework lacks something very fundamental in software development process. It is time for us to examine how software systems are produced and consumed differently than other engineered commodities are, and to consider about the implications of the difference.

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 259–267, 2011.

2 Immaterial Labor

When cultural and informational products (such as music records or videos) become usual commodities, people tend to think that those are another type of industrial products resulted from subjective intellectual labour.

Maurizio Lazaretto [2] argues the situation from a different viewpoint. He explores the hegemonic form of labour in today's post-Faradism industrial society. What happening is the mode of production, socialization and appreciation of cultural and informational products has gradually crept into the general economy, transforming industrial production processes into forms of immaterial labour in which "information and communication play an essential role in each stage of the process of production". Also, a series of various activities are involved in "defining and fixing cultural and artistic standards, fashions, tastes, consumer norms, and, more strategically, public opinion."

The cultural and (knowledge-oriented) informational activities performed during immaterial labour process are not new. Historically, they were once in the privileged domain of the intellectual bourgeoisie, and were not normally considered as labour because they were evaluated and appreciated by categorically different criteria and principles other than factory labour. What makes it important now is that more and more workers are engaged in the form of immaterial labour due to the advance of IT and network technology. Immaterial labour is not only becoming a dominant sector that replaces the industrial sector, but also becoming a predominant feature of all kinds of labour, transforming old-fashioned material production labour into new shape.

Immaterial labour reverses the relationship between production and consumption. Traditionally, production creates needs and consumption creates value. Immaterial labour does not produce merely for the satisfaction of known needs of consumption. Instead, it creates new consumption needs. In a world of abundance, most post-modern production (such as music or fashion) is geared toward people's immaterial (social and cultural) desire stimulated by producers, rather than material needs. The value created through immaterial labour tends to be expressed in terms of emotional, interpretational, and communicational experiences, through the materialization and realization of some needs that may not have existed in the first place.

In other words, immaterial products will have an economic value only when the ideas and intentions behind them are clearly communicated to and accepted by the consumers. As Georg Simmel points out, only until individuals are sufficiently acquainted with objects, they are able to assign their respective values [3]. Socialization is the precondition for the creation of a product's value because it gives "a place in life" of the society.

The value and quality of products produced through immaterial labour depend on the knowledge and the innovation of its entire workforce. This redefines the competences of work forces in the following three aspects.

(1) Intellectual skills: Cultural and informational values of a product cannot be created by the mere execution of predefined procedures. Instead, it depends on whether workers are able to innovate by identifying problems and creating new solutions. For any given problems at each stage of production,

there exist often many alternative solutions, and workers are responsible to make choices based on their own criteria. This innovation and decision-making process is mainly a factor of the knowledge, taste, and personality of the worker. Intellectual skills, which used to be individual and private, now become the main means of production and the direct force for creating values of a product.

(2) Communicative skills: Immaterial labor requires cooperation and collective coordination. The quality of work is not only defined by the worker's individual professional capacities, but also by his/her capabilities of initiating and managing productive cooperation with others. In addition to communicating with his/her peers, the worker also needs to be able to clearly communicate the value of the product with possible customers as discussed above.

(3) Autonomy: Workers of immaterial labor are responsible for their own control, and to make plans and follow through. Immaterial labor cannot be divided into simple and repetitive elements. It is hardly possible for a supervisor to intervene directly how the work should be done. Autonomy requires workers have the capability of dealing with unpredictable situations and to be self-motivated in times of uncertainty.

3 Software Development as Immaterial Labor

Taylor's principles of scientific management, which are based on the concept of planning and reducing work to simple elements to achieve efficiency, and the standardization and specialization of work processes, are not applicable for immaterial labor. Managers need to get out of the mentality of foreman that monitors and supervises their members. Instead, managers should work more like a facilitator, recognizing that the autonomy and freedom of labor as the only possible form of cooperation and innovation in production.

The systems theory views an organization as a system consists of inter-related and mutually dependent individual professionals who join the organization only when they feel the reward is fair to their contribution. Lazzarato calls for new approaches to organizing immaterial labor. He writes: "labor and direct subjugation (to organization) no longer constitute the principal form of contractual relationship between capitalist and worker. A polymorphous self-employed autonomous work has emerged as a dominant form." Various size of productive unit could be formed for specific projects, and exist only for the duration of that job. When job is done, workers are returned to the basin of immaterial labor.

The philosophical framework of immaterial labor is relevant to software development and software industry in two respects. First, software development can be viewed as a kind of immaterial labor. Software is not made of physical material, and most of software systems we develop today redefine the way we work, learn, communicate and entertain, the values of which come into existence only after the users had some degree of experience with them.

The second respect, which may not be as obvious as the first one, is that software systems are the driving force that transforms material labors into immaterial labors.

It is the use and consumption of software systems that characterize many labors as immaterial labors because software pushes labor activities "to the side of the production instead of being its chief actor" [4]. A large portion of employees of automobile companies and consumer electric companies are now engaged not in physical production lines but in interacting with software systems. Software developers are not only developing tools for users, they are also changing social and productive forms for those users.

Requirements for software systems are no longer something there to be captured and analyzed in advance. They are something to be innovated and designed by us. Software is not created to satisfy some existing needs, or to model a reality computationally; in contrast, software materializes some form of vague or even non-existing needs and reshapes the reality of its users. It is not the needs that lead to production; it is the production and consumption that leads to materialization and emergence of those potential needs.

Software systems are not isolated products that we deliver over the fence of factory to customers. Instead, they serve as the media to bear a social relationship between those who produce software and those who consume it. The sustained existence and success of the software industry relies very much on the new needs that continuously come from customers' use and experience of the systems, as well as at our capability of innovation to generate and stimulate new needs.

Therefore, the constantly changing requirements of users are not problems that software engineering research should aim to resolve; rather they are the very basis for the value of software systems, representing the opportunities that should be explored and nurtured further. The relative new concept of "forever beta" may be a mere reflection of the very nature of the type of software systems developed through immaterial labor.

Socialization of software is the precondition for the recognition of software value. This explains the increasing new practice of software sales: releasing software with free trial times. Free trials become essential means for software developers to communicate the value of the software. Once customers recognize the value by using the product through free trials, they may be more willing to pay for their future experience. Similarly, open source software becomes the means for the software industry as a whole to communicate to the society the value of software, and to generate new needs for software systems [5].

Although innovation in the production process, which is the main focus of current software engineering research, remains to be important, it becomes more important to be innovative in what to produce, not only how to produce.

When viewing software product as results of immaterial labor, it is easy to realize that software development is not about building computational models or representations of reality. Software is increasingly becoming a part of the society because it creation and consumption reshapes the world through the creation of new modes of production and consumption and through the enabling of new experience.

In reexamining software development from the perspective of immaterial labor, software development is considered as a kind of design task. It has two aspects: the design of a software system as a product (i.e., the traditional software design), and the design of the value that the software system would communicate with those who consume it. It is not that one precedes the other: the two types of design are the two faces of the one thing and need better integration. The challenge of software development in the realm of immaterial labor is how to deal with the duality of design.

New types of software development skills and competence are called for. Lazzarato. Beller suggest that aesthetic mode of production is a starting point [6]. As in all aesthetic production, the assurance of non-functional quality comes from the practice of software developers, determined by their competence and motivation of making tweaks driven by the "love of beauty and greediness for the exquisite" [7].

A few design practices have started to address some of the issues related to the aesthetic modes of software development. The participatory design methodology [8] and socio-technical theory [9] have focused on realizing the quality-of-use that is only determined by users in their specific socio-technical context. The interaction design and experience design focus on the design of how users interact with a system while identifying necessary functionality for the system. This is in contrast with the traditional user interface design approach where an interface is inserted for the pre-determined functionality of the system.

These design practices need to be weaved more tightly with the existing software engineering research and practices. A currently predominant view of software engineering research states that while software engineering focuses on building a software system correctly, what we lack is a way to build a correct system. However, through the looking glass of the immaterial labor, there are no correct systems because their relevance is not yet determined. The true value of the system will be is evaluated by users through the using process.

4 Organizing Software Development

In organizing software development as immaterial labor, it seems important to follow several principles listed bellow for innovative project management.

Principle 1: Motivation

How to sustain software developers' intrinsic motivation becomes the first priority. Developers are the protagonists in their own development practices. The quality of software systems hinges on the individual selection of alternatives, and the fusion of subjectivity and tastes. Hall et al. have identified "challenge, change, benefit, problem solving, team work, science, experiment and development practices" as motivators for software developers [10].

Motivation becomes essential not only for technical exploration, but also for social cooperation essential for software development. Ad hoc coordination needs arise all the time during the process of software development; project members need to be sensitive to each other's information needs and to be motivated to help each other for collaboration to proceed timely and smoothly.

Famous German conceptual artist Joseph Beuys has claimed once: "Everyone is an artist." He did not mean everybody could be a painter or a sculptor. He notified that there is a chance for creative action within any human activity. In software development, every developer will encounter such a chance to show his/her creativity maybe in troublesome situation during project practice.

Principle 2: Polyphony

The concept of "Polyphony and Unfinalizability" proposed by Russian philosopher Mikhil Bakhtin seems important for technical management of software project. Bakhtin analyzed the works Fyodor Dostoevsky and found these principles. In Dostoevsky's novel, a number of characters make discussions about some profound "truth" of life. Dostoevsky treats voices of these characters complete equally. They keep going into endless dialogue and the discussion never comes to final conclusion even the story ends at some point [11]. Bakhtin found in Dostoevsky's work a true representation of "polyphony" (many voices). Each character in Dostoevsky's novel represents a voice that speaks for an individual self, distinct from others. This idea of polyphony is related to the concepts of unfinalizability and self-and-others, since it is the unfinalizability of individuals that creates true polyphony.

Bakhtin criticized the assumption that, if two people disagree, at least one of them must be in error. For him, truth is not a statement, a sentence or a phrase. Instead, truth is a number of mutually addressed, albeit contradictory and logically inconsistent, statements. Truth needs a multitude of carrying voices. It cannot be held within a single mind, it also cannot be expressed by "a single mouth". The polyphonic truth requires many simultaneous voices. Bakhtin does not mean to say that many voices carry partial truths that complement each other. A number of different voices do not make the truth if simply "averaged" or "synthesized". It is the fact of mutual addressivity, of engagement, and of commitment to the context of a real-life event, that distinguishes truth from untruth.

We, software engineers, have been discussing software process issues; process models, process improvement, process support environment, etc., etc. Our dialogue will continue forever. We should realize the unfinalizability of discussion and notice the importance of polyphony.

Principle 3: Difference

Traditional software engineering discipline has been constructed upon repetitive nature of material labor practices.

For example, in CMM model, the second level of software process maturity is named as "Repeatable":

It is characteristic of processes at this level that some processes are repeatable, possibly with consistent results. Process discipline is unlikely to be rigorous, but where it exists it may help to ensure that existing processes are maintained during times of stress.

The model consists of a number of key practice areas, which are repeatable and producing consistent results. The discipline says that if you are engaging similar projects repetitively, process activities should be well documented, well trained, and be effectively executable. It is appropriate for the process of material labor of production. But, in the case of immaterial software development process, it will rather causes suppression of developer's creative action.

French philosopher Gilles Deleuze pointed out that it will never happen the same things be repeated, rather the difference is repeated anytime.

He wrote:

That identity not be first, that it exists as a principle but as a second principle, as a principle become; that it revolve around the Different: such would be the nature of a Copernican revolution which opens up the possibility of difference having its own concept, rather than being maintained under the domination of a concept in general already understood as identical.

In software development, it looks like the same things are repeated again and again, but there is always some kind of differences happening. And there will be a big opportunity of innovation in these different situations. Project managers should encourage developers in their team be creative to find out the chance for innovation.

Principle 4: Rhizome

Felix Guatari and Gilles Deleuze. Proposed a metaphor of "Rhizome" in their book "Thousand plateaus"[13]. In botany, a rhizome is a horizontal stem of a plant that is usually found underground. A typical example of a network of bamboo root. Guatari/Deleuze used it as a metaphor of the image of human thoughts. They proposed that the concept of "Rhizome" could be used as a model of knowledge and a model of society.

As a model for culture, the rhizome resists the organizational structure of the root-tree system which charts causality along chronological lines and looks for the original source of "things" and looks towards the pinnacle or conclusion of those "things." A rhizome, on the other hand, ceaselessly established connections between semiotic chains, organizations of power, and circumstances relative to the arts, sciences, and social struggles.

The fundamental characteristics of "Rhizome Model" is as follows:

(1) Connection and Heterogeneity: any point of a rhizome can be connected to anything other, and must be.
(2) Multiplicity: only when the multiple is effectively treated as a substantive, "multiplicity" that it ceases to have any relation to the One
(3) A signifying Rupture: a rhizome may be broken, but it will start up again on one of its old lines, or on new lines
(4) Cartography and Decalcomania: a rhizome is not amenable to any structural or generative model; it is a "map and not a tracing".

I think that these characteristics are very much suitable to construct a new model of software projects for innovation.

Principle 5: Openness

Software project should have a technology window open to the outside world, and have a person who plays the role of this window. Through this window, project will show the result of activities inside, and also import various innovative ideas from outside.

It is a well-known historical fact that in ancient China almost all king imported new political ideas from wandering philosophers like Confucius or Mencius through the open gate of city wall. The success of the First Emperor was the openness of his country to the various talents wondering outside.

Let's go one step further. The way that open source software is developed is similar to what Lazzarrato suggests. It is a new style of project organization for the production of immaterial products: An OSS project team consists of anonymous members scattered on the Internet. Each member identifies and solves different problems on his/her own responsibility. Coordination is bottom-up, initiated and managed by members themselves, not by managers. Every OSS developer plays somehow as an entrepreneur. Variations of OSS-like projects could be adopted in corporation settings.

Agile development methods also reflect the view of treating project members as an independent body, and managing them as individuals instead of assigned roles.

Such recent trends in organizing software development projects tend to be regarded as new styles of software development. However, by viewing software development as immaterial labor, we can see that such trends are the reflection of a better understanding of the essential features of software and the change of the nature of the software systems that are developed.

5 Concluding Remarks

Software is an immaterial product with cultural and informational nature. It redefines the way we work, learn, communicate, and entertain. Most of Traditional software engineering disciplines have been constructed on the concept of material labor of production. . We need to switch to perspective of immaterial labor. This paper shows some guiding principles for new style of project management towards innovation.

References

1. Naur, P., Randall, B.: Software engineering: A report on a conference sponsored by the NATO science committee, NATO (1969)
2. Lazzarato, M.: Immaterial labour. In: Virno, P., Hardt, M. (eds.) Radical Thought in Italy: A Potential Politics, pp. 133–147. University of Minnesota Press, Minneapolis (1996)
3. Simmel, G.: The metropolis and mental life. In: Wolff, K. (ed.) The Sociology of Georg Simmel. The Free Press, New York (1950)
4. Virno, P.: A grammar of the multitude. The MIT Press, Cambridge (2004)
5. Teranova, T.: Free Labor: Producing Culture for the Digital Economy. Social Text - 63 18(2), 33–58 (2000)
6. Beller, J.: The cinematic mode of production: Attention economy and the society of the spectacle. Dartmouth College Press, Hanover (2006)
7. Lazzarato, M.: European cultural tradition and the new forms of production and circulation of knowledge (2004)
8. Ehn, P.: Work-oriented design of computer artifacts. Almquist & Wiksell International, Stockholm (1988)

9. Ye, Y., Yamamoto, Y., Nakakoji, K.: A socio-technical framework for supporting programmers. In: Proceedings of 2007 ACM Symposium on Foundations of Software Engineering (FSE 2007), pp. 351–360 (2007)
10. Hall, T., Sharp, H., Beecham, S., Baddoo, N., Robinson, H.: What do we know about developer motivation? IEEE Software 2008, 92–94 (2008)
11. Bakhtin, M.: Problems of Dostevsky's Poetics. University Press, Minneapolis (1984)
12. Deleuze, G.: Difference and Repetition, trans. Paul Patton. Colombia University Press, New York (1994)
13. Guatari, F., Deleuze, G.: Thousand Plateaus. Univ. of Minnesota Press (1987)

The Future of SPI Knowledge and Networking in Europe – A Vision

Richard Messnarz[1], Miklós Biró[2], Sonja Koinig[1], Michael Reiner[3], Romana Vajde-Horvat[4], and Damjan Ekert[1]

[1] I.S.C.N., Austria & Ireland
{rmess,dekert,skoinig}@iscn.com,
[2] Dennis Gabor College, Hungary
biro@gdf.hu
[3] University of Applied Sciences Krems, Austria
michael.reiner@fh-krems.ac.at
[4] Prohuman, Slovenia
info@prohuman.si

Abstract. Authors of this paper have been founders of the EuroSPI (1994 – now, www.eurospi.net) network with the first networking of SPI strategies published at CON'93 conference. They were also founders of the idea and establishment of a Europe wide certification network ECQA (www.ecqa.org) in 2005 (strategy development 2005 – 2007, online systems set up 2008 – 2009, Europe wide roll out since 2009). In a think tank and network of leading SPI experts we have developed the idea of a future European knowledge networking strategy and how the existing SPI paradigms will shift into a new SPI world applying new principles for collaboration, networking, and using new media which became available in the last 3-4 years.This vision will then impact about how we collaborate and implement SPI in the future.

Keywords: Process Improvement, Networked Organizations, Knowledge Strategies.

1 EuroSPI and the SPI Hype Cycle Paradigm

In 1993 at the CON'93 conference the idea of a European expert network of SPI was presented the first time. The idea led to the first EuroSPI conference in 1994 in Dublin and further conferences till now. The basic concept of EuroSPI network is that a continuing exchange of SPI knowledge between industry and research is supported to assure SPI implementation, base SPI work on real practice and form a Europe wide agenda and movement with partners world-wide. From each conference ideas were created and a pool of experiences has been set up and made available as online library.

This strategy was based on the SPI Hype Cycle Paradigm which will continue to create a library of SPI practice for European industry, to increase European competitiveness. It is typical in research and industry that any new concept runs through an

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 268–277, 2011.

innovation hype cycle (ref. the idea of the Hype Cycle was introduced by Gartner in 1995). This hype cycle includes the following stages:

1. Technology Trigger
2. Peak of Inflated Expectations
3. Trough of Disillusionment
4. Slope of Enlightenment
5. Plateau of Productivity

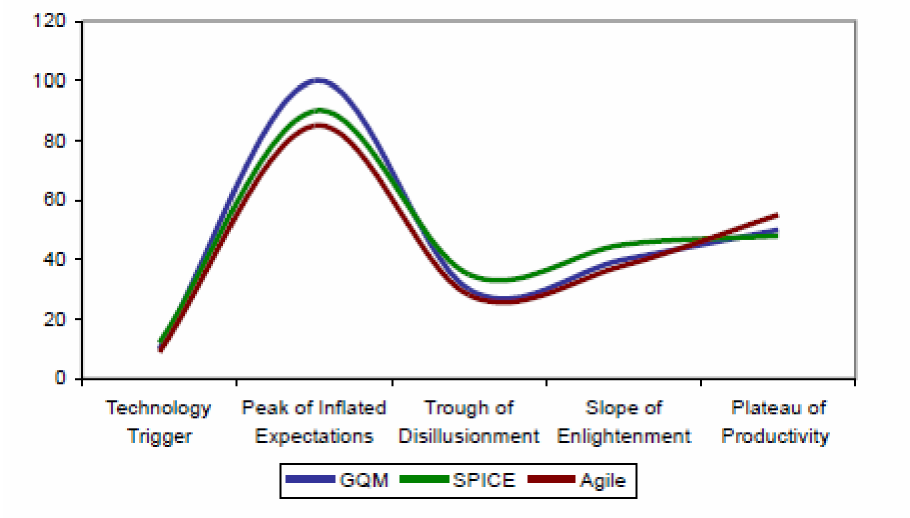


Fig. 1. The SPI Hype Cacle Based EuroSPI Network Strategy

Especially in SPI, we could observe many methods and concepts to come and claim that they represent the silver bullet to solve all problems. When we observe the last 20 years of SPI, it is actually not true that we ever found one single solution that solved all problems. It is rather as that all new concepts and methods ran through a hype cycle and were integrated later on a plateau of combined use of methods and concepts based on industrial feedback and experience.

Here are some examples:
In the early 90s there was a competition between the GQM (Goal Question Metrics) approach and the assessment methods. Nowadays people do not really remember this era any more and consider it natural, that in assessments and improvement methods we use techniques to align business with improvement goals and metrics and track SPI actions.

In the early 2000s there was the new strategy of teamwork based environments to apply processes organisation-wide. Meanwhile people consider it natural that Wikis, teamwork systems and process libraries support assessments, improvements and goal tracking (GQM).

Over the last years a strong agile community has emerged pointing out the higher value it attributes to approaches which were less favoured by earlier ones. However looking at 20 years of experience in the EuroSPI community, it is already visible that practical experience from industry shows its impact to enter the plateau of productivity. Leading engineering firms will hold this year a workshop about 14 practices, that really worked in agile development and outline what does not work.

Over the last years, a safety engineering community came up with safety standards and claimed the superiority of their own approaches. Since 2009 methods to combine SPICE and safety standards have been published.

What EuroSPI really makes interesting is, that we formed a platform for experience exchange, empowering new ideas (supporting researchers) but also collecting realistic practices on the plateau levels for industrial implementation (providing support to industry).

In [1] we published the first ideas in 1993, in [3] we published a work of 30 European experts in the US with a summary chapter about European developments and their future, and in [1] the SPI hype Cycle strategy was summarized.

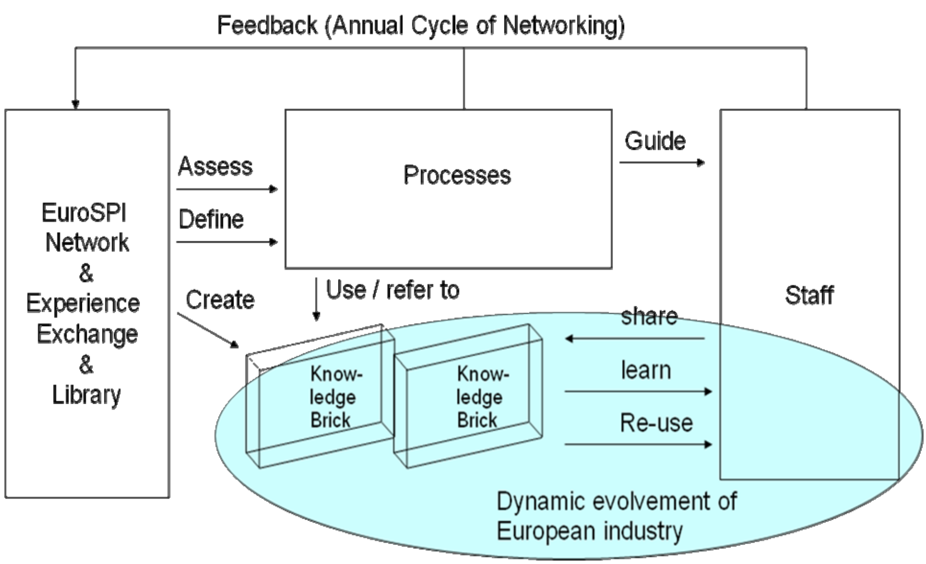


Fig. 2. The SPI Hype Cacle Based EuroSPI Network Strategy

Since 1993 in an annual cycle experiences are exchanged, the SPI library is extended, the partnerships are created and implementation in European industry is enforced.

2 SPI Networking Strategy to Package and Transfer Knowledge in Europe – The European Certification Network

After nearly 10 years EuroSPI the question in 2005 was how to transfer the existing experience library to the market and how to speed up the take up in industry. Thus in 2005 a new European strategy was created to package knowledge (experience papers based on similar topics) into skills sets, training materials and create (like the US strategy by PMI and related institutes) a European certification system around that topics.

In 2005 to 2007 [4], [5], [6] the strategy was developed in the EQN (European Quality Network Project with 13 partners from 8 countries), in 2008 – 2009 the online support system was established to support Europe wide training and certification (EU Cert Campus Project with 23 partners from 14 countries),and 2010 – 2011 local representatives across Europe are established in the DEUCERT project.

We then have achieved more than 10000 certifications since 2005 (the start), more than 40 training organizations supporting, more than 20 European consortia developing the different knowledge packages / professions.

2.1 European Certification and Qualification Association (ECQA) Platform

If there is a need a person can attend a course for a specific job role online through an advanced learning infrastructure [5]. See Figure 3.

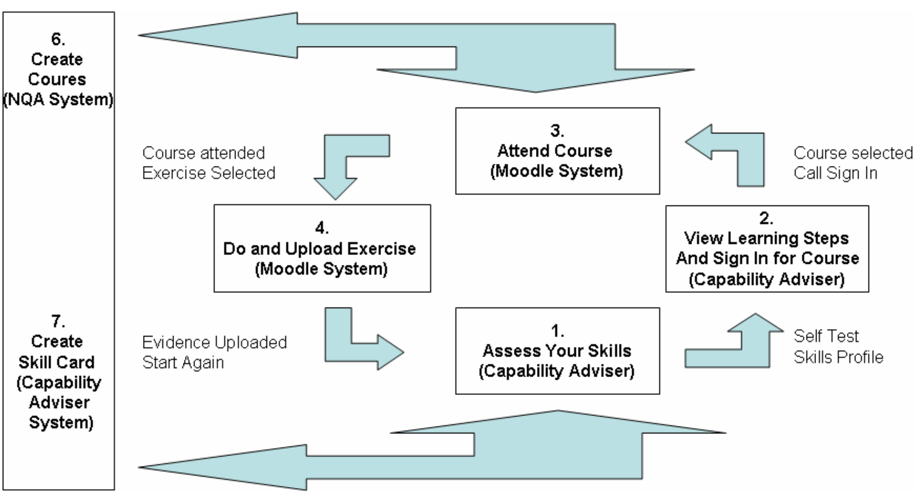


Fig. 3. The Integrated European Skills Acquisition System

You start with a self assessment against the skills [2], [5], [12]. Then you can sign into an online course. Here you are guided by a tutor and do a homework which is being corrected by the tutor. Finally the homework and real work done in your project is sufficient to demonstrate the skills.

Moodle – This is a web based learning management system which is public domain available. (www.moodle.com)

Capability Adviser – This is a web based assessment portal system with a defined interface database to connect the systems. (http://www.capability-adviser.com) [12]

NQA – Network Quality Assurance – This is a web based team working tool which was developed in the EU IST 2000 28162 project. [5]

So far the following profession have been configured –

- ECQA Certified E-Learning Manager
- ECQA Certified EU Internal Financial Control Assessor
- ECQA Certified EU Project Manager
- ECQA Certified Governance SPICE Assessor
- ECQA Certified Incubation Manager
- ECQA Certified Innovation Manager
- ECQA Certified Integrated Design Engineer
- ECQA Certified IT Consultant for SMEs
- ECQA Certified ISECMA© Professional for IT-Security Management
- ECQA Certified Lean Six Sigma - Yellow Belt (in development)
- ECQA Certified Lean Six Sigma - Orange Belt (in development)
- ECQA Certified Lean Six Sigma - Green Belt (in development)
- ECQA Certified Lean Six Sigma - Black Belt (in development)
- ECQA Certified Researcher-Entrepreneur (in development)
- ECQA Certified SCOPE Manager
- ECQA Certified Social Responsibility Manager (in development)
- ECQA Certified SPI Manager
- ECQA Certified Terminology Manager – Basic
- Etc.

See www.ecqa.org.

Since 2005 job role consortia were formed from the networked pool of experts and organizations which packaged the knowledge into more than 20 key professions for European industry and formed a European certification system and a pool of training bodies (see Figure 4).

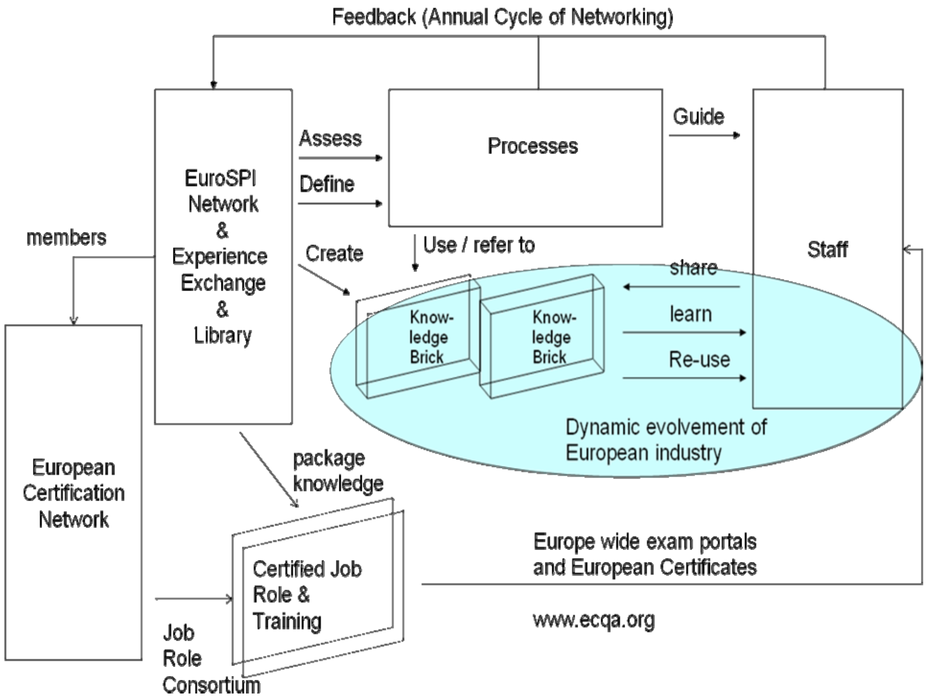


Fig. 4. European Networking Strategy – Evolution Step 2 – Transfer Phase

3 Future SPI Networking Strategies – The Know-Net Strategy

Since 2010 the partners elaborate a evolution for the SPI (innovation and improvement) strategy on a European level which is called Know-Net. The Know-Net strategy is based on the following key principles. The idea is to create a set of rules and infrastructure that allow social media based learning clusters that can dynamically evolve around EuroSPI and ECQA key topics (= topics needed to help European industry to increase their competitiveness).

Clusters of Experts [1], [2], [3], [5]:

The old paradigm of knowledge management assumes that we store all knowledge and implement more and more sophisticated algorithms to search that knowledge and give advice. The new paradigm which we follow in expert and knowledge networks does not believe in that because the tacit knowledge is much larger than the stored one so that any decisions made on just the stored knowledge are questionable in an expert network.

Thus we create social web based expert clusters around key industry topics. Key articles are stored around topics and experts can be connected from there. Instead of a query to a database we use a query to experts clustering around key topics.

It's like the Google principle transported to the European improvement and innovation network of experts idea.

Network Building [1], [2], [7], [9], [10]:

We started with creating one network around EuroSPI. In ECQA we created above 20 sub-networks (Job Role Committees) supporting specific packaged know how. We plan to create a critical mass of further topic driven sub-networks that interface with the main network. You can imagine that like a facebook like number of clusters which have a common middle.

International Multicultural Teams/Networks [7], [9], [10]:

Europe is different from e.g. the US because in Europe we speak many different languages, have different traditions and history, and inside the EU we support the development and independence of regions. This means that even if we will have the same population in total we still will have to deal with many different cultures.

We will need to integrate this by creating a library for cross cultural collaboration experiences and illustrating success stories of cross country collaborations across all European countries and to partners from Japan, India, Middle East, Russia and the US (with European interest).

Continuing Willingness to Learn:

Creating a network and topic based Knowledge network means that we base on people and personal skills of people and experts. People with the ability to understand the forces of change, accepting innovation and knowing that continuous learning is a major success factor will become the main players in the network.

This will be joined by many hundred managers from ECQA (e.g. innovation manager) job roles who implement principles to create system design, innovation, and learning strategies on organizational level.

Continuous Contact to Customers [5]:

It is important to have a mix of industry and research. EuroSPI has meanwhile 4 publishers supporting (Springer for research, Wiley and IG Global and IET Software for experiences from applied research and industry) and also the social web based knowledge clusters around key topics will need to be driven 50/50 by research and industry.

Create Customer Needs in Advance [5]:

European industry is driven by lead engineering companies who themselves have innovation departments. It is important to include their vision in the knowledge topics so that it is possible to create a critical mass of research support in advance for future engineering visions in Europe.

The closers the innovation cycles are to industrial innovation of lead companies, the better the knowledge network can serve strategic interests in the European industry.

Continuous knowledge and idea exchange and management (= knowledge sharing [11]):

A social media based knowledge net around required key topics for industry will support the forming of expert clusters, where industry and research can meet, elaborate joint ideas, implement, and publish the parts which can become public at the European conference.

Foundation of Trust [5]:

Different to the Facebook like architecture European SPI and knowledge networks will need a private (not shared and protected for industry interest) and public space (open for the world). While facebook has this separation, in European expert networks it is planned to make this separation more secure. Only if the industry has trust in the protection of knowledge the industry collaboration is assured in the private space.

For the key experts in the private sphere this will mean increasing contact to industry implementation inside the private space.

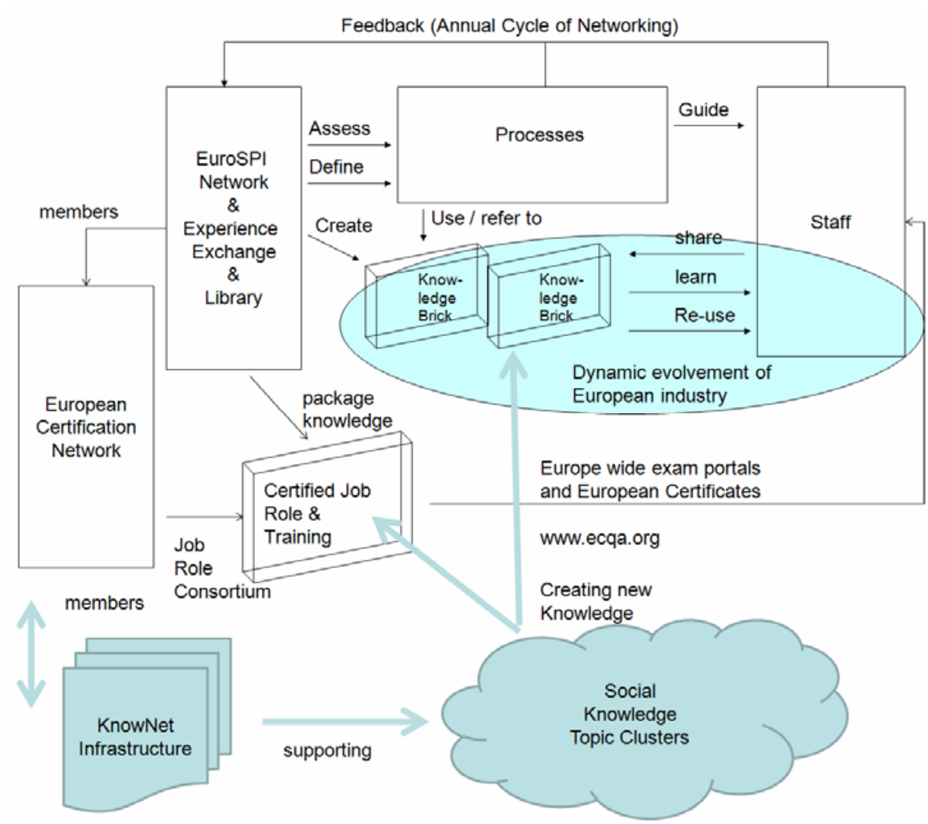


Fig. 5. European Networking Strategy – Evolution Step 2 – KnowNet Phase

Social responsibility & ethics:

Working inside a social web community, dealing with knowledge, requires an extensive understanding of social responsibility and the code of ethics. We have currently a number of research thesis running to support creating this basic rules for participating in the future European Knowledge Net for SPI.

Idea pool and later exploitation:

It is now enough to just share knowledge, or create a social media based knowledge net around required key topics for industry. The critical mass of experts around topics shall also function as a think tank for new ideas. New ideas shall be transported to / disseminated by the annual event, the job role committees of ECQA, and shall lead to new initiatives and new key topics in periodical innovation cycles.

4 Future Outlook and Implementations

4.1 Future Systems

In 1993 EuroSPI was the first conference to start SPI networking. After 1994 the EU ESSI program financed many conferences across Europe to do similar things. In 2005 the idea to create a Europe wide certification network which is recognized seemed nearly impossible competing with existing powerful networks from US. Now 7 years later it is reality.

The idea to create such a modern European knowledge net strategy seems big and maybe some will say (as it happened in the past) that this is too big and not working for Europe. However, considering our success record of EuroSPI and ECQA in the past 18 years, for us it is only 3 to 4 years that you will see this strategy alive.

This means that the infrastructure underlying ECQA and EuroSPI will need to evolve by including more social media web based and knowledge net based functionality. It also means that we need to elaborate social responsibility and ethics rules for the network.

4.2 Impact on SPI

SPI implementations will become like Java programming nowadays. Twenty years ago a programmer first structured the program, then coded each function, then integrated and built the code, and tested the executable. Nowadays we have a MS Developer framework, Java libraries, public domain applications and the same programmer can create in less time many times more code by re-using existing libraries and structures.

Lets assume that SPI is like Google and we get like in cloud computing a proper selected set of methods and experiences by accessing an expert cluster.

This shall speed up the innovation capacity of European research and industry.

4.3 Continuously Provoking Radical Innovation

The previously described SPI innovation cycles in Europe must continuously get fuel to turn the wheels. Innovation does not happen if we do not contradict to the traditional ways of work. So it makes sense to create focused workshops to motivate the next innovation cycle by e.g. organizing workshops for brainstorming like “We have done agile – what comes next?, which is done currently by the Scandinavian partners of the EuroSPI network.

Acknowledgements. These ideas were developed in a collaboration with existing networks in Europe and their active partners. This includes ECQA (www.ecqa.org, European Certification and Qualification association), EuroSPI (www.eurospi.net, European systems and Software Process Improvement and Innovation), SOQRATES (cross company task forces of above 20 leading companies), and the recent EU projects supporting the job role consortia development.

References

1. Biró, M.: The Software Process Improvement Hype Cycle. Invited contribution to the Monograph: Experiences and Advances in Software Quality. In: Dalcher, D., Fernández-Sanz, L. (Guest eds.) Cepis Upgrade, vol. X (5), pp. 14–20 (2009)
2. Messnarz, R., Biró, M.: ECN – European Consulting Network. In: Proceedings of the CON 1993. Oldenbourg Verlag, Szombately (1993)
3. Messnarz, R., Biró, M., et al.: Better Software Practice for Business Benefit. IEEE Computer Society Press, Washington (1999)
4. Messnarz, R., et al.: Assessment Based Learning Centers. Wiley SPIP Journal 12(6) (June 2007)
5. Messnarz, R.: From process improvement to learning organisations (p. Wiley SPIP Journal (Software Process Improvement in Practice) 11(3), 213–335 (2006)
6. Messnarz, R., Ekert, D., Reiner, M., O’Suilleabhain, G.: Human resources based improvement strategies—the learning factor. Wiley SPIP Journal 13(3), 355–362 (2008)
7. O’Suilleabhain, et al.: The Perception of Quality Based on Different Cultural Factors and Value Systems. In: Proceedings of EuroSPI 2000 (2000)
8. Siakas: Agile Software Development in Distributed Environments. In: Proceedings of EuroSPI 2009 (September 2009)
9. Siakas, K.V., Balstrup, B.: Software outsourcing quality achieved by global virtual collaboration. Wiley SPIP Journal 11(3), 319–328 (2006)
10. Siakas: Process Improvement: The Societal Iceberg. In: Proceedings of EuroSPI 1999, Pori, Finland (1999)
11. Fehér, P., Gábor, A.: The role of Knowledge Management Supporters in software development companies. Wiley SPIP Journal 11(3), 251–260 (2006)

Innovation Managers 2.0: Which Competencies?

Andreas Riel

G-SCOP UMR5272, Grenoble Institute of Technology (Grenoble-INP)
Grenoble, France
Andreas.Riel@grenoble-inp.fr

Abstract. This keynote paper investigates modern trends in Innovation Management in industrial companies with the ultimate objective to identify key competencies of Innovation Managers. It aims at pinpointing new innovation management challenges that have evolved in product development and manufacturing industries. As tackling these challenges successfully demands specific competencies, this article can serve as a guideline for establishing Innovation Manager competence specification, as well as training and certification programs. It can also help company executives design career paths of Innovation Managers.

Keywords: Innovation Management, Sustainable Innovation, Open Innovation, New Product Development, Integrated Development.

1 Introduction

As industrial organizations are facing increasing innovation pressure, effective innovation management has become indispensable for them, and requires dedicated job roles with specific competencies and skills. Due to the importance of the subject, attempts to define, train, and certify these competencies have become frequent. To date, there exists only one Europe-wide accepted training and certification program for Innovation Managers [1]. Departing from this program, which has been established by an IT- and software centric international consortium of SMEs, this research investigates new competence requirements for Innovation Managers in industrial organizations. The research methodology is to derive them from key trends in innovation management as it is defined by a recently published, well validated framework, the reasoning being that implementing these trends as an innovation manager within a particular organization requires specific competencies.

This paper is structured as follows: Section 2 introduces the innovation management framework that has been used to guide this research. Section 3 zooms on the three elements that constitute the basic layer of this framework, idea management. Section 4 deals with innovation organization and culture, which together form the second layer of the framework, upon which the innovation strategy can be built as the top covering layer. Section 5 derives competence requirements for innovation managers from the trends identified for each of the framework's elements. It also considers the relevance of designing career paths for that job role. Finally, section 6 concludes the paper and gives and outlook.

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 278–289, 2011.

2 A Framework for Innovation Management

Innovation may refer to incremental, emergent, or radical and revolutionary changes in thinking, products, processes, or organizations. Following the OECD manual [2], which adopts the notion of innovation inspired by Schumpeter in 1930, contributors to the scholarly literature on innovation typically distinguish between invention, an idea made manifest, and innovation, ideas applied successfully in practice. In many fields, such as the arts, economics and government policy, something better must be substantially different to be innovative. In economics the change must increase value, customer value, or producer value. Innovation is an important topic in the study of economics, business, entrepreneurship, design, technology, sociology, and engineering.

Invention is the embodiment of something better and, as a consequence, new. While both invention and innovation have "uniqueness" implications, innovation is related to acceptance in society, profitability and market performance expectation. An improvement on an existing form or embodiment, composition or processes might be an invention, an innovation, both or neither if it is not substantial enough. According to certain business literature, an idea, a change or an improvement is only an innovation when it is put to use, is accepted by users and effectively causes a social or commercial reorganization.

In business, innovation can be easily distinguished from invention. Invention is the conversion of business into ideas. Innovation is the conversion of ideas into business. Therefore, any innovative activity has to take into consideration the underlying interdependencies as well as the company´s competitive strategy. At any stage of development innovative decisions have to be consistent with business strategy. Central factor of competitive strategy is the choice of the market position and it´s realization.

According to one of the most extensive recent European studies [3] Innovation Management is the capability to continuously manage inventions/ideas for new products or services, processes, production methods, organizational forms or elementary improvements of a business (model) system and their successful realization. The important part of this definition is the term: *up to its successful realization*. “Successful” is defined in the business environment by the business success resulting in sustainable income and profit growth. All the dimensions of Innovation Management have to be directed to the overall goal of sustainable business impact and growth.

More than 1,500 small and medium sized enterprises (SME) from all over Europe were involved in the testing and validation of the concepts given in [3]. According to those, all dimensions of innovation management are geared to profitable growth. This is manifested in [3] in the so-called “A.T. Kearney House of Innovation”, which is shown in Fig. 1.

Although there were only SMEs involved in this particular study, the elements of the proposed innovation management framework are sufficiently general to be applied equally well to large enterprises. Only their particular challenges are somewhat different, as will be pointed out later.

The framework’s four dimensions of innovation management, which are all geared to increase the innovation and business performance and to drive a company’s profitable growth, are characterized in [3] as follows:

Fig. 1. A.T. Kearney's House of Innovation [3]

- The *Innovation Strategy* identifies the most promising areas where the company can achieve superior profit growth rates either with new products/services or with existing products/service in new markets or with new or improved processes or business models.
- The company's *Organization and Culture* must support this innovation strategy so that the profit growth targets can be reached. Companies must have the structures, for example, to integrate external partners in their development processes or to seamlessly manage the development processes. Their culture must be open to new ideas no matter where they come from. The organization has to translate the innovation strategy to pursue those ideas that are most promising for their focus areas.
- In the *Innovation Life-Cycle Management* there are many steps where leading innovators avoid inefficiencies and ensure short time-to-profit, while the average company might only focus on the time-to-market and forget about proper life-cycle management after the launch of the innovation.
- *Enabling factors* such as knowledge management or capabilities in specific technologies or expertise in new market development also have a significant impact on growth through innovation management. They must be aligned with the company's innovation strategy, allocated in the right manner in the organization and leveraged for successful innovation management to fully exploit the growth potential of the innovation.

In this paper, A.T. Kearney's House of Innovation will be used as the principal basis for the investigation of Innovation Management trends in industrial companies. Furthermore, the management of radical or breakthrough innovations will be the focus of all considerations, as this is considered the biggest challenge for Innovation Managers.

3 Innovation Life Cycle Management

This section investigates the trends in the three elements that constitute the Innovation Life Cycle Management layer of the House of Innovation: Idea Management, Product/Process Development, and Launch/Continuous Improvement. From these trends, competencies required for innovation managers to master and control this innovation life cycle can be derived.

3.1 Idea Management

One of the most cited recent papers in the context of modern innovation management [4] identifies the Idea Management element of the House of Innovation as key part of Innovation Management. They argue that boosting a company's innovation strategy by sticking to best practices is not the right way to go. Best practices are always context-dependent, and every company has unique innovation challenges. So another firm's best innovation practice could become another's worst nightmare. In order to avoid this erroneous path, they recommend viewing innovation as a value chain comprising three phases:

- *Idea generation*,
- *idea conversion*, and
- *idea diffusion*.

This model thus covers the complete life cycle of ideas, from their generation over their realization to their capitalization on the market. Key issues across all those phases are internal, external, and cross-unit collaboration of stakeholders, and their involvement in the whole idea management process. Following Freeman, stakeholders are defined as "any group or individual who can affect or is affected by the achievement of the organization's objectives" [5]. Although it should be interest of a company to maximize the sources of ideas, traditional innovation management schemas are much closed in terms of idea management, effectively enabling only a few stakeholders inside the organization to participate in the process effectively. The same applies to idea selection, which is nowadays often largely restricted to some individuals, typically innovation managers or the managers. However, even if the focus of innovation is originally on economic and technical criteria, modern and sustainable innovation management is also increasingly challenged by ecological and societal issues. Nowadays, no innovation can be sustainable without meeting key challenges of our modern environment, and today's society. The main consequence for the innovation management function within an organization is that it has to understand and take into account challenges imposed by the environment, i.e., by ecology, economy, and society, and take this knowledge into account in idea selection and conversion. As it is practically impossible that this knowledge can be concentrated in one person and job role, modern sustainable innovation management has to be built on a system rather than on the decision of some individuals. This system has to integrate a large number of stakeholders inside and outside the company all along the innovation management process, each of them having different expertise and views on the complete product/system life cycle [6].

Any weak link between the three idea management phases can break the company's innovation efforts, so the focus has to be set on pinpointing and strengthening the company's deficiencies in that process. To remedy deficiencies lying in idea generation, building external and/or cross-unit networks is recommended. Weaknesses in idea conversion can be overcome by creating cross-unit funding and creating some dedicated safe environment for the incubation of ideas. Idea diffusion is also leveraged by internal and external networks, as well as some designated "Idea Evangelists", who have the capability and the mission to convince customers and/or development and distribution partners of the idea.

3.2 Product/Process Development

As a response to global competition, industrial companies are increasing the introduction of technologically sophisticated products as well as the adoption of advanced technologies and changes in organizational structure and processes. A company's competitive position is determined by the ability to innovate its product and service portfolio in the cycle time demanded by the markets that they address. However, they also have to assure their ability to ramp up to full scale production volume at the required speed and with the demanded quality. Therefore, for industrial companies, product and services innovations have to go hand in hand with innovations of the related processes.

For industrial companies innovations of the product system as well as innovations of the manufacturing processes are essential competitive factors. Due to technological facts there is a tight relationship between technical products and the processes implemented to realize these products, much less tight than in the software industry. Innovation management has to take into account the dynamics of the underlying product-process interactions and the resulting constraints [7]. This close coupling of products and the associated manufacturing processes has to be taken into account from the very early design phases of the product [8], leading to the need of integrated product-process design and innovation [9].

Particular challenges to innovation are posed by New Product Development (NPD), which is typically done in the context of creating radical innovations. Contrary to incremental innovations, which add value to existing products, radical innovations imply significant new challenges to the organization, mainly in terms of competencies and skills, and market knowledge and experience. The key issue is that if a 'new product' is really 'new', then it automatically means that the designer or design team must go beyond a simple extrapolation of previous designs or products. Otherwise, this is merely product development but not *new* product development. By stepping into the domain of the new, the NPD process is necessarily creative and not simply mechanistic. This means that the process at the initial time has different possible outcomes, and these will be explored to a greater or lesser extent by the NPD process. This process must therefore be one in which a new 'bundle' of components are discovered that give rise to a new set of performance attributes that satisfy the customer. This is why it is important that the customer and the design team interact sufficiently for the designers to show some of the possibilities, and for the customers to reflect on which new bundle of performance attributes is most attractive to them. In [10] it is shown that complexity science can significantly help to adopt a holistic perspective to innovation processes.

Complexity science is also the key to showing the necessity to adopt the notion of architecture to NPD. Up to now the notion of architecture has practically not been used in the context of technical product design outside the building construction sector. In software, however, the terms "architect" and "architecture" have been used for a long time [11]. Riel has shown in [12] that integrated development and product architecture are closely related to the architecture concept adopted in software and systems engineering, and that product architecture has a significant influence on product and system complexity.

Another trend in product/process development requires special consideration b innovation managers. In the IT and software sector, there is clear trend towards providing software as a service (SaaS) rather than as a product that can be installed and used on a single computer. While this trend was initially mainly driven by technical arguments, new business models have come up which are based on the idea of selling the value of the product in use rather than selling the product itself. This allows the creation and selling of value-added services to customers all through the time they have access to the software services. Service Oriented Architecture (SOA) denotes the answer on this trend from an architectural design point of view. Providing software as a service imposes some prerequisites to the architecture of the software as well as the enabling IT infrastructure. These architectural challenges have to be addressed explicitly in the design of these systems.

In increasingly many industrial sectors a very similar trend can be observed. The economic target of companies is to support and be involved in value creation on customer side as long as possible, which implies that they have to conceive and support services strongly associated with the products that they sell, and which go far beyond maintenance and repair. Thus, in an economic sense, the margins that are achievable with all other kinds of services which can be delivered throughout the whole life cycle have to be exploited. Products are becoming enablers of services, and their evolution is strongly linked all along the life cycle.

The need to co-design products and services introduces new challenges to design, and has led to a new field of close collaboration between industry and academia. It is called Industrial Product-Service Systems (IPS²), or simply Product-Service Systems (PSS). Research in this field aims at establishing methods that allow putting in place an integrated industrial product and service offering that delivers value in use. In [13] Aurich et al. present a concept of the evolution of a purely product-oriented design strategy towards a strategy which integrates the consideration of product, production, and associated services issues throughout the complete design process.

3.3 Launch/Continuous Improvement

From the point of view of Innovation Management, ideas that have entered this phase are likely to be in a safe haven. Industries have developed quite stable and structured processes for this phase of launching the product and ensuring incremental innovation. Nonetheless, the launching of a radical or breakthrough innovation poses specific challenges both from the interior and the exterior of the organization. In the internal dimension, such innovations require completely new knowledge and/or resources. From an external point of view, new technological and market challenges have to be captured and mastered.

Therefore, innovation management has to assure that the organization can learn rapidly from the experiences from launching a breakthrough innovation, and from market feedback. It is thus important to establish a learning organization environment

[14] to accelerate the transition from a situation of radical innovation to a context of incremental innovation, the latter building upon existing knowledge and resources within the company to further enhance competences, as well as on known markets.

4 Innovation Organization and Culture

Innovation, whether related to products, process, organizational methods, or marketing, is a complex, multidisciplinary activity that involves several areas of a single firm (such as Marketing, R&D, Manufacturing, Finances, etc.), its clients, and its suppliers. In order for this system to function effectively, an active coordination and management of the different activities it entails is required. Organization forms are sought that facilitate and leverage the innovation life cycle management.

4.1 Stakeholder Integration

As was shown in research on integrated product and system design [15] and New Product Development [16], integrating stakeholders of the complete product/system life cycle throughout the entire product/system development process from the earliest phases on, is the key to creating sustainable innovation. The sustainability aspect is leveraged by the fact that only the integration of different views on the product/system in terms of its functions and its economic, ecologic, and social environment allows to identify requirements and constraints on the product/system in a holistic manner, and therefore to take them into account both in the composition of development teams, as well as in the design and architecture of the product/system. The same issue applies to idea generation and assessment, which is part of the earliest upfront phases in the product/system life cycle.

When it comes to product and process innovation, the most successful companies are those whose organizational structures foster the development of knowledge through formal research and development processes and the development of knowledge based on experience, practice, and interaction between employees, clients, and suppliers [17]. In general, it has been observed that big companies are much better at incremental innovation than they are at radical innovation, despite countless programs aimed at strengthening innovation capabilities. The study [18] of 21 large companies' efforts to build a capability for breakthrough innovations over several years reveals that even though companies declare innovation as important strategic target, most fail to provide the formal structure and support that programs need to succeed. It turns out that the classical organizational configurations are not very appropriate for companies that have superior product and process innovative performance as their organizational strategy. More flexible and agile structures are required, structures that allow interaction and communication between employees, without rigidly defined functional areas, and with functional integration instead.

In his landmark book on organizational configurations [19], Mintzberg states "adhocracy" is strongly connected to providing innovation. It is not inspired by classical principles, and is particularly distant from the concepts of unified command, high behavioral formalization, and planning and control systems. It is defined as follows:

- Organic structure, made up of ad-hoc project teams;
- Low degree of formalization;
- High degree of horizontal specialization of labor, based on formal individual knowledge;
- Mutual adjustment between teams, without the need for formal coordination of roles;
- No standardization of products or processes;
- Decentralized decision-making for inter- and intra-team activities.

This “adhocratic” or organic structure would permit the development of knowledge based on practical experience and interaction, consequently leveraging the organization’s innovative capacity [17]. This organizational configuration would also be the most readily able to handle events, i.e., to deal with unforeseen actions and chance occurrences, which are characteristic of innovative environments.

4.2 Open Innovation

In classical industrial organizations, innovation processes have been dominated by the so-called Innovation Funnel model. This model is essentially based on the fact that innovation is driven and controlled exclusively by stakeholders which are internal to the organization. This paradigm can be called Closed Innovation, and it says successful innovations require control. Companies must generate their own ideas and then develop them, build them, market them first, distribute them, service them, finance them and support them on their own. It counsels firms to be strongly self-reliant, as it is impossible to be sure of the quality, availability and capability of others’ ideas. Consequently, this view also suggests that companies should hire the best and the brightest people, so that the smartest people in their respective industry work for them. Furthermore, intellectual property has to be strictly controlled in order to avoid that competitors can profit from the company’s ideas.

Open Innovation is the opposed paradigm that assumes firms can and should use external ideas as well as internal ideas, and internal and external paths to market, as the firms look to advance their technology. Both external and internal ideas are used to create value, and internal mechanisms are defined to claim some portion of that value. Open Innovation assumes that internal ideas can also be taken to market through external channels, outside the current businesses of the firm, to generate additional value. Ideas can also start outside the firm’s own labs and can move inside. Open Innovation allows the recovery of overlooked innovations, which increases the chance for projects will find value in a new market or be combined with other projects.

Open Innovation has been coined by Chesbrough in 2003 [20], although the paradigm has been around in some industries for a long time. A stereotype example is the Hollywood film industry, which has innovated for years through a network of partnerships and alliances among production studios, directors, talent agencies, actors and scriptwriters. Many industries are in transition between the two paradigms, e.g., automobiles, biotechnology, pharmaceuticals, healthcare, computers, software, communications, banking, insurance, and consumer packaged goods. The locus of innovation in these industries is moving beyond the confines of the central R&D laboratories of the largest companies to start-ups, universities and other outsiders. In so doing, the company can renew its current business and generate new business, capitalizing on abundant distributed knowledge resources.

By its very definition, Open Innovation leads to networks of people, companies, and other different kinds of organizations. The management and coordination of such networks requires specific competencies which are not relevant in closed innovation organizations. Moreover, new metrics have to be found which allow the performance of such innovation networks in terms of several criteria. This is a very important subject of research in management and economy. An exhaustive overview of the state of the art is given in [21]. The Open Innovation paradigm is the basis of more specific derivates like Coopetition [22] and Crowdsourcing [23], and has also become a key concept for tackling the challenges of economic crisis [24].

5 Innovation Manager Competence Requirements

A comparison of these observations with the exiting IT-oriented Innovation Manager competence specification [1] immediately reveals that the latter is very much focused on the fundamental basis upon which the House of Innovation is built, i.e., the enabling factors. Competencies linked to the house itself are largely underrepresented. This section attempts to identify and categorize key competencies linked to the identified trends, with a particular focus on the context of implementing radical, or breakthrough innovations, as they impose particular challenges to innovation managers.

Radical innovations share a lot of characteristics with entrepreneurial activities: development of new businesses, products and/or processes that transform the economies of a business, exploration of new technologies, teaching the market about the new technology and learning from the markets how valuable that technology is in that application arena, business model and plan evolving through discovery-based learning. Idea generation and opportunity recognition occur sporadically, often in discontinuities in the project directory. Key players are cross-functional individuals, and informal networks—increasingly facilitated by social networks in the Web 2.0. Rapid learning is essential in the creative acquisition of competencies and resources from a variety of internal and external sources.

The previously mentioned study of 21 large companies' efforts to build a capability for breakthrough innovations over several years [18] makes a significant contribution to nailing down, categorize, and specify required competencies. The research linked to this study reveals that at present companies tend to fundamentally mismanage their innovation talent. Typically, large companies rotate high-potential managers in and out of the innovation leadership role on a regular basis. That may give the rising stars broad experience, but it deprives the company of any real innovation expertise at a senior level. Even more damaging, companies do not provide meaningful growth opportunities for their innovation professionals. Thus, although there are plenty of great jobs in innovation, there are no careers.

The study points out that it is essential for companies to realize that breakthrough innovations consist of the following three phases:

- *Discovery*: Creating or identifying high-impact market opportunities.
- *Incubation*: Experimenting with technology and business concepts to design a viable model for a new business.
- *Acceleration*: Developing a business until it can stand on its own.

This consideration is a key to a competence analysis of Innovation Managers, as each phase requires unique specific competencies [25]. It resembles very much the process of creating an enterprise; the competences of Innovation Managers are thus susceptible to resemble those of entrepreneurs:

During *discovery*, scientific or technological experimentation is required to find out about how an innovation might satisfy a marketplace need. The mission is thus to create and identify market opportunities, and to explore the fit between technological capabilities and market needs. In this phase, Innovation Managers' competencies are focused on scientific and technical issues, however always in close link with market comprehension, systems thinking and understanding, network developing, opportunity identification and widening, coaching teams in strategically thinking.

During *incubation*, employees experiment recursively with technology and market opportunities and try to anticipate the impact the breakthrough business may have on the company's strategy. Products, processes and associated services have to be co-developed with respect to profound architectural considerations which are targeted towards creating value for the customer and the company. The target is to create new business that delivers breakthrough value to the customers and the company itself. Competencies needed are focused on creating new business, as well as on rapidly assimilating and capitalizing on new information. Multidisciplinary team composition, integration and coaching, resource identification, as well as managing the new business in the context of the larger organization's ecosystem are further key qualifications. Especially this phase demands the agility of changing direction whenever needed.

During *acceleration*, established business capabilities such as scaling up processes, imposing discipline, and specialization are needed. The new business has to be nurtured until it can stand on its own. This mainly demands traditional leadership skills either for a specific function, or as a general manager for a high-growth business. These skills, however, increasingly have to be applied to networked organizations following the Open Innovation paradigm with respect to a large variety of both internal and external stakeholders.

As is further pointed out in [25], each of these phases lends itself to distinct career paths. Scientists involved in discovery, for example, may eventually want to be involved in policy discussions about emerging technologies and how they may influence the company's future. The incubator may want to pursue a technical path—managing larger, longer-term projects—or to manage a portfolio of emerging businesses. Accelerating managers may want to stay with the business as it grows, take on a leadership role in a functional specialty, or move into other general-management roles in the corporation. Develop such specialized career paths for innovation managers should be in the interest of modern organizations, rather than promoting individuals along with a project as it grows from discovery through to acceleration, and pushing them through different roles requiring different competencies and interests.

6 Conclusion and Outlook

This paper investigates new competence requirements for Innovation Managers in industrial organizations. The applied research methodology is to derive them from key trends in innovation management according to the validated A.T. Kearney House of Innovation framework for innovation management. This framework contains as basic layer the key elements idea management, product/process development, launch/continuous improvement. Major trends identified are the idea value chain, co-innovation of products, processes, and services, as well as a learning organization environment for quickly moving from radical innovation to incremental innovation.

These basic layer elements are facilitated by an adapted innovation organization and culture. Market pressure and sustainability aspects are pushing towards increasingly networked organizations that develop and maintain an Open Innovation culture, aiming at involving a large number of different internal and external stakeholders in the innovation management process. All these trends are to be observed both in the software sector and manufacturing industries.

It is pointed out that radical innovations evolve in the three different phases, discovery, incubation, and acceleration—very similar to an entrepreneurial endeavour. Each of these phases poses particular challenges to the competencies of Innovation Managers, and can launch different career paths. Consequently, the variety of competencies and skills required by job roles associated with successful sustainable innovation management has become very large, making it difficult to define a particular competence profile to be fulfilled by an Innovation Manager as an individual who covers the complete breadth of the competencies.

The collected findings can serve as a guideline for the elaboration of new Innovation Management training and certification programs, which take into account modern innovation management challenges, and the associated competency profiles.

References

1. ECQA Certified Innovation Manager, `http://www.ecqa.org` (last accessed on 22/04/2011)
2. OECD, The Oslo Manual: Guidelines for Collecting and Interpreting Innovation Data, 3rd edn. (2005)
3. Engel, K., Diedrichs, E., Brunswicker, S.: Imp3rove: A European Project with Impact. 50 Success Stories on Innovation Management. Europe INNOVA Paper no. 14, European Union (2010) ISBN 978-92-79-14070-9, `http://www.improve-innovation.eu` (01/04/2011) (retrieved)
4. Hansen, M.T., Birkinshaw, J.: The Innovation Value Chain. Harvard Business Review 85(6), 121–130 (2007)
5. Freeman, R.E.: Strategic Management: A Stakeholder Approach. Pitman, Boston (1984)
6. Hall, J., Vredenburg, H.: The Challenges of Innovating for Sustainable Development. MIT Sloan Management Review 45(1), 61–68 (2003)
7. Stumpfe, J.: Product design and manufacturing process: dynamic implications for innovation management. In: Proceedings of the 19th International Conference of the System Dynamics Society, Atlanta, GA, pp. 1–7. CD-ROM (2010)
8. Tichkiewitch, S., Veron, M.: Integration of Manufacturing Process in Design. Annals of the CIRP 47(1), 99–102 (1998)
9. Tichkiewitch, S., Brissaud, D.: Co-ordination between product and process definitions in a concurrent engineering environment. Annals of the CIRP 49(1), 75–79 (2000)
10. Rose-Anderssena, C., Allena, P.M., Tsinopoulosb, C., McCarthy, I.: Innovation in manufacturing as an evolutionary complex system. Technovation 25, 1093–1105 (2005)
11. Mills, J.A.: A Pragmatic View of the System Architect. Comm 28(7), 708–717 (1985)

12. Riel, A.: Integrated Design – A Set of Competences and Skills required by Systems and Product Architects. In: Riel, A., O'Connor, R., Tichkiewitch, S., Messnarz, R. (eds.) EuroSPI 2010. CCIS, vol. 99, pp. 233–244. Springer, Heidelberg (2010)
13. Aurich, J., Fuchs, C.: An Approach to Life Cycle Oriented Technical Service Design. Annals of the CIRP 53(1), 151–154 (2004)
14. Messnarz, R., Spork, G., Riel, A., Tichkiewitch, S.: Dynamic Learning Organisations Supporting Knowledge Creation for Competitive and Integrated Product Design. In: Proceedings of the CIRP Design Conference 2009, Cranfield, pp. 104–108 (2009) ISBN 978-0-9557436-4-1
15. Riel, A., Tichkiewitch, S., Messnarz, R.: Qualification and Certification for the Competitive Edge in Integrated Design. CIRP Journal of Manufacturing Science and Technology, Special Issue on Competitive Design 2(4), 279–289 (2010)
16. Gomes, J.F.S., de Weerd-Nederhof, P.C., Pearson, A.W., Cunha, M.P.: Is more always better? An exploration of the differential effects of functional integration on performance in new product development. Technovation 23, 185–191 (2003)
17. Jensen, M.B., Johnson, B., Lorenz, E., Lundvall, B.A.: Forms of knowledge and modes of innovation. Research Policy 36(5), 680–693 (2007)
18. O'Connor, G.C.: Open, Radical Innovation: Toward an Integrated Model in Large Established Firms. In: Chesbrough, H., Vanhaverbeke, W., West, J. (eds.) Open Innovation: Researching a New Paradigm. Oxford University Press, Oxford (2006)
19. Mintzberg, H.: Mintzberg on Management – Inside Our Strange World of Organizations. Simon&Schuster, USA (2003) ISBN 9-780-02-921371-1
20. Chesbrough, H.: Open Innovation: The New Imperative for Creating and Profiting from Technology. Harvard Business School Press, Boston (2003)
21. Rampersad, G.: Managing innovation networks: Exploratory evidence from ICT, biotechnology and nanotechnology networks. In: Industrial Marketing Management (2009), doi:10.1016/j.indmarman.2009.07.002 (2009)
22. Bengtsson, M., Kock, S.: "Coopetition" in Business Networks—to Cooperate and Compete Simultaneously. Industrial Marketing Management 29, 411–426 (2000)
23. Howe, J.: The Rise of Crowdsourcing. In: Wired, retrieved from `http://www.wired.com/wired/archive/14.06/crowds.html` (last accessed on 21/04/2011)
24. Chesbrough, H.W., Garman, A.R.: How Open Innovation Can Help You Cope in Lean Times. Harvard Business Review 87(12), 68–76 (2009)
25. O'Connor, G.C., Corbett, A., Pierantozzi, R.: Create Three Distinct Career Paths for Innovators. Harvard Business Review 87(12), 78–79 (2009)

Adapting the FMEA for Safety Critical Design Processes

Ovi Bachmann[1], Bernhardt Messner[2], and Richard Messnarz[3]

[1] SIBAC GmbH, Germany
info@sibac.de
[2] Performing Facilities GmbH, Austria
b.messner@performing-facilities.com
[3] I.S.C.N. GesmbH, Austria & Ireland
rmess@iscn.com

Abstract. Functional safety standards (ISO 26262, IEC 61508) require a safety life cycle which demands additional design and engineering tasks to be managed. This paper addresses how the existing FMEAs have to be extended and refocused to address and overview signal paths throughout the system. The safety standards require to classify signals with a SIL (Safety Integrity Level) and the higher the SIL the more parallel controls and checks must assure that the signal is correctly calculated, used, and monitored. This paper illustrates this extension of the FMEA using the FMEA to investigate the effect of false sensor signals resulting out of failures in software monitoring functions and false failure reactions on system level resulting out of either false sensor signals or failures within the diagnostic software. AS a complementary activity to the FMEDA a FMEA method is introduced that allows an analysis during the development process that is performed prior to the “in-use” FMEDA.

Keywords: Functional safety, FMEA, signals, extended safety design process.

1 Introduction

In electronically controlled Systems there is a trend to put safety functionality that formerly was hidden in the mechanical or hydraulic part of the system into the electronics and software. At the same time the possibility to monitor the mechanical and the hydraulic parts with the help of software and carry out failure reactions can increase the availability of the system to a large extent. These reasons lead to a permanently rising complexity and overall number of codes of the monitoring software.

In order to be able to handle the complexity and ensure the safety, a number of processes and methods have been tested and are demanded for example in the ISO 26262 standard [1], [2], [4] asking for a design process and requiring the coverage of specific methods in that process. This, for instance, includes the recommendation for an inductive and a deductive method. The design process and how to handle these two methods will be introduced in this paper, showing that they can fulfil the requirements of the design process definitions in ISO 15504 (SPICE) [3] as well as ISO 26262, both being mandatory for the automotive industry [4].

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 290–297, 2011.

The FMEA has the advantage that the mechanical and hydraulic system is usually being investigated in depth. By building signal paths in the same FMEA it is possible to link all failures along the signal path to the system-failures which again directly lead to the severity of such a failure. This again can be used to determine the safety relevant functions with an estimation of the safety ranking. By building function trees that start on the system level and branch down into all the sub systems with a link to an affiliated test for every function the demands of the Eng. processes of SPICE can be met and at the same time spare for example the FTA otherwise demanded by ISO 26262.

2 Extending the FMEA Method for Safety Design Processes

2.1 The Traditional FMEA Method and Meaning

Within the product development life cycle processes in Automotive industry the FMEA is a commonly used method to find functions carried out by mechanical and/or hydraulic parts and to identify potential errors and define measures how to avoid them.

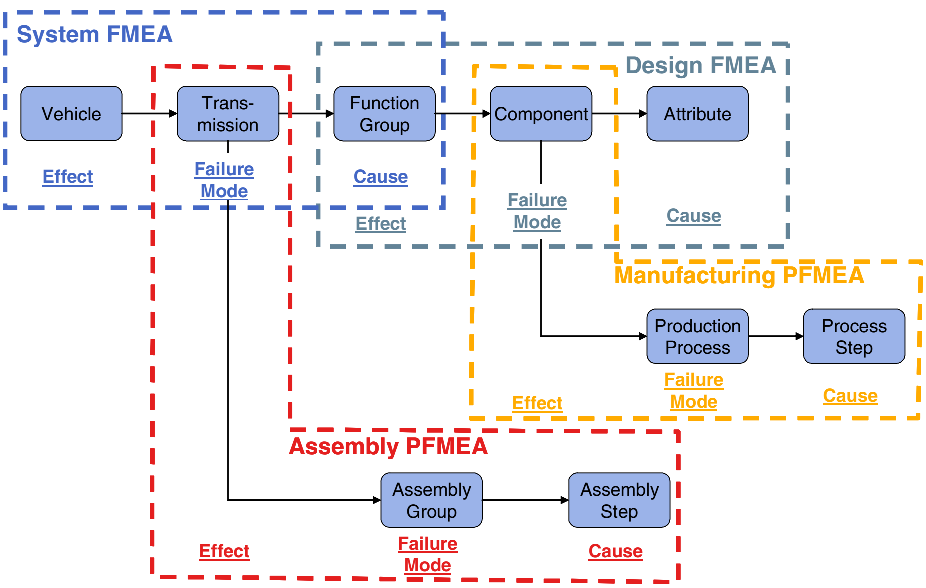


Fig. 1. Failures, effects and causes of a to be analyzed system

The ECU (ECU - Electronic Control-Units) provides the possibility to carry out diagnostic functions to detect mechanical or hydraulic failures and carry out failure reactions. This fact lead to a basic change in the way a FMEA is build. Since then there is a different view on the FMEA during the development phase than during the time after start of production. The time after start of production is described in a so called “in-use” or “field” FMEA.

During the development system failures are analyzed in a way that in a functional tree for each failure three levels of information are analyzed: effect (one level higher), failure mode (in the middle), and cause (one level lower). The severity of failures of the system can be traced by linking the failures to the effects.

In the analysis process the causes for system failures are identified and linked to preventive measures and detection measures. The preventive measures usually are layout measures like using common standards, FEM calculations, etc. Detection actions are tests that should be carried out to prove the layout of the function group that is being analyzed. This way the right layout is ensured during the development as the tests will lead to a loop in the layout, leading to an new test until the test is positively passed.

In this way the FMEA has a big influence on the design process and especially the design of control functionality of a system.

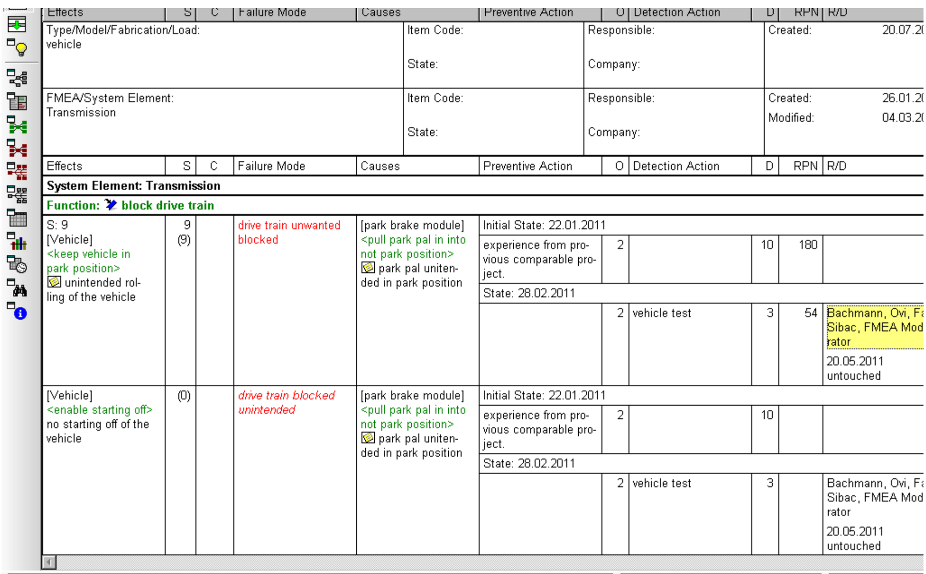


Fig. 2. Preventive- and detection actions in the FMEA

2.2 Extending the FMEA for Safety Design Processes

2.2.1 Introducing Signals Paths in the Process of Analyzing the Functional Architecture

The fundamental idea is to put the ECU in the focus and to investigate monitoring and diagnostic software which bases on signals as inputs, and the diagnosis and monitoring of them. In the abstract design of this architecture it doesn't matter where the signal is coming from. Possibilities are: From Sensors, from the CAN-bus or also calculated Signals coming out of the software itself.

In the design process we build signal workflows (also called signal paths) which are considered in the FMEA.

In a standard electronic control unit that is built according to the rules of the e-gas concept. In this concept there are usually three levels of signal diagnosis and monitoring functions (see Fig. 3).

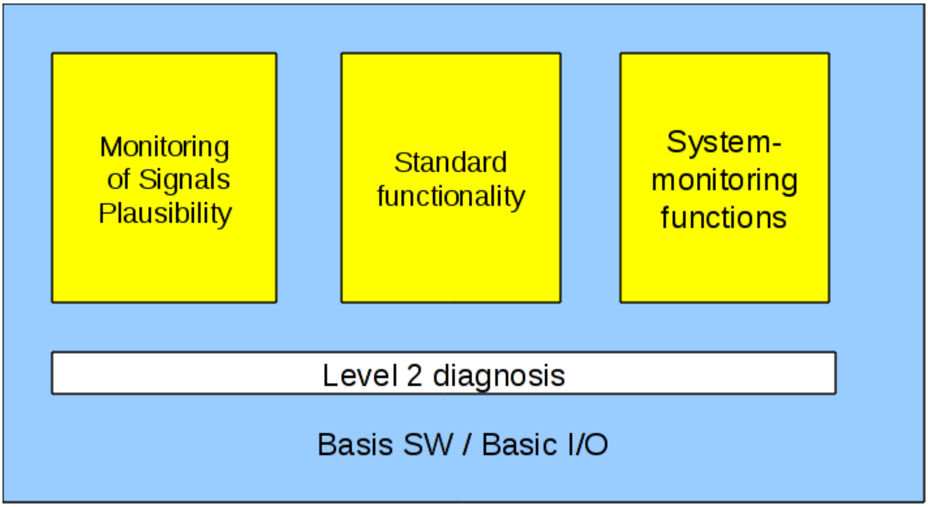


Fig. 3. Three levels of Monitoring and Diagnosis

2.2.2 Process Step 1 for Signal Path Analysis – Signal Transfer to ECU

In the lowest level the FMEA focuses on contains all failures that can possibly occur within the signal transfer from the sensor or the CAN-bus into the monitoring software.

Usually this comprises e.g. checksum, reading out errorflags, etc.. with outcomes such as no signal, or replaced signal (e.g. with a basic value) from the sending control-unit.

And the sensors of the system are checked by the basis software for all detectable electronic failures like short to plus, short to ground, broke cable, etc. All electrical failures are comprised under: wrong signal with error flag. Altogether the following failure cases are used:

- wrong signal with error flag,
- wrong signal not plausible,
- wrong signal plausible,
- no signal.

2.2.3 Process Step 2 for Signal Path Analysis – Plausibility Checks

Within the monitoring part of the software various algorithms are used to perform a plausibility check on the signals forwarded by the basis software to the functional high level software. A rather simple check for plausibility is the check for thresholds which will not be considered here. More sophisticated is the plausibility check using other signals to check one signal for plausibility (redundant control as required by safety standards).

Within the FMEA the signal-paths are built using the function and failure trees. Undetected wrong signals coming out of the monitoring part of the software are only possible, if the signals that come out of the basis software are undetected wrong but nevertheless plausible. Any other combination of wrong signals always leads to a signal with error flag going into the diagnostic part of the software. Following the signal path of signals with an error flag is rather trivial. More important are the signals handed forward to the diagnostic software, that cannot be detected as wrong. These are the ones that are rated as plausible because they are within the thresholds and other signals are not available.

Therefore the failure trees link all the possible failures of the lowest level that are needed for the plausibility check to the signal that is checked on the monitoring level. This way the FMEA shows the dependencies between the signals for the plausibility checks in the monitoring software (see an example in Fig. 4 which shows one signal with one level down all the signals needed for plausibility checks).

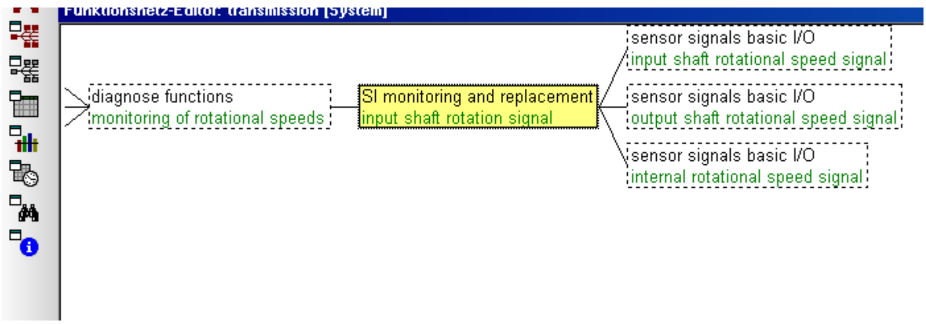


Fig. 4. Signals used to check one signal for plausibility

2.2.4 Process Step 3 for Signal Path Analysis – Independent Control / Diagnosis

The mechanical and hydraulic functions of the systems that are controlled by an electronic control-unit are the ones ordered and perceived by the customer.

This, for instance, includes an active suspension and the adaption of the stiffness of the suspension. In case of an electronic steering system, for instance, the supporting force on the steering wheel, and in case of an automatic transmission, for instance, the shift from one gear to the next. Most of these functions are relevant for the safety and the availability of the system.

There is a difference in the basic understanding going from the monitoring level to the diagnostic level in the FMEA. Whereas on the monitoring level the signals were checked for plausibility, the diagnostic level does not check signals but the mechanical/hydraulic part of the system. The signals that are used as an input to the functions of the software are taken as correct, when they come out of the monitoring part without an error flag. This again means that any failure found on this level must be a mechanical/hydraulic failure in the system. If a wrong signal that is still plausible occurs it will thus lead to a false detection of a failure in the system.

Very interesting for the assessment of the importance of a signal are the effects of a false detection of a failure in the system usually leading to an unwanted failure reaction (see Fig.5)

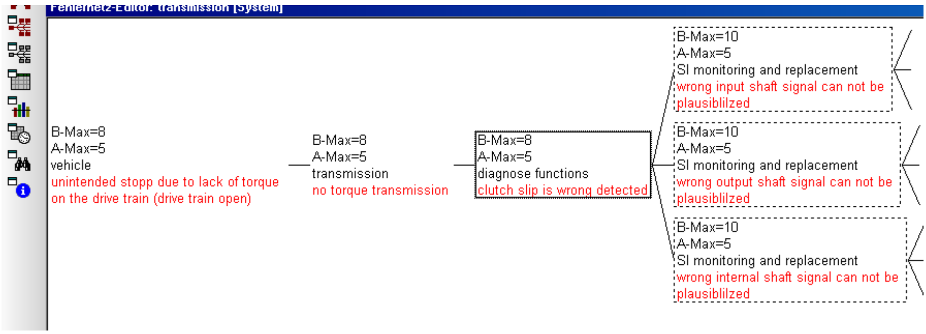


Fig. 5. Unwanted failure reaction because of an unnoticed wrong signal

2.2.5 Reasons for Using This Extended Safety Design Process and FMEA Method

With the above used method every possible failure of a signal and its effect is investigated. Furthermore, looking at the functions and failure trees this approach

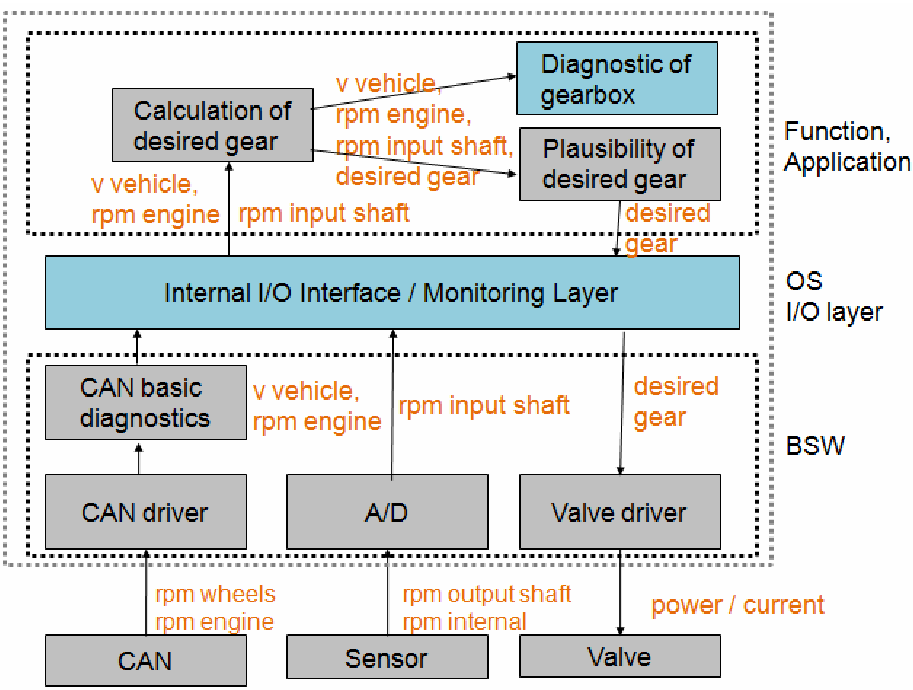


Fig. 6. Signal flow to set desired gear

provides a check for completeness of the signals used for the plausibility checks. As a complementary design process activity to the "in-use" FMEA method looking at diagnostic functions, it is now possible to investigate signals that allow a monitoring on two levels and diagnosing itself on one level. For each level detection and prevention measures are documented within the FMEA form sheet. These measures are typically software tests in case of the detection and usually reviews of specifications or programming guidelines and so on for the prevention.

Using this approach we receive a much more thorough inside view of the system behavior and can thus derive and document more extensive prevention measures increasing the safety and availability of ECU controlled systems.

2.3 Impacts on the System and Software Architectural Design Process

Based on the FMEA and Risk Analysis a signal flow is derived which is represented in the overall software architecture. For every signal path within the FMEA (see Fig. 6) a software scenario is designed and illustrated in the software architecture. This needs to be considered in the software architectural design process.

After designing the software architecture – and keeping in mind the three different levels of monitoring and diagnosis – we can track the safety level (ASIL) for each signal and therefore also for each signal path.

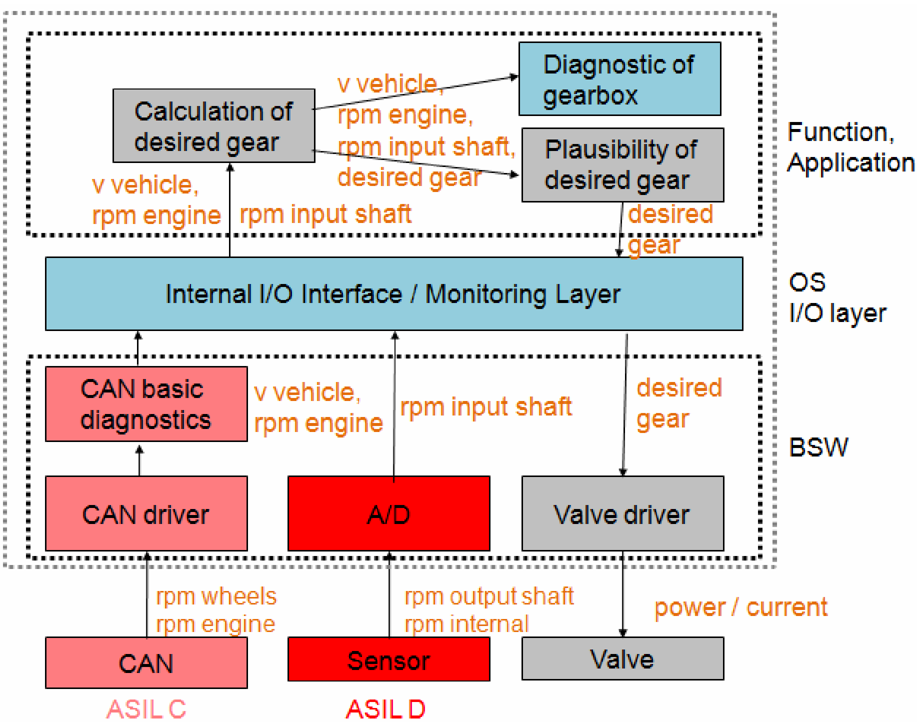


Fig. 7. Path including safety level to calculate the desired gear

This leads to the critical path and determines all software functions (along this path) that have to be assigned as safety relevant.

3 Outlook and Conclusion

The monitoring and diagnosis software is one of the keys to fulfil safety standards like ISO 26262 and development processes required by ISO 15504. The monitoring as well as the diagnosis depend heavily on signals coming either from sensors, CAN or even out of the software itself.

There are several methods to analyze the quality of the signal processing. ISO 26262 for example asks for at least one inductive and one deductive method. This paper shows a methodology with which the signal-paths can be analyzed with a FMEA tool that differs from the commonly known FMEDA and shows crucial benefit concerning the handling in the development phase.

By using the outcome of the FMEA to build function trees as they are needed in order to reach a higher SPICE level one can also spare the failure tree analysis otherwise demanded by the safety standard. In this case the functions on subsystem-level need to marked with the ASIL that was found in the risk analysis and follows the signal-paths found in the FMEA.

By documenting the impact on other subsystems than the software, it is also possible to come to a safe hardware design as well as a safe software-architectural design. This way the basis for higher SPICE levels is also layed.

References

1. IEC 61508 (1998)
2. ISO 26262, Road vehicles — Functional safety
3. Spork, G.: Establishment of a Performance Driven Improvement Program. In: Proceedings of the EuroSPI 2007 Conference (2007)
4. Volker, B., Richard, M.: Improving the Software-Development for multiple Projects by applying a Platform Strategy for Mechatronic Systems. In: Proceedings of the EuroSPI 2009 Conference (2009)

Extending Automotive SPICE to Cover Functional Safety Requirements and a Safety Architecture

Richard Messnarz[1], Ivan Sokic, Stephan Habel[2], Frank König[3], and Ovi Bachmann[4]

[1] ISCN GesmbH, Schieszstattgasse 4, A-8010 Graz, Austria
Tel.: +43 316 811198; Fax: + 43 316 811312
rmess@iscn.com
[2] Continental Automotive, Germany
[3] ZF Friedrichshafen AG, Friedrichshafen, Germany
[4] SIBAC GmbH, Mittelbiberach, Germany

Abstract. This paper discusses (based on Automotive industry examples) how the functional and requirements traceability concepts in Automotive SPICE had to be extended to cover the criteria and content demanded by ISO 26262. In a second section the paper describes how these new concepts are considered in the integrated Automotive SPICE and Safety assessment approach which was proposed by the SOQRATES initiative (www.soqrates.de) where more than 20 leading German firms collaborate in cross company task forces. See previous papers about the integration approach and assessment method proposed in [4]

Keywords: Automotive SPICE, Functional Safety Requirements, Safety Architecture.

1 Functional and Requirements Traceability in Automotive SPICE

Systems became more and more complex over the last 20 years. Most manufacturers in aerospace, cars, trains, etc. illustrate an exponential increase in functionality controlled by a combination of mechanics, electronics, and software. E.g. a function inside a car nowadays is distributed among subsystems (brake, motor, gear system, ESP, etc.) which communicate via a bus and report to a joint failure storage in the car.

To manage the increasing complexity most manufacturers pushed certain methodologies which allow to monitor the functionality, to track the progress of completion and to ensure the coverage in testing [1], [2], [3]. If you are a supplier in fields like automotive, medical, aerospace, etc. you will be asked to implement a requirements management method.

Requirements have to be defined on different levels and linked. This creates a functional / requirements tree.

Car manufacturers link car requirements to systems (delivered by suppliers), system requirements are linked to component requirements (mechanical component, hydraulic component, etc.), and in case of a SW component it is expected that more

R.V. O'Connor, J. Pries-Heje, and R. Messnarz (Eds.): EuroSPI 2011, CCIS 172, pp. 298–307, 2011.

detailed SW module requirements will be derived and traced as well [5], [8]. The idea underlying this requirement tree is to create a structure which allows to do an impact analysis. You have an impact network and if the manufacturer sees a malfunctioning of one of the car functions they can directly address the different components which are affected and influence the fault situation.

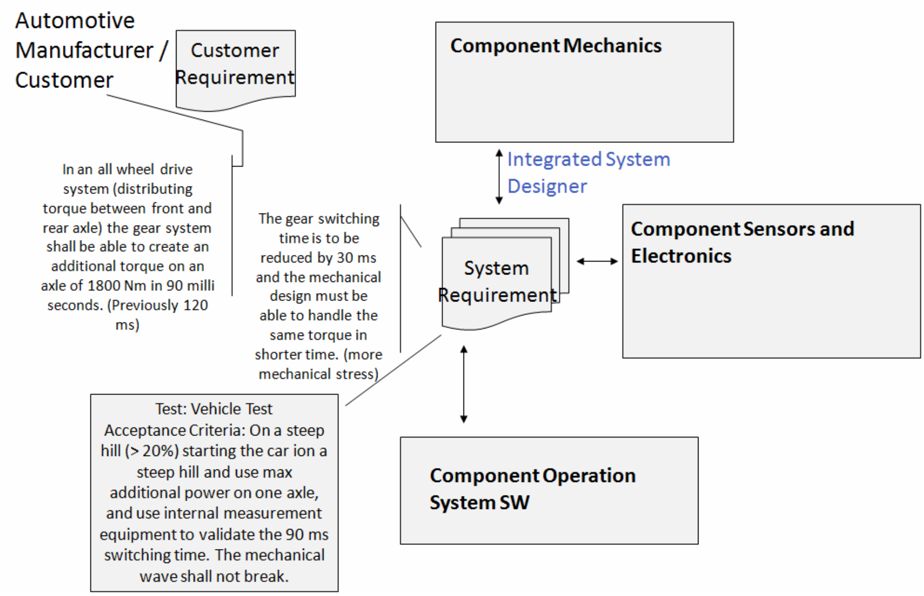


Fig. 1. Example for Analyzing Customer Requirements and Deriving a Linked System Specification

Figure 1 shows an example of an all-wheel-drive system where a dynamic requirement (how fast the system can actuate) is analyzed, mapped onto a system test, and the impact on subsystems is illustrated.

Figure 2 shows an example of how this system requirement impacts the functionality of the mechanical, ECU, and Software subsystems leading to changes in more than one subsystem to satisfy this one system requirement.

Automotive SPICE demands that we can trace the impact of customer to system and down to subsystem functions and vice versa (bilateral traceability).

In practice this means that we have a clear understanding of the functional decomposition and dependencies in the system.

Usually in Automotive industry this is supported by a DOORS or MKS RM tool set which allow to store these documents, link content, and create reports about requirements coverage.

This situation was the starting point when we had to implement the ISO 26262 concepts.

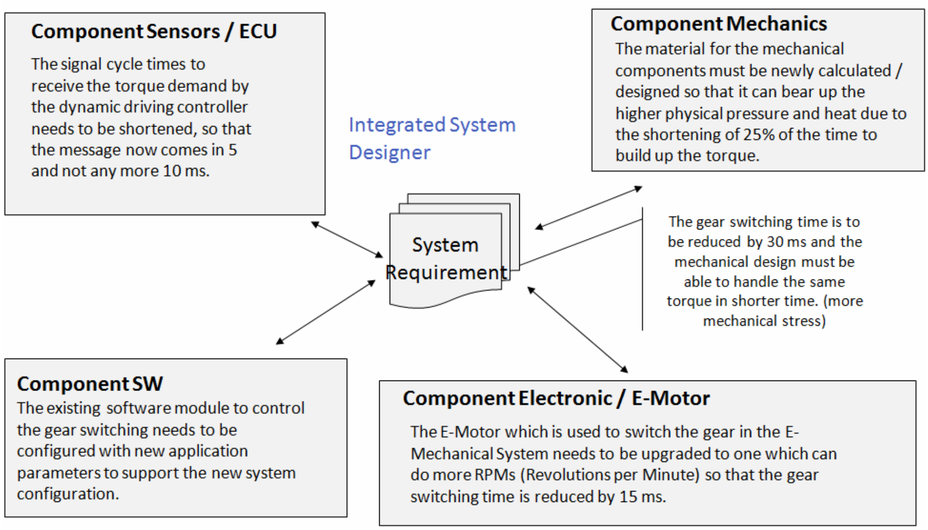


Fig. 2. Example for Analyzing System Requirements and Deriving Linked Software, ECU, Mechanics Requirements

2 Extensions of Automotive SPICE Implemented for ISO 26262

The idea was to add further content and safety related design on the basis of the existing functional understanding and traceability mechanisms of Automotive SPICE [4], [9], [10].

In Figure 3 an Automotive example is illustrated in which the extension is based on a hazard which is derived from an over - temperature of the system.

The customer requirement would generally state that in case of a system temperature > 120 degrees Celsius the system must reduce power by a degradation function. The hazard analysis (an extension of risk management in Automotive SPICE!) would deliver a safety goal “No uncontrolled actuation of the steering system”, and would state that an uncontrolled actuation can happen with a system temperature > 130 °C.

The hazard analysis could lead to an extension of the Automotive SPICE process MAN.5 Risk management. MAN.5 normally addresses project risks but by including FMEAs and FMEDAs and hazard analysis prevention / mitigation actions on product level can be added.

The FMEA and FMEDA (FMEA is already mentioned in MAN.5 Risk management in Automotive SPICE, the tracing of signal paths and analyzing diagnostics software in the FMEA has been added in safety analysis) would deliver a more detailed analysis of the error situation and propose a measure “to implement a redundant temperature control and a safe state to be reached in case the two temperature values differ”. An analysis of the signal-paths showing the severity of a failure in this path together with a required diagnosis will lead logically lead to the redundancy.

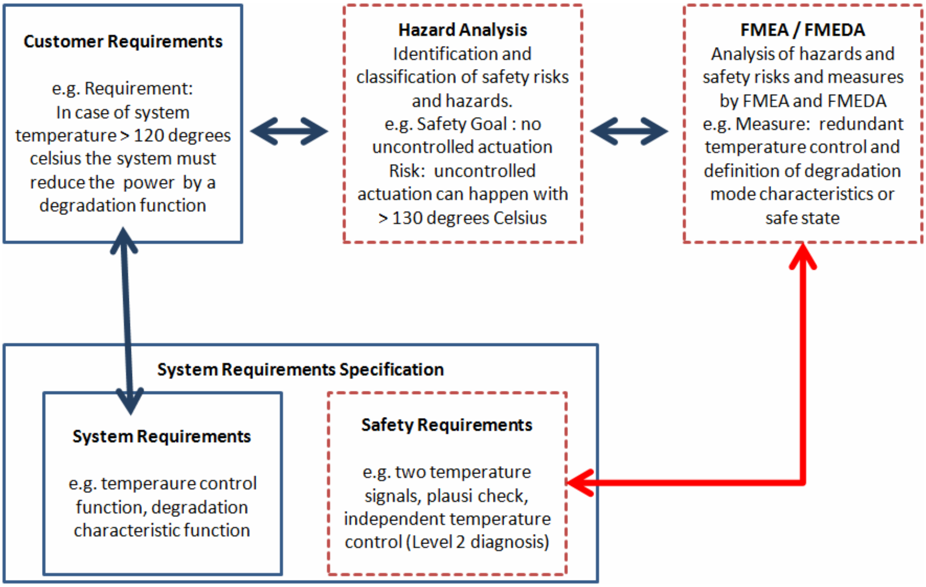


Fig. 3. Strategy Extensions in the process to Derive System Requirements from Customer Requirements

In Figure 4 the previously shown Automotive example illustrates the extension based on a technical safety concept as an integrated part of the overall systems architecture and linked to the system safety requirements.

The system requirements specification would contain safety requirements (new in comparison to Automotive SPICE!) dealing with e.g. the plausibility check of the temperature running in an independent diagnosis control function, while in the technical safety concept (new in comparison to Automotive SPICE!) the details of the control cycle with plausibility check, tolerance values, will be designed, e.g. two analogue temperature sensors with one on the electrical circuit, one on the power amplifier component, a system evaluation that both positions allow a maximum difference of 5 °C, and a plausibility check which leads to a safe state if the difference is above the tolerance threshold for more than a certain time.

New to Automotive SPICE is the consideration of ASIL levels A – D and a method to reduce the ASIL levels and to decrease the probability that the hazard will occur and cannot be controlled. If e.g. a signal is declared ASIL – D it is possible to reduce the ASIL level by making an independent parallel control of that signal and to check whether the signal is plausible or not. Usually an ASIL – D requires to independent processors in the ECU, an ASIL – C requires two independent signals and plausibility checks for important values, and this way an ASIL decomposition usually requires a more expensive design with more parallel independent controls and plausibility checks.

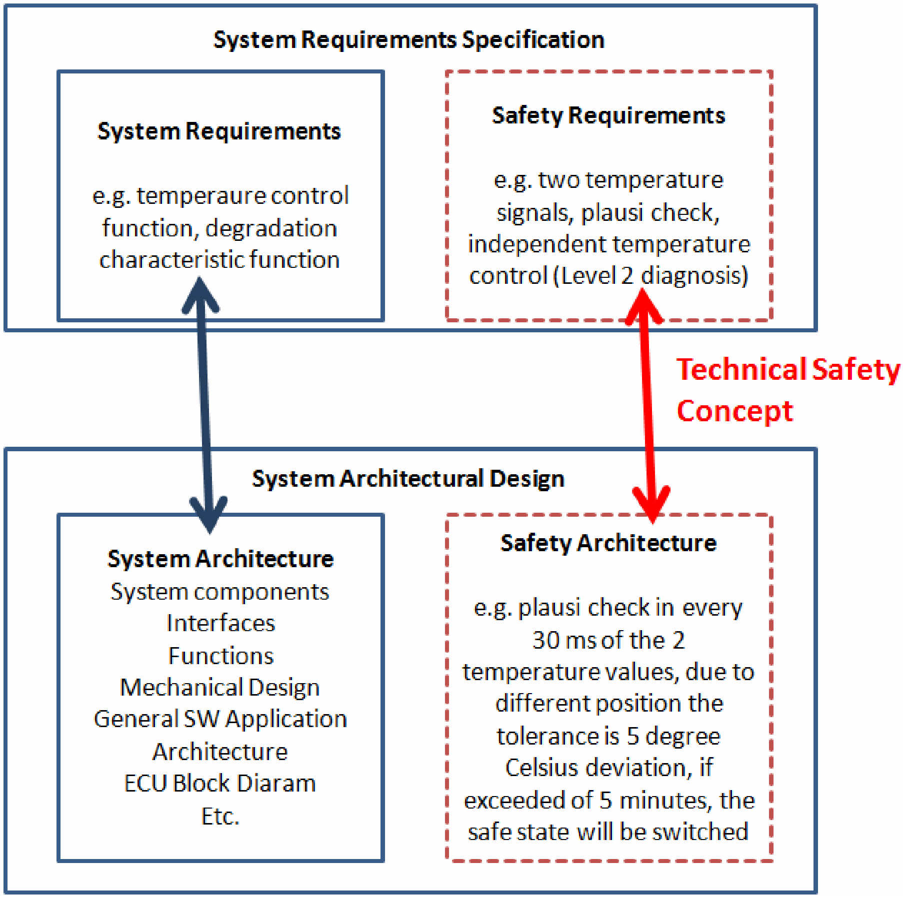


Fig. 4. Extensions in the process to Derive a System Architecture from System Requirements

Figure 5a Part 1 and Figure 5b Part 2 illustrates an Automotive example with such an extension based on a set of safety sub-system requirements as well as a HSI (Hardware-Software-Interface) specification.

In the HW safety requirements specification there will be e.g. two temperature sensors (independent control of temperature and plausibility check), one on the circuit board, and one on the power amplifier. Both would be analogue, with a Volt-Out range from 0.1 to 2 V, and a temperature range from -40 to 130 °C. The ECU would reserve pins 21 and 24 for the 2 sensors.

In the HSI (new in comparison to Automotive SPICE!) specification the signal flow from sensors to the software is described in detail so that we can test and assure the signals. E.g. for the temperature signal 1 we describe the type of sensor (analogue), Volt range, temperature range, scaling algorithm used to calculate the temperature, cycle times to read updates for the software, etc.

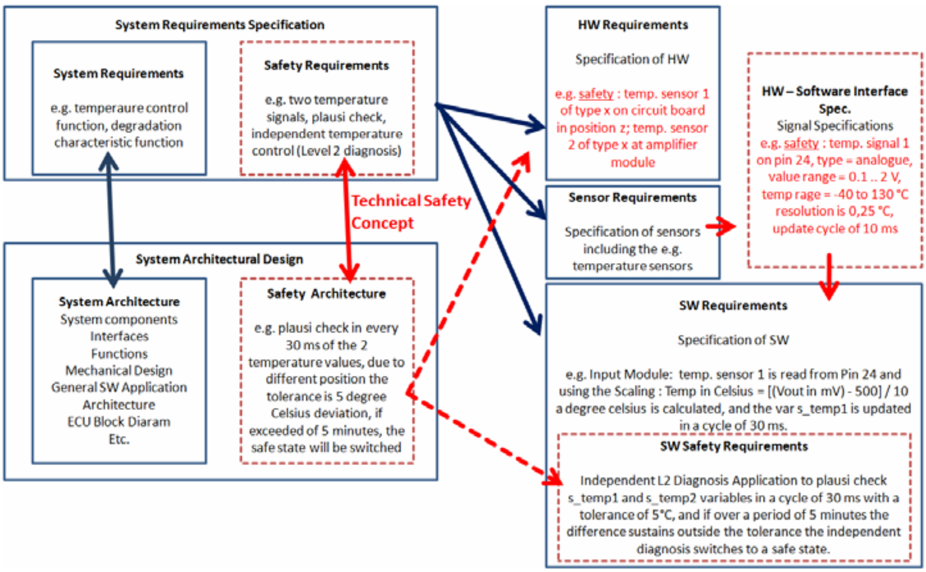


Fig. 5a. Part 1 – Overall Picture of the Extensions in the Process to Derive Sub-System Requirements (see also Part 2)

In the software requirements a special diagnosis function (control function independent from the functional software) is defined which can do the plausibility check of the 2 temperature variables s_temp1 and s_temp2 and switch to a safe state if the difference sustains over e.g. a period of 5 minutes (new in comparison to Automotive SPICE!).

The advantage of the HSI is that it contains a mapping between messages and signals on system level to signal names and variables on software side, thus building a bridge for tracking signal flows from system to software level. In almost all projects where we started to create a specific HSI specification we could solve a number of misunderstandings between system and software level preventively.

This extended structure in Automotive SPICE also has direct impact on the testing strategy. Figure 6 illustrates the impact on e.g. Software Test.

The additional extended HSI specification leads to a more intensive testing approach for the safety critical signal flows. Usually a white box (not black box) test was required to demonstrate the correct signals at the following 4 measurement points.

See M1, M2, M3, M4 marks in the above Figure 6.

M1: Sensor is working, signal is produced with correct (voltage) range and resolution.

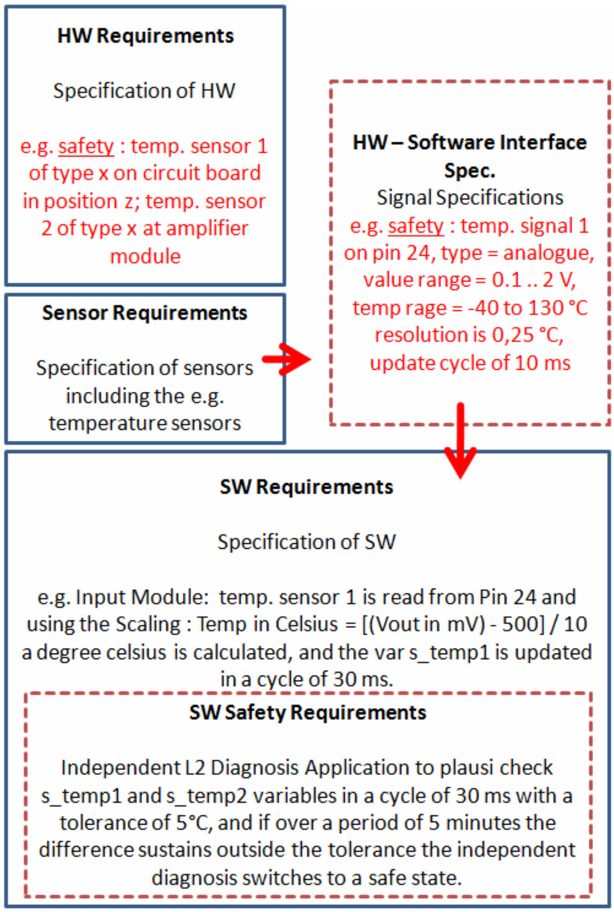


Fig. 5b. Part 2 – Details of the Overall Picture of the Extensions in the Process to dDerive Sub-System Requirements (see also Part 1)

M2: The scaling algorithm is working and signals are correctly transferred into variables, and updated in the correct cycle times in ms.

M3: The plausibility check is working and the correct tolerance is applied within correct latency time.

M4: The correct diagnosis level and program is started to solve the risk, leading to a safe state.

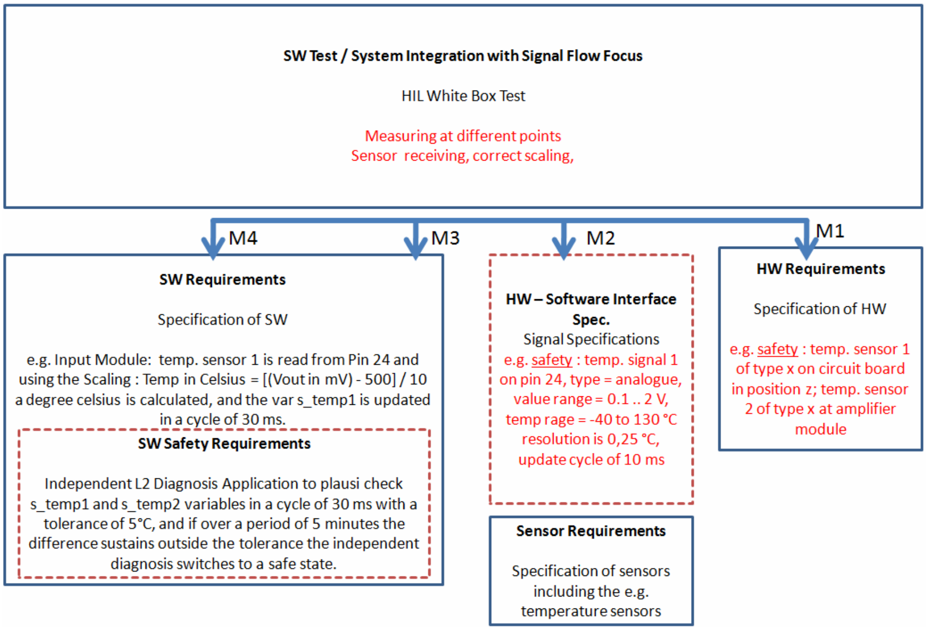


Fig. 6. Extension of SW Testing Procedures based on Safety

3 Assessment of Specific Safety Content Based on the Extensions

In [4] and [10] we described how we mapped Automotive SPICE towards IEC 61508 and ISO 26262.

During an assessment the “Functional Safety View” can be activated with the following effects:

- Base practices will have additional criteria.
- Generic practices include additional criteria.
- New safety practices will appear.
- A safety methods table per process can be opened to consider the use of methods when assessing the practices.

In the functional assessment view we ask about extended safety practices which relate to the previously described strategy of extending Automotive SPICE to cover safety as well.

For instance, in Figure 7 a new safety base practice ENG.3.BP7 has been added by additional requirements originating from IEC 61508 and ISO 26262.

The overall text for ENG.3.BP7 derives from general safety requirements from IEC 61508, while the specific element related text (reference ISO 26262 Part 6.6, chapter 6.6.4.2) describes specific safety requirements stemming from the Automotive specific functional safety standard ISO 26262.

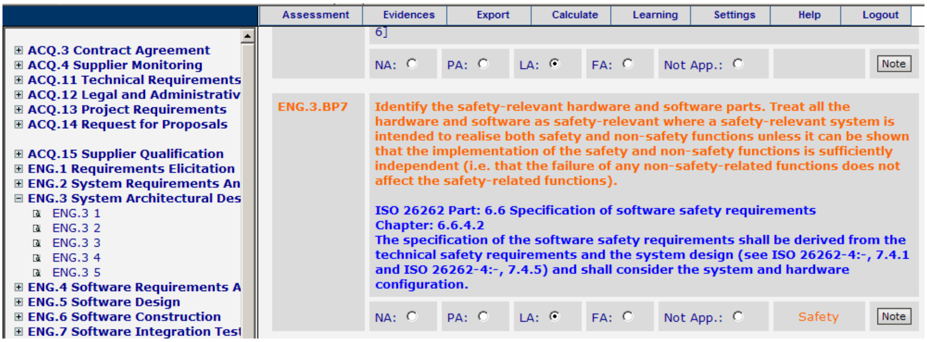


Fig. 7. Activated Functional Safety Views – ISO 26262 Extensions

The ISO 26262 part of the question asks about the aspect illustrated in Figure 5 where software safety requirements are derived from technical safety requirements.

The IEC 61508 part of the question asks about the independence of the control function as described in the technical safety concept as part of the systems architecture in Figure 4.

All extended or new safety base practices in the integrated safety and SPICE assessment are derived from such extensions needed to move from Automotive SPICE to cover safety aspects, as well.

For further assessment approach details please read the articles [5], [10].

4 Lessons Learned

There is a step by step approach to extend existing Automotive SPICE based processes and system designs to also cover aspects and requirements demanded by functional safety standards. It needs consequent extensions of requirements specification, architectures, control mechanisms and tests.

Integrated SPICE and safety assessments become possible if we can create a complete understanding of how to extend systems and specifications from Automotive SPICE to safety architectures and traceability in safety systems as outlined in chapter 2 of this paper.

Acknowledgements and SOQRATES Membership

We are grateful to the experts who have contributed to the Design AK and Safety AK: O. Bachmann (SIBAC), S. Habel, Leo Ross, I. Sokic, R. Dreves (Continental Automotive), F. König, A. Koundoussi (ZF), Bernhard Krammer (Takata Petri AG), G. Spork (Magna Powertrain), M. Haimerl (IMBUS), K. Dussa-Zieger (Methodpark), A. Riel (Grenoble INP), J. Unterreitmayer (SQS), and D. Ekert, R. Messnarz (ISCN).

References

[1] Messnarz, R., et al.: Assessment Based Learning Centers. In: Proceedings of the EuroSPI 2006 Conference, Joensuu, Finland (October 2006); also published in Wiley SPIP Journal (Software Process Improvement in Practice) 12(6), 505–610 (2007)

[2] Messnarz, R., et al.: Better Software Practice for Business Benefit. IEEE Computer Society Press, Washington (1999)

[3] Messnarz, R.: From process improvement to learning organisations. Wiley SPIP Journal (Software Process Improvement in Practice) 11(3), 213–335 (2006)

[4] SOQRATES Safety Team, Messnarz, R., Ross, H.-L., Habel, S., König, F., Koundoussi, A., Unterrreitmayer, J., Ekert, D.: Integrated Automotive SPICE and safety assessments. Wiley SPIP 14(5), 279–288 (2009)

[5] Automotive SPICE, `http://www.automotivespice.com`

[6] ISO 26262, Road vehicles — Functional safety

[7] SOQRATES Initiative, `http://www.sowrates.de`

[8] HIS, `http://www.his-automotive.de`

[9] Volker, B., Richard, M.: Improving the Software-Development for multiple Projects by applying a Platform Strategy for Mechatronic Systems. In: Proceedings of the EuroSPI 2009 Conference, Wiley SPIP Journal (August 2010)

[10] SOQRATES Safety Team, Messnarz, R., Ross, H.-L., Habel, S., König, F., Koundoussi, A., Unterreitmayer, J., Ekert, D.: Integrated Approach for Automotive SPICE and ISO 26262 Assessments. In: Proceedings of the EuroSPI 2009 Conference, Wiley SPIP Journal (August 2010)

Author Index